

**Implementation of ISO/IEC TR 10183-2 :
1993**

**Information technology — Text
and office systems — Office
Document Architecture (ODA)
and interchange format —
Technical Report on ISO 8613
implementation testing —**

Part 2: Framework for abstract test cases

UDC 681.3 : 621.39 : 651.5/.7

National foreword

This British Standard reproduces verbatim ISO/IEC TR 10183-2 : 1993 and implements it as the UK national standard.

This British Standard is published under the direction of the Information Systems Technology Standards Policy Committee whose Technical Committee IST/18 has the responsibility to:

- aid enquirers to understand the text;
- present to the responsible international committee any enquiries on interpretation, or proposals for change, and keep UK interests informed;
- monitor related international and European developments and promulgate them in the UK.

NOTE. International and European Standards, as well as overseas standards, are available from Customer Services, Publications, BSI, Linford Wood, Milton Keynes, MK14 6LE.

Compliance with a British Standard does not of itself confer immunity from legal obligations.

Information technology — Text and office systems — Office Document Architecture (ODA) and interchange format — Technical Report on ISO 8613 implementation testing —

Part 2:

Framework for abstract test cases

Technologies de l'information — Bureautique — Architecture de documents de bureau (ODA) et format d'échange — Rapport technique sur la mise en application de tests de l'ISO 8613 —

Partie 2: Cadre général pour cas de tests abstraits



Reference number
ISO/IEC TR 10183-2:1993(E)



Documents

Document Sharing Hub

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.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

The main task of technical committees is to prepare International Standards, but in exceptional circumstances a technical committee may propose the publication of a Technical Report of one of the following types:

- type 1, when the required support cannot be obtained for the publication of an International Standard, despite repeated efforts;
- type 2, when the subject is still under technical development or where for any other reason there is the future but not immediate possibility of an agreement on an International Standard;
- type 3, when a technical committee has collected data of a different kind from that which is normally published as an International Standard ("state of the art", for example).

Technical Reports of types 1 and 2 are subject to review within three years of publication, to decide whether they can be transformed into International Standards. Technical Reports of type 3 do not necessarily have to be reviewed until the data they provide are considered to be no longer valid or useful.

ISO/IEC TR 10183-2, which is a Technical Report of type 3, was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 18, *Document processing and related communication*.

ISO/IEC TR 10183 consists of the following parts, under the general title *Information technology — Text and office systems — Office Document Architecture (ODA) and interchange format — Technical Report on ISO 8613 implementation testing*:

- *Part 1: Testing methodology*
- *Part 2: Framework for abstract test cases*

© ISO/IEC 1993

All rights reserved. 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 the publisher.

ISO/IEC Copyright Office • Case postale 56 • CH-1211 Genève 20 • Switzerland

Printed in Switzerland

Information technology — Text and office systems — Office Document Architecture (ODA) and interchange format — Technical Report on ISO 8613 implementation testing —

Part 2:

Framework for abstract test cases

1 Scope

The purpose of ISO/IEC TR 10183 is to define a testing methodology and provide a framework for specifying abstract test cases for ISO 8613 implementation testing, the overall objective being the provision of a suitable base for testing the *interworking* capability of ODA implementations.

Such testing will assist in the analysis of an implementation's ability to interwork in an environment of implementations of ISO 8613 and implementations of International Standardized Profiles (ISPs) based on ISO 8613. ISPs are standardized document application profiles that have been internationally harmonized. As such they represent agreed stabilized subsets of ISO 8613 designed for the interworking of ODA systems at different levels of functionality.

In ISO 8613 the term “conformance” refers to the conformance of a data stream to the rules specified in ISO 8613. This includes the conformance of a data stream to a document application profile based on ISO 8613. Conformance testing methodology as defined in Annex G of ISO 8613-1 covers the analysis of data streams without regard to the capabilities of implementations to generate or receive conforming data streams. To achieve an environment of interworking ODA systems, it is necessary to have a testing methodology that can verify implementation support for ISO 8613 and DAP's at the semantic level as well as the data stream or syntax level.

Hence, implementation testing is additional testing that supplements the conformance testing of data streams. It increases the probability that different implementations of ISO 8613 and ISPs are able to interwork. Implementation testing is based on the concept of measuring an implementation's ability to generate and/or receive a representative set of documents. If an implementation can exhibit this capability, then it is more likely to interwork successfully with other verified implementations exchanging a wider range of documents.

The implementation testing methodology introduces the requirement for abstract test cases as well as procedures for their use in the generation and reception testing of implementations.

In establishing a framework for implementation testing, a conceptual model of ODA systems has been developed to describe, in an abstract sense, the multitude of configurations and limitations of real ODA systems.

The methodology contained in ISO/IEC TR 10183 caters for the testing of implementations of ISP's based on ISO 8613. The methodology may also be used for testing other Document Application Profiles based on ISO 8613.

This part of ISO TR 10183

- specifies a framework for the development of abstract test cases;
- specifies a test case notation used to specify abstract test cases;
- gives examples of abstract test cases.

ISO/IEC TR 10183 does not cover the testing of user interfaces in an ODA based system. Any suitable system interface

is only used as a point of control and observation to verify that ODA document transformations associated with the various ODA processes have been carried out as claimed by an implementor.

Abstract test cases created from the specifications in this part of ISO/IEC TR 10183 should be used in the definition of executable test suites and data streams for testing implementations of ISO 8613 and implementations of International Standardized Profiles based on ISO 8613.

Where appropriate, concepts and terminology described in ISO/IEC 9646 have been used. In some cases, definitions and concepts have been adapted to cater for the fact that ODA is not an OSI protocol.

2 References

The following documents are referenced within ISO/IEC TR 10183 and provide additional background information.

ISO/IEC 646:1991, *Information technology - ISO 7-bit coded character set for information interchange (3rd edition)*.

ISO 2022:1986, *Information processing - ISO 7-bit and 8-bit coded character sets - Code extension techniques*.

ISO/IEC 6429:1992, *Information technology - Control functions for coded character sets*.

ISO 8613:1989, *Information processing - Text and office systems - Office Document Architecture (ODA) and interchange format*.

ISO/IEC 9646-1:1991, *Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 1: General concepts*.

ISO/IEC TR 10183-1:1993, *Information technology - Text and office systems - Office Document Architecture (ODA) and Interchange Format - Technical Report on ISO 8613 implementation testing - Part 1: Testing methodology*.

ISO/IEC ISP 10610-1:1993, *Information technology - International Standardized Profile FOD11 - Open Document Format - Simple document structure - Character content architecture only - Part 1: Document Application Profile (DAP)*.

ISO/IEC ISP 11181-1:1993, *Information technology - International Standardized Profile FOD26 - Open Document Format - Enhanced document structure - Character, raster graphics and geometric graphics content architectures - Part 1: Document Application Profile (DAP)*.

ISO/IEC ISP 11182-1:1993, *Information technology - International Standardized Profile FOD36 - Open Document Format - Extended document structure - Character, raster graphics and geometric graphics content architectures - Part 1: Document Application Profile (DAP)*.

3 Definitions

The definitions of ISO 8613 apply to ISO/IEC TR 10183. Some terms from ISO 9646 are also used where there is an equivalence of meaning. Additionally, the definitions given in ISO/IEC TR 10183-1 also apply.

4 Abbreviations

The abbreviations given in ISO/IEC TR 10183-1 apply. Additionally, the following abbreviations are used.

DAP: Document Application Profile.

ISP: International Standardized Profile.

IUT: Implementation Under Test.

PCO: Point of Control and Observation.

Pc: Process component.

Sc: System interface component.

Ic: Interchange component.

GSS: Generating Support Statement.

RSS: Receiving Support Statement.

DUT: Document Under Test

TCS: Test Case Specification (used in generation testing)

TDS: Test Data Stream (used in reception testing)

FDAR: A local representation of a formatted form (FDA) ODA document

PDAR: A local representation of a processable form (PDA) ODA document

FPDAR: A local representation of a formatted processable form (FPDA) ODA document

ATCN: Abstract Test Case Notation

TCID: Test case Identifier

TCP: Test Case Purpose

5 Framework for the development of abstract test cases

This clause presents the framework for the development of abstract test cases for generation and reception testing. The structure of abstract test cases is described in 5.1 and the design of generation and reception test cases is described in 5.2. Examples of test cases are contained in Annexes A and B.

5.1 Structure of abstract test cases

Every abstract test case has a Test Case Identifier (TCID), a Test Case Purpose (TCP) and a test case body. The TCP is a description of the objective that the test case is designed to achieve. The test case body contains a specification, written in the Abstract Test Case Notation (ATCN), for a set of ODA functional elements that are necessary to achieve the TCP and assign verdicts to the possible outcomes. In the case of reception testing, the test case body may also contain a test case script that can be used to aid analysis of a document and help in determination of the test case verdict.

A Document Under Test (DUT), used in generation testing, or a Test Data Stream (TDS), used in reception testing, may contain one or more test cases.

The requirements for ODA test cases are that they be configurable to different DAPs and different Implementations under Test (IUTs). This can be achieved by having test cases which are feature based and which contain the minimal amount of DAP dependent information. These constraints impact on the structure of the test case specification.

The test case specification part of the test case body consists of one or more constituent specifications for one or more ODA structures. The test case specification contains only the information necessary to express the purpose of the test case. One or more test case specifications do not necessarily represent a complete document specification since not all constituents, attributes and values are present. However, any DUT or TDS resulting from the compilation of one or more abstract test cases must be a conforming ODA document.

5.1.1 Relationships between constituents in a test case

The constituent specifications in a test case result in constituents that are connected in a less constraining fashion than in ODA. Two types of relationship describe the possible connections: the hierarchical relationship and the logical-layout relationship.

The *hierarchical relationship* is used to identify particular descendants of a constituent. It allows the definition of a hierarchy of constituents without having to specify all the constituents in a document structure. A constituent's descendant can be defined as being either a subordinate or an immediate subordinate. The order of appearance of descendants in relation to the sequential order can also be specified by the hierarchical relationship.

The *logical-layout relationship* is used to constrain a particular logical constituent and its subordinates to be laid out in a particular layout constituent and its subordinates. This type of relationship allows the guidance of the layout of the logical objects defined in the test case specification without having to specify layout directives which are DAP dependent or a complete document.

5.1.2 Preconditions

There is some ambiguity in allowing missing attributes and attribute values in the test case. This ambiguity occurs when a DUT or TDS is derived for the test case and can result in attributes and attribute values being in conflict with the purpose of the test case. It is necessary to prohibit the occurrence of such attributes and attribute values in any DUT or TDS containing the test case. This can be achieved using preconditions. A *precondition* is a constraint on the presence and values of attributes of particular constituents in any TDS derived for the test case. The constraint can apply to the subordinates of a particular constituent specified in the test case, to the objects which are superior (at any hierarchical level) to a particular constituent specified in the test case, or to the constituents hierarchically situated between two particular constituents belonging to the same ODA structure in the test case.

5.1.3 Attribute value range

In a test case, attribute values must be expressed in a way that allows them to be as unconstrained as possible so that they can be configurable to different DAPs. Attribute values can be specified in an unconstrained way by providing a range of possible values instead of a single value (provided more than one value is possible in the test case). The specification of a range of values in the test case is done in a similar way to the specification of DAPs using the Document Application Profile Proforma Notation in Annex F of ISO 8613-1.

5.1.4 Constituent names

It is often necessary to restrict a constituent specification in a test case to be of a particular type. Normally, in ODA, the type of a constituent is determined by a combination of attributes. In order to facilitate the specification of such a

constraint in the test case, a constraint, called a 'constituent type constraint', is introduced. This type of constraint allows the type of a constituent to be constrained without having to set particular attribute values (which might restrict the test case being configurable to different DAPs).

DAPs assign semantics to the constituents of the possible documents allowed. The constituents represent the allowed structural elements such as objects and object classes in a document for instance paragraphs, sections and columns, each having allowed properties such as indentation, numbering and size. It is also necessary to assign similar semantics to the constituents in a test case. This can be achieved by similarly associating the constituents, specified in a test case, with names identifying their basic functionality (e.g. paragraphs, sections and columns). Since the test cases need to be DAP configurable, the constituent names should represent general elements applicable to different DAPs.

5.1.5 Constraints

It is necessary to ensure that test cases provide a predictable outcome. When implementation testing involves the analysis of a view transformation, the test cases also need to consider an predictable result in the analysis of, for example, the imaged DUT containing the test case. Three types of constraint are necessary to achieve such predictability:

- constraints on the visibility of a layout object;
- constraints on the size of content and
- constraints on the available area of a layout object.

These constraints cannot be realized without knowledge of the DAP and must be expressed in a generic manner to allow the test cases to be DAP configurable. These three constraints are described in more detail in the following clauses.

5.1.5.1 Constraints on visibility

The *visibility* of layout objects is necessary when testing certain features such as 'New Pageset'. Visibility is necessary to ensure that particular content is laid out in particular layout objects. For example, in the case of the layout directive 'New Pageset' applied to some logical objects, it will be necessary to confirm that the content associated with one or more basic logical objects has been laid out in a new pageset. In order to verify this, it is necessary to observe the occurrence of the new pageset in the DUT. The way a layout object can be made visible may vary from DAP to DAP. In test cases, layout objects are designated visible by assigning the characteristic 'VISIBLE' to their constituent. This allows the specification of the visibility constraint in a generic manner and defers the decision on how to realize the constraint to a later phase when the DAP information is used.

5.1.5.2 Constraints on content

The testing of certain features requires the use of content information of specific sizes. In some cases, the content is used to view the dimensions of a layout object in the DUT. In other cases, the content is used to assess the layout process for a given feature in a particular context. The context must be such that the occurrence of the feature produces a predictable effect in the DUT. For example, consider the situation when a TDS for a processable document is being used to test the layout support of a receiving IUT through an imaged DUT. And consider that the TDS contains a test case for the feature that requires two logical objects to be contained in the same layout object. This feature could be observed in an imaged DUT if the processable TDS was such that a new page would occur when the particular logical objects were to be laid out. The recipient's layout process could be tested in such a context by defining three basic logical objects in the test case, the last two being those for which 'Same Layout Object' applies. The content associated with the basic logical objects must be designed in such a way that they do not fit together in the body of one page. It is not possible to achieve this using fixed pieces of content since the available area in the page body, which is DAP dependent, is not known when the generic test case is designed. It is therefore necessary to bind the relative size of the content portions associated with the basic logical objects to the available area of the page body and leave the generation

of the actual pieces of content and hence their size to a later stage when the available area is determined. This type of content is termed *relative content*. In the example, the desired effect can be obtained using relative content by defining three basic logical objects with content occupying respectively one third, one third and one half of the page body's available area. The constraint on the size of the content associated with the basic logical objects ought to force the layout process to generate a new page since the three pieces of content together exceed the page body's available area.

5.1.5.3 Constraints on available area

It is necessary to have a mechanism that effectively determines the size of the available area within particular layout objects. Such control is required in order to ensure that content associated with particular basic logical objects can be laid out in layout objects which were designed, in a given test case, to receive them. It is also necessary for test cases with raster or geometric graphics content to make sure that the available area of the associated layout object is of sufficient size to allow observation of the visible content. In the example in 5.1.5.2, where it was required to ensure that some logical content could be laid out in a layout object, there was no need to define constraints on the available area of the layout object since the relative content could be used. However, there are situations where it is more appropriate to use fixed pieces of content. The use of fixed content allows the incorporation of readable text in imaged DUTs. The readable text can be used as a guide during analysis and determination of the verdict of the test case. When fixed content is specified, there is no link between the size of the content and the size of the receiving layout object's available area, and therefore no guarantee that the content can be laid out in the layout object. It is therefore necessary to include a restriction on the layout object's available area to ensure that the content can actually be laid out in the object. A constraint on the available area of a layout object in a test case specification consists of a minimum and/or maximum size for the horizontal and/or vertical directions of the area.

5.2 Design of abstract test cases

The design of test cases for generation testing and for reception testing is different since the former must create a document specification language which is understandable by an implementor/operator of the IUTs proprietary system. Reception testing must cater for the analysis of DUTs to determine the verdict of test cases. 5.2.1 and 5.2.2 describe the design of generation and reception test cases.

5.2.1 Generation test case design

In generation testing, each test case includes a feature name and a feature description. The feature name is expressed in document processing terms (e.g paragraph, footnote, column) used in clause 6 of DAPs defined in accordance with the Document Application Profile Proforma Notation in ISO 8613-1. The feature description expresses the feature in the abstract test case notation (see clause 7) using ODA constituents and attribute specifications. In the case where a feature can be described using different combinations of ODA functional elements, a set of descriptions can be associated with the feature.

Examples of feature descriptions are contained in Annex A. The examples include test case feature descriptions for 'Paragraph with text and graphic' (A.2) and 'Page with header or footer frame' (A.3).

5.2.2 Reception Test Case Design

A reception test case can be viewed as a constraint on a TDS. The notation used to express the constraint is similar to the one used for generation test cases. However, some functionality is added to the notation to cater for the specific requirements of reception testing.

The reception test cases can be divided in three categories:

- Test cases which check that the IUT can receive and interpret the semantics of attributes and attribute values correctly. The semantics of attributes are specified in ISO 8613;

- Test cases which check that the IUT can receive and correctly process combinations of attributes which affect each other. The result of such interactions between attributes is also specified in ISO 8613;
- Test cases which check that the IUT follows the ISO 8613 defaulting mechanism correctly.

Hence, reception test cases check that the IUT can process test data streams in accordance with the semantics of the ODA architectural model.

In reception testing, the results of the tests are determined from analysis of the DUTs provided by the IUT. Each test case in a TDS must result in a predictable outcome in a DUT. In the design of such test cases effects on the layout process need to be anticipated and controlled to some extent.

5.2.2.1 Algorithm for designing a reception test case

When designing a reception test case for a particular ODA requirement, the first step is to determine an environment suitable for the test case. A test case environment is a collection of constituents and attributes, organized in such a manner that, following reception of the associated TDS, the effect of the test case is predictable in a DUT produced by the IUT.

The second step involves representing the test environment in the test case notation. The test case must also express any necessary conditions needed to make the test environment valid. Each condition can identify disallowed attributes or attribute values for certain parts of the document if their inclusion would cause a conflict with the purpose of the test case.

The third optional step consists of writing a script indicating how to determine the verdict from DUT analysis. Scripts should indicate how to determine the verdict of a test case in a way that is, as far as possible, independent of the DUT representation. Because of the differences of equipment and applications, scripts should be written in general and flexible terms stating only the minimum requirements necessary to satisfy the purpose of the test cases.

Some further principles should be followed when designing reception test cases:

- A reception test case should not specify a complete document or document structure. The test case specification should only include what is needed to test the purpose and should be as general as the requirement will allow. For instance, a test case for the binding reference 'B_REF (SUP (CURR_OBJ))' ("binding name") should not specify that the binding "binding name" be in an immediately superior object but rather in any superior of an object. This approach allows the test case to be more flexible.
- Scripts should indicate how to determine the verdict of a test case in a way that is, as far as possible, independent of the DUT representation. Expected results should be described in a relative manner by comparison rather than in absolute terms. In the case where the results must be described in an absolute manner, the script should specify a range of acceptable values rather than precise measures. This tolerance margin is necessary because of equipment precision differences and DAP tolerances. For example, when assessing a DUT containing a test case for the attribute "Position", the script might state "object B is below and to the left of object A" rather than "object B is positioned at 200, 500".

6 Abstract Test Case Proforma

This clause defines a proforma for ODA abstract test cases. The objectives of the proforma and notation are:

- to allow the specification of non-ambiguous implementation testing abstract test cases;
- to allow the configuration of abstract test cases to different DAPs;
- to provide test cases which are both human-readable and machine-processable.

The test case proforma consists of up to four sections:

- Section 1 Abstract Test Case Identifier
- Section 2 Abstract Test Case Purpose
- Section 3 Abstract Test Case Specification
- Section 4 Abstract Test Case Script

6.1 Abstract Test Case Identifier

This section contains an identifier that uniquely identifies the abstract test case.

6.2 Abstract Test Case Purpose

This section gives a human-readable description of the abstract test case. It is intended to help understand the test case and hence facilitate its use. The purpose also provides the test case intention and any assumptions. This information is useful when verifying the correctness of the test case.

6.3 Abstract Test Case Specification

This section provides the notational specification of the test case using the ATCN as specified in clause 7. The ATCN is machine processable and therefore suitable in a test document generation process that can generate complete documents containing one or more test cases. These documents can then be used during reception testing as part of an implementation test system that can generate Test Data Streams.

6.4 Abstract Test Case Script

This section is optional and used only for reception test cases. It provides a human readable description of the test case that can be used to determine the test case verdict. Such scripts should contain both general and specific requirements. The general requirement for an IUT is that the provision of a suitable system interface for the test operator in order that the DUT can be verified correctly. The specific requirements for the IUT are based on the particular test case under consideration.

The content of the text should include a list of checkpoints. Each checkpoint describes what predictable effects are expected in a DUT and indicates how these effects impact on the verdict of the test case.

7 Abstract Test Case Notation

There is some commonality between the specification of DAPs and the specification of test cases. Both types of specification define constraints on what is allowed by ODA. The Abstract Test Case Notation used in ISO/IEC TR 10183 is based on the DAPPN in Annex F of ISO 8613-1. Two types of revisions have been made to the DAPP notation in deriving the ATCN. The first removed functionality which was not applicable to test case specifications and the second added extensions based on constructs already defined in the DAPPN. Unless otherwise stated, constructs used in the DAPPN have the same semantics when used in the ATCN apart from the fact that they are constraints on document specifications rather than DAP specifications.

7.1 Overview

A test case specification consists of a set of constraints on an ODA document. The constraints can be used by an implementation test system during reception testing as part of the process that determines a test document or during generation testing as part of the process that checks the presence of test cases in a DUT received from the IUT. The

constraints are specified using one or more constraint definitions. There are two types of constraint definitions: constituent constraints and factor constraints.

A constituent constraint is composed of two elements: an environment section and an attribute constraint specification. The purpose of the attribute constraint specification is to define constraints on the presence and values of attributes in the constituent. The purpose of the environment section is to set necessary restrictions on the relationship of the constituent to other constituents in any document containing the test case. The restrictions can take the form of prescribed relationships between the constituent and other constituents, references to preconditions on the presence and values of attributes of one or more constituents in a document and any other characteristics necessary to achieve predictability of the DUT. Additionally, the environment section can also be used to place restrictions on the type of a constituent.

Factor constraints are used to specify preconditions or to factorize information. A factor constraint consists of an attribute constraint section, as specified in constituent constraints. The factorization of information in a test case specification is achieved by referencing one or more factor constraints using the factor reference mechanism.

A precondition is a set of constraints used by one or more constituents in the test case to restrict their subordinates, objects which are superior to them, or constituents hierarchically situated between two constituents of the test case. The precondition applies not only to the constituents which are defined in the test case specification but also to the constituents which are added to the test case specification to obtain a complete document. The constraints associated with the precondition are specified using the environment section and the attribute constraint specification of a factor constraint. A factor constraint can be referenced by one or more constituent constraints.

7.2 Definition of the symbols used in the notation

7.2.1 Symbols used in the meta-language

The symbols listed and defined below are used in the production rules contained in 7.3

::=	Used to specify that the string of symbols on its right side is to be substituted for the non-terminal symbol on its left side.
	Used to separate alternatives.
< >	Used as a pair of symbols to delimit a non-terminal symbol.
-- --	Used as a pair of symbols to delimit a comment string.
()	Used as a pair of symbols to delimit a syntactical unit.
[]	Used as a pair of symbols to delimit an optional syntactical unit.
...	Used following a syntactical unit to indicate that the syntactical unit may be occur one or more times.
' '	Used as a pair of symbols to delimit a terminal symbol.

7.2.2 Terminal symbols used in the notation

The following is a list of terminal symbols which are used in the notation.

...	Used to specify that an attribute or parameter value may be repeated one or more times.
{ }	Used as a pair of symbols to delimit a syntactical unit.
[]	Used to delimit an optional syntactical unit.
()	Used to delimiter parameters in functions and macros.
< >	Used as a pair of symbols to delimit a non-terminal symbol.
-- --	Used as a pair of symbols to delimit a comment in an abstract test case.
	Used to separate alternatives.
,	Used to separate attribute value specifications.
..	Used to specify a range of integers.
#	Used to announce a parameter.
" "	Used as a pair of symbols to delimit a printable character string. If a double quotation mark is to be used within the character string then it shall be announced with another double quotation mark.
' '	Used as a pair of symbols to delimit token words such as enumerated attribute values and terminal symbols in ISO 8613.
\$	Used to announce a macro substitution.
>	Used to specify greater than.
<	Used to specify less than.
>=	Used to specify greater than or equal to.

<=	Used to specify less than or equal to.
\\	Used as a pair of symbols to delimit a non-printable character string.
/	Used to separate two hexadecimal numbers representing an octet.
ABSTRACT	Used to specify a constraint on the name of a constituent abstract feature associated with the constraint definition.
AGG	Used to specify an unordered set of elements with no additional elements.
AGG*	Used to specify an unordered set of elements with additional elements allowed.
ALT	Used to specify possible construction expressions that result in an instance from a choice of the specified object classes, where the instance cannot be preceded by or followed by other objects.
ALT*	Used to specify possible construction expressions that result in an instance from a choice of the specified object classes, where the instance can be preceded by or followed by other objects.
ANNOUNCER	Used to specify an announcer as defined in ISO 2022.
ANY_INTEGER	Used to specify any integer value allowed by ODA and an associated DAP.
ANY_LEVEL	Used to specify a subordinate of an object or object class description where each intermediate subordinate has no siblings.
ANY_LEVEL*	Used to specify a subordinate of an object or object class description where each intermediate subordinate may have siblings.
ANY_STRING	Used to specify any character string allowed by ODA and an associated DAP.
ANY_VALUE	Used to denote that any attribute or (sub-) parameter value may be specified that is permitted in ISO 8613 and an associated DAP.
ASN.1	Used to announce an ASN.1 value.
ATTRIBUTE	Used to announce a particular attribute or parameter in a particular constituent.
AVAILABLE_AREA	Used to specify a constraint on the available area of a layout object associated with the constraint definition.
BMU	Used to announce a generic measurement.
BOUNDED_PRE_CONDITION	Used to specify pre-conditions which apply to the objects hierarchically situated between two constituents associated with two particular constituent constraints.
CASE ... OF	Used to specify a conditional selection that depends on some referenced value.
CHARACTER	Used to denote any character content information allowed by ODA and an associated DAP.

CHO	Used to specify a set of alternative elements.
CONSTITUENT_TYPE	Used to specify a constraint on the type of constituents associated with the constraint definition.
CONTENT	Used to specify a piece of content the size of which is dependent on the size of the area which the piece of content must fill when it is laid out.
CONTENT_ID_OF	Used to specify implicitly the content portion identifiers of the set of instances of a particular constituent constraint.
DEFINE	Used to announce a string substitution macro.
DIS	Used to specify that a defaultable or non-mandatory attribute/parameter must not be specified in an associated constituent/attribute.
ENVIRONMENT	Used to specify restrictions on the environment of the constituents associated with the constraint definition.
EXCEPT	Used to specify values not allowed for a particular attribute or parameter.
FACTOR	Used to announce a common set of constraints that apply to one or more constituent constraints either directly or via the ODA defaulting mechanism.
FIXED	Used to specify a fixed measurement for the size of an area occupied by content information.
GEOMETRIC	Used to denote any geometric content information allowed by ODA and an associated DAP.
HORIZONTAL	Used to announce the horizontal dimension of an area.
ITER	Used to specify possible construction expressions that result in one or more instances of the specified object class, where the instances cannot be preceded by, interspersed with or followed by other objects.
ITER*	Used to specify possible construction expressions that result in one or more instances of the specified object class, where the instances can be preceded by, interspersed with or followed by other objects.
LAID_OUT_IN	Used to specify a relationship between a constituent in the logical structure and a constituent in the layout structure. That is, the logical constituent will be laid out in the specified layout constituent.
NOVR	Used to specify that the attribute value is specified on another constituent and its value may not be overridden.
MAX	Used to announce the specification of the maximum size of the horizontal or vertical dimension of an area.
MIN	Used to announce the specification of the minimum size of the horizontal or vertical dimension of an area.
MUL	Used to specify a group of parameters that occur one or more times.
OBJECT_CLASS_ID_OF	Used to specify any object class identifier from the set of instances of a particular constituent constraint.