

This is a preview of "ISO/IEC 10744:1997". [Click here to purchase the full version from the ANSI store.](#)

Second edition
1997-08-01

Information technology — Hypermedia/Time-based Structuring Language (HyTime)

*Technologies de l'information — Langage de structuration
temporelle/hypermédia (HyTime)*



Reference number
ISO/IEC 10744:1997(E)

Contents

1	Scope	1
1.1	Definition of scope	1
1.2	Field of application	2
2	Normative references	2
3	Definitions	3
4	Symbols and Abbreviations	16
5	Notation	17
5.1	RCS name, full name, description, and clause	18
5.2	Lexical type	18
5.3	Constraints	18
5.4	Note	18
5.5	Associated attribute forms and attribute lists	19
5.6	Referrers	19
5.7	Conventions for attribute form declarations	19
5.8	Identification of optional facilities	19
6	Base module	20
6.1	Concepts and definitions	20
6.1.1	Object representation	20
6.1.1.1	Entity structure	21
6.1.1.2	Data	21

© ISO/IEC 1997

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 the publisher.

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

Printed in Switzerland

6.1.2	Object identification and addressing	22
6.1.2.1	Name space addressing	22
6.1.2.2	Coordinate addressing	22
6.1.2.3	Semantic addressing	23
6.2	Hyperdocument management facilities	23
6.2.1	Object representation	23
6.2.2	Object identification and addressing	24
6.2.3	Object access	24
6.2.4	Bounded object set (BOS)	24
6.2.5	Hyperdocument interchange format	27
6.2.5.1	SDIF packer	27
6.2.5.2	SDIF unpacker	27
6.3	HyTime support declarations	28
6.4	HyTime document	31
6.5	HyTime Bounded object set	32
6.5.1	HyTime bounded object set attributes	32
6.5.2	HyTime BOS control data attributes	34
6.5.3	Bounded object set exception specification	35
6.6	HyTime architectural bridging forms	37
6.7	Common attributes	38
6.7.1	Value Reference	39
6.7.2	Descriptive text	41
6.7.2.1	Descriptive text attributes	41
6.7.2.2	Description table	42
6.7.2.3	Descriptive text	43
6.7.2.4	Descriptive text definition	43
6.7.3	Activity policy association	43

6.8	Coordinate Specifications	51
6.8.1	HyTime axis marker list notation	51
6.8.1.1	Axis marker list	52
6.8.1.2	Marker Functions	52
6.8.2	HyTime dimension specification notation	53
6.8.3	Dimension Specification	54
6.8.4	Dimension List	55
6.8.5	Overrun handling	56
6.8.6	HyTime Marker Function Language (HyFunk)	57
7	Location address module	58
7.1	Concepts and definitions	58
7.1.1	Location types	59
7.1.2	Location Sources	60
7.1.3	Location Paths	61
7.1.4	Groves and Location Addressing	62
7.1.4.1	Grove Plan	63
7.1.4.2	HyTime Default SGML Grove Plan	66
7.1.4.3	Effective SGML Document Grove Plan	67
7.1.4.4	Grove Definition Elements	68
7.2	Location source	69
7.3	Implied location source	70
7.4	Multiple location	72
7.5	Tree type	72
7.6	Span Location Address	73
7.7	Reference control	74
7.7.1	Reference element type	75
7.7.2	Reference resolution range	76

7.7.3	Reference resolution level	76
7.8	Reference location address	77
7.9	Name-space locations	81
7.9.1	Identified local element or entity	81
7.9.2	Property location address	81
7.9.3	Name-space location address	82
7.9.4	Mixed location address	84
7.9.5	Named location address	84
7.9.6	Name list specification	85
7.10	Coordinate locations	86
7.10.1	Node locations	86
7.10.1.1	Node lists	86
7.10.1.2	List location address	87
7.10.1.3	Tree combination	87
7.10.1.4	Tree location address	88
7.10.1.5	Path location address	89
7.10.1.6	Relative location address	90
7.10.2	Data location address	92
7.11	Querying	98
7.11.1	Query location address	98
7.11.2	Name list query	99
7.12	Bibliographic location address	100
8	Hyperlinks module	101
8.1	Concepts and definitions	101
8.1.1	Link creation	102
8.1.2	Link traversal	103
8.1.3	Traversal Rules	104

8.2	Hyperlink architectural forms	107
8.2.1	Hyperlink	107
8.2.2	Contextual link	110
8.2.3	Aggregation link	111
8.2.4	Variable link	112
8.2.5	Independent link	114
8.3	Hyperlink-related location addresses	115
8.3.1	Hyperlink location address	115
8.3.2	Hyperlink anchor location address	116
9	Scheduling module	118
9.1	Scheduling concepts and definitions	118
9.2	Measurement units	118
9.2.1	Measurement domain definition	119
9.2.2	HyTime granule definition notation	121
9.2.3	Useful measurement domains	121
9.2.3.1	Common Standard Measurement Units	122
9.2.3.2	Measurement domain definitions	122
9.2.3.3	Other standard measurement units	125
9.3	Finite coordinate space	127
9.3.1	Axis calibration	129
9.4	Scheduling and extents	130
9.4.1	Schedules	130
9.4.2	Extent specification	132
9.4.3	Group extent specification	132
9.4.4	Scheduled extent	136
9.4.5	Scheduled extent list	138
9.4.6	HyTime extent list notation	138

9.5	Event schedule	139
9.5.1	Scheduled Event	140
9.5.2	Event group	142
9.6	Objects	143
9.7	Pulse maps	144
9.8	Dimension referencing	144
9.8.1	Implicit dimension reference	144
9.8.2	Explicit dimension reference	145
9.8.2.1	Referencing dimensions of directly scheduled events, modscopes, and/or proscopes	146
9.8.2.2	Referencing dimensions of indirectly scheduled events, modscopes, and/or proscopes	152
9.9	Calibrated real time axes	154
9.9.1	HyTime calendar specification notation	154
9.9.2	Calendar specification	155
9.10	Finite coordinate space location address	157
10	Rendition module	161
10.1	Common rendition attributes	161
10.1.1	Precision of Selection	161
10.2	Object Modification	163
10.2.1	Object modifier	163
10.2.2	Direct association of modifiers (modifier rule)	163
10.2.3	Association of modifiers by position in finite coordinate spaces	164
10.2.3.1	Wand Rule	164
10.2.3.2	Wand	166
10.2.3.3	Modifier scope	166
10.2.3.4	Modifier scope group	167
10.2.4	Modifier Patch and Wand Patch	168

10.3	Projection	169
10.3.1	Projector	170
10.3.1.1	HyTime Projector Notation	172
10.3.1.2	Extent Projector	173
10.3.1.3	HyTime Extent Projector Notation	174
10.3.1.4	Dimension Projector	175
10.3.1.5	HyTime Dimension Projector Notation	176
10.3.2	Direct association of projectors	176
10.3.2.1	Projection of modified and unmodified objects	177
10.3.2.2	Projector Rule	177
10.3.2.3	Projector sequence	178
10.3.3	Association of projectors by position in finite coordinate spaces	179
10.3.3.1	Baton rule	179
10.3.3.2	Baton	180
10.3.3.3	Projector scope	181
10.3.3.4	Projector scope group	181
10.3.3.5	Baton sequence	182
10.4	Rendition rule	182
11	Conformance	183
11.1	Conforming HyTime document	183
11.1.1	Basic hyperlinking HyTime document	183
11.1.2	Basic scheduling HyTime document	184
11.1.3	Minimal HyTime document	184
11.1.4	Minimal scheduling HyTime document	185
11.2	Conforming HyTime application	185
11.2.1	Application conventions	185
11.2.2	Conformance of documents	186

11.2.3	Conformance of documentation	186
11.3	Conforming HyTime system	186
11.3.1	Conformance of documentation	186
11.3.2	Conformance to HyTime system declaration	186
11.3.3	Support for minimal HyTime documents	186
11.3.4	Application conventions	186
11.4	Validating HyTime engine	187
11.4.1	Error recognition	187
11.4.2	Identification of HyTime messages	187
11.4.3	Content of HyTime messages	187
11.5	Documentation requirements	187
11.5.1	Standard identification	188
11.5.2	Identification of HyTime constructs	188
11.5.3	Terminology	188
11.6	HyTime system declaration	188
A	SGML Extended Facilities	191
A.1	Introduction	191
A.1.1	Conformance	191
A.1.1.1	Application conventions	191
A.1.1.2	Conformance of documents	192
A.1.1.3	Conformance of documentation	192
A.1.1.3.1	Standard identification	192
A.1.1.3.2	Identification of Extended Facilities constructs	192
A.1.1.3.3	Terminology	193
A.1.1.3.4	Application conventions	193
A.2	Lexical Type Definition Requirements (LTDR)	193
A.2.1	Lexical type set	193

A.2.1.1	Lexical types	194
A.2.1.2	Lexicographic ordering	195
A.2.1.3	Additional lexical constraints	196
A.2.2	Lexical model notations	196
A.2.3	HyTime lexical model notation (HyLex)	197
A.2.3.1	Syntax	198
A.2.3.2	Normalized HyLex models	198
A.2.3.3	Intrinsic lexical types	199
A.2.4	Lexicographic ordering definition notations	212
A.2.4.1	HyTime lexicographic ordering definition notation (HyOrd) ...	212
A.3	Architectural Form Definition Requirements (AFDR)	213
A.3.1	Enabling architectures	214
A.3.1.1	Architectural forms	214
A.3.1.2	Architectural document	215
A.3.2	SGML conventions	215
A.3.2.1	Element forms	216
A.3.2.1.1	Element type declaration	216
A.3.2.1.2	Meta-DTD	217
A.3.2.1.3	Attribute definition list declaration	218
A.3.2.2	Attribute forms	218
A.3.2.3	Attribute list conventions	219
A.3.2.3.1	Default value prescription	219
A.3.2.4	Processing link attributes	220
A.3.3	Architecture base declaration	220
A.3.3.1	Enabling architecture use of APPINFO parameter	221
A.3.4	Architecture support declarations	221
A.3.4.1	Architecture notation declaration	222

A.3.4.2	Architecture support attributes	222
A.3.4.3	Architecture entity declaration	225
A.3.5	Architecture control attributes	227
A.3.5.1	Architectural form attribute	227
A.3.5.2	Architectural attribute renamer	228
A.3.5.3	Architecture suppressor attribute	229
A.3.5.4	Architecture ignore data attribute	230
A.3.6	Other architecture-related considerations	230
A.3.6.1	Architectural document element	230
A.3.6.2	Architectural markup minimization	231
A.3.6.3	Derived enabling architectures	232
A.3.6.4	Relating applications and architectures	232
A.3.7	Summary of AFDR support options	233
A.3.8	Conformance	233
A.3.8.1	Conformance of meta-DTDs	233
A.3.8.2	Conformance of documents and derived meta-DTDs	233
A.3.8.3	Conforming architecture engine	233
A.3.8.3.1	Conformance of documents	234
A.3.8.3.2	Conformance of documentation	234
A.3.8.3.3	Application conventions	234
A.3.8.4	Validating architecture engine	234
A.3.8.4.1	Identification of architecture messages	234
A.3.8.4.2	Content of architecture messages	234
A.3.8.5	Architecture system declaration	234
A.4	Property Set Definition Requirements (PSDR)	235
A.4.1	Concepts and terminology	235
A.4.1.1	Property sets	236

A.4.1.2	Classes and properties	236
A.4.1.3	Nodes	238
A.4.1.4	Groves	238
A.4.1.5	Content trees	239
A.4.1.6	Grove plan application	239
A.4.2	Property set definition architecture	240
A.4.2.1	Shared constructs	241
A.4.2.1.1	Component names	242
A.4.2.1.2	Specification and clause	243
A.4.2.1.3	Descriptive elements	243
A.4.2.1.4	Member of default grove plan	244
A.4.2.2	Modules	244
A.4.2.3	Class definition	245
A.4.2.4	Property definition	246
A.4.2.4.1	Enumerated value definition	248
A.4.2.5	Normalization rule definition	248
A.4.3	Intrinsic properties	248
A.4.4	Useful grove construction processes	250
A.4.4.1	Value-To-Node (VTN) grove construction	251
A.4.4.1.1	The Value-To-Node property set	251
A.4.4.1.2	VTN groves	253
A.4.4.2	Data tokenizer (DATATOK) grove construction	253
A.4.4.2.1	Data tokenizer property set	254
A.4.4.2.2	Data tokenizer notation form	255
A.4.4.3	Plain text (PLAINTEXT) grove construction	256
A.4.4.3.1	Plain text property set	257
A.4.5	Canonical Grove Representation (CGR)	257

A.4.5.1	Canonical grove representation document type	257
A.4.5.2	Constraints on CGR source construction	260
A.4.5.3	Algorithm for assigning IDs to nodes	261
A.4.6	Conformance	262
A.5	General Architecture	262
A.5.1	General Architecture Declaration Template	262
A.5.2	Common attributes of elements	263
A.5.3	Data Attributes for Elements (DAFE)	265
A.5.3.1	Data control attributes	265
A.5.4	Lexical types	266
A.5.5	ID immediate referent type control	267
A.5.6	Default value list	267
A.5.6.1	Default value list attributes	268
A.5.6.2	Default value list element	269
A.5.7	Data attributes	270
A.5.7.1	Common data attributes	270
A.5.8	Conformance	271
A.5.8.1	Conforming General Architecture document	271
A.5.8.1.1	Minimal General Architecture document	271
A.5.8.2	Conforming General Architecture application	272
A.5.8.2.1	Application conventions	272
A.5.8.2.2	Conformance of documents	272
A.5.8.2.3	Conformance of documentation	272
A.5.8.3	Conforming General Architecture system	273
A.5.8.3.1	Conformance of documentation	273
A.5.8.3.2	Conformance to General Architecture system declaration .	273
A.5.8.3.3	Support for minimum General Architecture documents	273

A.5.8.3.4	Application conventions	273
A.5.8.4	Validating General Architecture engine	273
A.5.8.4.1	Error recognition	273
A.5.8.4.2	Identification of General Architecture messages	274
A.5.8.4.3	Content of General Architecture messages	274
A.5.8.5	Standard identification	274
A.5.9	General Architecture system declaration	275
A.6	Formal System Identifier Definition Requirements (FSIDR)	275
A.6.1	System identifiers	276
A.6.1.1	Storage object specification (SOS)	276
A.6.1.2	Informal system identifiers	276
A.6.1.3	Entity usage attributes	276
A.6.2	Auxiliary processes	277
A.6.3	Containers	278
A.6.4	FSI identification facilities	278
A.6.4.1	FSI use of APPINFO parameter	279
A.6.4.2	FSI declaration	279
A.6.4.3	FSI syntax	280
A.6.5	Storage manager attribute definitions	281
A.6.5.1	Record-related attributes	281
A.6.5.2	Encoding-related attributes	282
A.6.5.2.1	Encoding notations	283
A.6.5.2.2	BCTF algorithm notations	286
A.6.5.3	Common storage manager attributes	287
A.6.6	Entity usage attribute definitions	289
A.6.7	Storage manager notation forms	290
A.6.7.1	Local storage manager notation form	291

A.6.7.2	Starter set local storage managers	291
A.6.7.3	Portable storage manager notation form	292
A.6.7.3.1	URL Portable storage manager	293
A.6.7.3.2	Neutral file identifier storage manager	293
A.6.7.3.3	Notation processor storage manager notation form	294
A.6.7.3.4	Notation processor storage managers	295
A.6.7.4	Global storage manager notation form	295
A.6.7.5	Global storage managers	295
A.6.7.6	Container storage manager notation form	297
A.6.7.6.1	Container storage managers	298
A.6.7.6.2	Standard BENTO (sbento)	299
A.6.8	Conformance	301
A.7	SGML Property Set	302
A.7.1	SGML Notation	302
A.7.2	SGML property set	303
B	HyTime Property Set.....	349
B.1	Hyperdocuments and HyTime documents	349
B.2	HyTime Property Set	349
C	Architectural Meta-Declarations	373
C.1	HyTime Lexical Types	373
C.1.1	Calendar-Related Lexical Types	375
C.2	HyTime Meta-Declarations	377
C.3	General Architecture Meta-Declarations	462
D	Supplementary materials.....	471

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. 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.

International Standard ISO/IEC 10744 was prepared by Joint Technical Committee JTC1, *Information technology*, Subcommittee SC 18, *Document processing and related communication*.

This second edition cancels and replaces the first edition (ISO/IEC 10744:1992), which has been technically revised.

Annexes A, B, and C form an integral part of this International Standard. Annex D is for information only.

Introduction

The Hypermedia/Time-based Structuring Language (HyTime), defined in this International Standard, provides facilities for representing static and dynamic information that is processed and interchanged by hypertext and multimedia applications. HyTime is an application of ISO 8879, the Standard Generalized Markup Language (SGML).

HyTime supports the classic bibliographic model of information referencing, whereby it is possible to represent links to anything, anywhere, at any time, in a variety of ways. The extension of this model to the computerized information age, known as "integrated open hypermedia" (IOH), is the field of application of HyTime.

HyTime provides standardized mechanisms for specifying interconnections (hyperlinks) within and between documents and other information objects, and for scheduling multimedia information in time and space.

Without HyTime, such information is typically embedded in the processing instructions of hypermedia "scripts" that govern the rendition of such documents, and is therefore not usable for other forms of processing. When HyTime is used, those properties of the information that are independent of specific processing are available for processing by applications and platforms other than the one on which the information was created.

It is for the application designer and user to decide which properties can be isolated from the scripts in this way. In an ideal world, the sole consideration would be whether the properties are intrinsic to the information, regardless of how it is processed. For example, the title of this clause is intrinsic information; the font that it appears in normally is not.

In the real world, representation strategies will vary from one situation to another and will depend on such other considerations as the expected uses of the information, the flexibility of the scripting language, and performance considerations. For this reason, HyTime is highly modularized so that application designers need use only the facilities for the properties they care to describe in a standardized way.

HyTime's rules for the standardized expression of hypermedia structuring are expressed as an "enabling architecture", consisting of a number of "architectural forms" and their associated semantics. The HyTime standard's formal definition as an architecture conforms to the Architectural Form Definition Requirements in annex A of this International Standard.

0.1 HyTime modules

The architectural forms and attributes of the HyTime language are grouped into five modules, each of which have both required and optional facilities. Support for the modules and their options is indicated by “HyTime support declarations.”

— Base module

The base module consists of independent utility facilities, some of which are optional. The required facilities support hyperdocument management (using SGML) and identification of object properties. The optional facilities provide lookup tables for commonly used elements, a mechanism for associating use and access policies with objects, and a mechanism for relating attributes and the content of elements to their semantic values by reference. The base module also defines the fundamental coordinate addressing notation used by all the other HyTime modules.

— Location address module

The location address module allows the identification of objects that cannot be addressed by SGML unique identifiers, and objects that are in external documents.

Three basic types of address may be supported: name, semantic location, and coordinate location. Addressing of multiple locations is also possible. The syntax and semantics of these addressing mechanisms are independent of the data content notations of the data being addressed.

NOTE 1 The ability to resolve HyTime addresses in a given notation is dependent on software that can interpret that notation in terms of the abstractions HyTime uses for all addressing (see 6.1.1 *Object representation*).

HyTime's system- and notation-independent way of expressing addresses of hypermedia objects also provides the basis of its hyperlinking and scheduling power.

— Hyperlinks module

This module allows relationships (“hyperlinks”) to be established among objects, either within a single document or among the constituent documents and information objects of a hyperdocument.

— Scheduling module

This module allows events — occurrences of objects — to be scheduled on the coordinate axes of “finite coordinate spaces” in such a way that their positions can be expressed in terms of their relationships to one another. Measurement along the coordinate axes can be in terms of spatial or temporal units.

— Rendition module

When the scheduling module is used, object modification and/or event projection can be used to represent parameters governing the rendition process.

- Object modification

The object modification facility allows specification of the order in which objects are to be modified during rendition and of the "object modifiers" (such as amplifiers and filters) that will affect them.

NOTE 2 The semantics of the modifiers are not defined by HyTime.

- Event projection

Rendition requires the projection of events into a coordinate space where they can be perceived; for example, from a coordinate space with a virtual time axis to one based on real time. The event projection facility allows specification of the factors for computing the positions and sizes of events in the target coordinate space.

In situations where the rendered position and size of an event is not predictable (as when user interaction will affect it), the virtual dimensions of the original events may be projected onto real space/time via a formula in some arbitrary user-defined expression language. Such an expression may, among other things, accept late-binding values during rendition to resolve the positions and sizes of projected events.

NOTE 3 The semantics of formatting the objects to fit the new extents is not defined by HyTime.

Applications can choose to include rendition information as an essential part of a hyperdocument, or it can be incorporated in the "style sheets" of the processing programs. The choice depends on the nature of the information being rendered. In multimedia documents, for example, rendition style tends to be more essential to the document than is the case in conventional documents.

0.2 HyTime applications

HyTime provides a generic level of support for a variety of applications, rather than the semantics for a specific application (that is, HyTime is like a carrier or infrastructure).

The boundary between an application and HyTime is variable, and is determined by the application designer, who is free to decide how much of the information will be expressed in a standardized way using HyTime and how much will be application-specific (for example, in a data content notation).

Because the semantics of HyTime's architectural forms and attributes are standardized, it will be possible to implement supporting software and/or hardware usable for a variety of applications. Applications can define additional attributes when defining an element type that is based on an architectural form. The semantics of the application-defined element types and attributes are the only ones defined by the applications themselves. They could be standardized by an industry group or formally by a national or international standards body.

HyTime attributes have no intrinsic meaning other than that specified in this International Standard. However, an application can impute additional semantics to them, either implicitly, or by defining appropriate element types and attributes. For example, to HyTime, the "dimension reference" architectural form means only that the dimension of one object is calculated from the dimension of another. An application, however, could specify (if it wished) that use of dimension referencing implies a synchronization relationship between the objects, and could emphasize this by using "sync" as the generic identifier of a dimension reference element type.

HyTime elements can occur wherever an application's DTD and the HyTime meta-DTD allow. A finite coordinate space could occur, for example, within a paragraph of a memo in order to represent a calendar or project plan in that context, or several paragraphs could occur as the content of a timed event.

Clients of HyTime, including applications and application architectures, can define non-HyTime architectural forms as well as elements. Although an application may not add architectural forms to HyTime, nor combine HyTime architectural forms with one another, it can create its own architecture (for example, "MyArch") defining its own set of architectural forms. These architectures may be derived wholly or in part from the HyTime architecture. The facilities for defining and using architectures are defined in annex A.3.

If, for example, a document is derived from the HyTime and MyArch architectures, after the content and attributes of each element are processed and validated in SGML terms by the SGML parser, elements with HyTime attributes would be subject to processing and validation by the HyTime engine, while elements with MyArch attributes would be subject to appropriate processing and validation by the application, perhaps aided by a MyArch engine.

HyTime defines some of the parameters needed by an application to accomplish rendition, and some of the rendition functionality. The remainder is provided by the application, or by a document architecture to which the application conforms.

Many different HyTime-conforming applications and architectures could exist, to address different requirements and serve different user constituencies. Such architectures could be incompatible in their non-HyTime aspects, but would still be supportable by a single HyTime engine.

NOTE 4 For example, no application would need to invent its own system for representing finite coordinate spaces, even if its projection functions were extremely intricate and application-specific. HyTime allows application-specific projection functions, using application-chosen (or defined) function languages, to be represented in conjunction with standardized representations of the unprojected and projected finite coordinate spaces.

HyTime's design is optimized for the sequencing and alignment problems encountered in typical hypermedia applications; it is not intended as a