International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

X.680

Amendment 3
(06/2006)

SERIES X: DATA NETWORKS, OPEN SYSTEM COMMUNICATIONS AND SECURITY

OSI networking and system aspects – Abstract Syntax Notation One (ASN.1)

Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation

Amendment 3: Time type support

ITU-T Recommendation X.680 (2002) – Amendment 3

ITU-T X-SERIES RECOMMENDATIONS

DATA NETWORKS, OPEN SYSTEM COMMUNICATIONS AND SECURITY

PUBLIC DATA NETWORKS	
Services and facilities	X.1–X.19
Interfaces	X.20–X.49
Transmission, signalling and switching	X.50–X.89
Network aspects	X.90–X.149
Maintenance	X.150–X.179
Administrative arrangements	X.180–X.199
OPEN SYSTEMS INTERCONNECTION	
Model and notation	X.200–X.209
Service definitions	X.210–X.219
Connection-mode protocol specifications	X.220–X.229
Connectionless-mode protocol specifications	X.230–X.239
PICS proformas	X.240–X.259
Protocol Identification	X.260–X.269
Security Protocols	X.270–X.279
Layer Managed Objects	X.280–X.289
Conformance testing	X.290–X.299
INTERWORKING BETWEEN NETWORKS	
General	X.300–X.349
Satellite data transmission systems	X.350–X.369
IP-based networks	X.370–X.379
MESSAGE HANDLING SYSTEMS	X.400–X.499
DIRECTORY	X.500–X.599
OSI NETWORKING AND SYSTEM ASPECTS	
Networking	X.600–X.629
Efficiency	X.630–X.639
Quality of service	X.640–X.649
Naming, Addressing and Registration	X.650–X.679
Abstract Syntax Notation One (ASN.1)	**X.680–X.699**
OSI MANAGEMENT	
Systems Management framework and architecture	X.700–X.709
Management Communication Service and Protocol	X.710–X.719
Structure of Management Information	X.720–X.729
Management functions and ODMA functions	X.730–X.799
SECURITY	X.800–X.849
OSI APPLICATIONS	
Commitment, Concurrency and Recovery	X.850–X.859
Transaction processing	X.860–X.879
Remote operations	X.880–X.889
Generic applications of ASN.1	X.890–X.899
OPEN DISTRIBUTED PROCESSING	X.900–X.999
TELECOMMUNICATION SECURITY	X.1000–

For further details, please refer to the list of ITU-T Recommendations.

INTERNATIONAL STANDARD ISO/IEC 8824-1
ITU-T RECOMMENDATION X.680

Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation

Amendment 3

Time type support

Summary

This amendment provides support for the full range of time types specified in ISO 8601:2004 by the introduction of a generic type called **`TIME`** and a number of useful and defined time types that are derived from it (and are subsets of it). A subtype notation is specified that can be used to define additional time types, and that enables designers to produce variations of those types suited to their particular applications (for example, restrictions on the values used to represent midnight). The useful time types are **`DATE`** (specified using year, month, day), **`TIME-OF-DAY`** (specified using hours, minutes, seconds, to various accuracies), **`DATE-TIME`** (specified by a combination of a **`DATE`** and a **`TIME-OF-DAY`**) and **`DURATION`** (specified in any combination of time units). The defined time types include types that can be used to specify dates and times (in various forms), time intervals, and recurring time intervals. This amendment specifies the type notation, the subtype notation, the basic value notation, and the XML value notation for these types. It also contains a tutorial annex on the ISO 8601 concepts used in the definition of these types.

Source

Amendment 3 to ITU-T Recommendation X.680 (2002) was approved on 13 June 2006 by ITU-T Study Group 17 (2005-2008) under the ITU-T Recommendation A.8 procedure. An identical text is also published as ISO/IEC 8824-1, Amendment 3.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications. The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this Recommendation is voluntary. However, the Recommendation may contain certain mandatory provisions (to ensure e.g. interoperability or applicability) and compliance with the Recommendation is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the Recommendation is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementors are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database.

CONTENTS

Page

INTERNATIONAL STANDARD
ITU-T RECOMMENDATION

Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation

Amendment 3

Time type support

1) Contents

Update the Contents as follows:

2) Introduction

Insert before "Clauses 35 to 40 ...":

Clause 34 *bis* and Annex A *bis* define the types that provide support for ISO 8601.

Insert before "Annex B ..."

Annex A *bis* forms an integral part of this Recommendation | International Standard, and defines an ASN.1 module containing the definition of a set of time types providing the full functionality of ISO 8601. These types can be imported from this ASN.1 module by an application designer if the useful time types specified in clause 34 *bis* are not adequate for the application.

Insert before "Annex H ..."

Annex G *bis* does not form an integral part of this Recommendation | International Standard and provides a tutorial introduction to ISO 8601 and to the **`TIME`** type. It is recommended that this be read before the normative text.

Annex G *ter* does not form an integral part of this Recommendation | International Standard and provides information on how to identify the time properties of an abstract value from an instance of value notation.

3) Subclause 2.2

Replace ISO 8601:2000 *with* ISO 8601:2004.

4) New subclause 3.5 *bis*

Insert the following after 3.5:

3.5 *bis* Representation of dates and times

This Recommendation | International Standard uses the following terms defined in ISO 8601:

a) basic format;

b) calendar date;

c) common year;

d) duration;

e) extended format;

f) Gregorian calendar;

g) instant;

h) leap second;

i) leap year;

j) local time;

k) ordinal date;

l) recurring time interval

m) time axis;

n) time interval;

o) time point;

p) time-scale;

q) UTC;

r) week date.

5) Subclause 3.6

Add the following definitions to 3.6, in the appropriate position:

3.6.2 *bis* additional time type: A type defined as a subtype of the time type (see 3.6.71 *quin*) by applying the property setting subtype notation to the time type or to a useful or defined time type.

3.6.18 *ter* defined time type: A type defined in Annex A *bis* as a subtype of the time type (see 3.6.71 *quin*) that is intended for importation by application designers when needed for their application.

3.6.63 *bis* setting (of a time property): One of a number of values that can be associated with a given time property (see 3.6.71 *quat* and the note in G *bis*.4.2).

NOTE – Any time property that applies to a particular time abstract value has only a single setting.

3.6.71 *bis* time abstract value: An abstract value of the time type.

3.6.71 *ter* time component: Part of the definition of a time abstract value that specifies a part of that abstract value.

NOTE – Examples of time components are a date component (that would have a year component), a time-of-day component, or a time difference component.

3.6.71 *quat* time property (of a time abstract value): One of a number of terms used to describe a time abstract value (see 3.6.71 *bis*).

NOTE – The time properties that can be used to describe a time abstract value often depend on the setting of some other time property of that abstract value. The time properties are listed in Table 5 *bis*, column 1.

3.6.71 *quin* time type: The `TIME` type that supports all the abstract values implicitly defined by ISO 8601.

3.6.76 *bis* useful time type: A built-in type defined as a subtype of the time type (see 3.6.71 *quin*) that is intended for direct use by application designers.

6) Subclause 8.4

In clause 8.4, modify Table 1 as follows:

Replace UNIVERSAL 14-15 *with:*

UNIVERSAL 14 The time type

UNIVERSAL 15 Reserved for future editions of this Recommendation | International Standard

Replace UNIVERSAL 23-24 *with:*

UNIVERSAL 23-24 `UTCTime` and `GeneralizedTime`

Replace UNIVERSAL 31-... *with:*

UNIVERSAL 31-34 `DATE`, `TIME-OF-DAY`, `DATE-TIME` and `DURATION` respectively.

UNIVERSAL 35-... Reserved for future editions of this Recommendation | International Standard

7) New subclauses 11.15 *bis*, 11.15 *ter*, 11.15 *quat* and 11.15 *quin*

Insert new 11.15 bis, 11.15 ter, 11.15 quat and 11.15 quin after 11.15 as follows:

11.15 *bis* The simple character string lexical item

Name of item – simplestring

A "simplestring" shall consist of one or more ISO/IEC 10646-1 characters whose character code is in the range 32 to 126, preceded and followed by a QUOTATION MARK (34) character ("). It shall not contain a QUOTATION MARK (34) character ("). The "simplestring" may span more than one line of text, in which case any characters representing end-of-line shall be treated as spacing characters. In analysing an instance of use of this notation, a "simplestring" is distinguished from a "cstring" by the context in which it appears.

NOTE – The "simplestring" lexical item is only used in the subtype notation of the time type.

11.15 *ter* Time value character strings

Name of item – tstring

A "tstring" shall consist of one or more of the characters:

```
0 1 2 3 4 5 6 7 8 9 + - : . , / C D H M R P S T W Y Z
```

preceded and followed by a QUOTATION MARK (34) character (").

NOTE – The "tstring" lexical item is only used in the value notation for the time type.

11.15 *quat* XML time value character string item

Name of item – xmltstring

An "xmltstring" shall consist of one or more of the characters:

```
0 1 2 3 4 5 6 7 8 9 + - : . , / C D H M R P S T W Y Z
```

NOTE – The "xmltstring" lexical item is only used in the XML value notation of the time type.

11.15 *quin* The property and setting names lexical item

Name of item – psname

A "psname" shall consist of an arbitrary number (one or more) of letters, digits and hyphens. The initial character shall be an upper-case letter. A hyphen shall not be the last character. A hyphen shall not be immediately followed by another hyphen.

NOTE – The "psname" lexical item is only used in the contents of the "simplestring" used in the subtype notation for the time type.

8) Subclause 11.27

Add the following reserved words to 11.27, in their alphabetical position:

```
DATE      DATE-TIME      DURATION      SETTINGS      TIME      TIME-OF-DAY
```

9) Subclause 16.2

*In 16.2, add the following lines to the production "***BuiltinType***":*

After "**ChoiceType**":

```
| DateType
| DateTimeType
| DurationType
```

After "**PrefixedType**":

```
| TimeType
| TimeOfDayType
```

10) Subclause 16.2

In 16.2, add the following lines to the references to defining clauses:

After "ChoiceType":

DateType	34 *bis*.4.1
DateTimeType	34 *bis*.4.3
DurationType	34 *bis*.4.4

After "PrefixedType":

TimeType	34 *bis*.1.1
TimeOfDayType	34 *bis*.4.2

11) Subclause 16.9

*In 16.9, add the following line to the production "***BuiltinValue***":*

After "**PrefixedValue**":

| **TimeValue**

12) Subclause 16.10

*In 16.10, add the following line to the production "***XMLBuiltinValue***":*

After "**XMLPrefixedValue**":

| **XMLTimeValue**

13) New clause 34 *bis*

Insert a new 34 bis after clause 34 as follows:

34 *bis* The time type

34 *bis*.1 General

34 *bis*.1.1 The time type (see 3.6.71 *quin*) shall be referenced by the notation "TimeType":

TimeType ::= `TIME`

34 *bis*.1.2 The tag for types defined by this notation is universal class, number 14.

34 *bis*.1.3 The value of a time type shall be defined by the notation "TimeValue", or when used as an "XMLValue", by the notation "XMLTimeValue". The syntax of these notations is defined in 34 *bis*.3 as the contents of a "simplestring", using notation defined in ISO 8601, 3.4.

34 *bis*.2 Time properties and settings of time abstract values

34 *bis*.2.1 Table 5 *bis* specifies in column 1 the description and names of the time properties of time abstract values. In column 2, it specifies the names of the possible time property settings for the column 1 time property. Column 3 specifies (generally by reference to ISO 8601) the abstract values to which the time property is applicable, and that have the corresponding time property settings.

NOTE 1 – ASN.1 does not specify abstract values that are not supported by ISO 8601 representations.

NOTE 2 – The names of time properties and of their settings appear in the property assertions of the property settings subtype notation (see clause 47).

Table 5 *bis* – Properties and settings for time abstract values

Time property	Names of property settings	Abstract values that have this property setting
Basic nature of the abstract value **Name**: `Basic` **Comment**: The setting of this property identifies the basic nature of the abstract value. All time abstract values have this property.	`Date`	See ISO 8601, 4.1. All abstract values that are dates only.
	`Time`	See ISO 8601, 4.2. All abstract values that are a time-of-day only.
	`Date-Time`	See ISO 8601, 4.3. All abstract values that are a date and a time-of-day.
	`Interval`	See ISO 8601, 4.4. All the time interval abstract values.
	`Rec-Interval`	See ISO 8601, 4.5. All the recurring interval abstract values.
Time-scale and accuracy for a date **Name**: `Date` **Comment**: This applies only to an abstract value that includes identification of a date. It identifies the time-scale and accuracy of that date. NOTE – Any abstract value identifying more than one date (for example, an interval) has a single setting for `Date` that applies to both dates.	`C` (Century)	See ISO 8601, 4.1.2.3 c). All abstract values containing a date that represents only a century.
	`Y` (Year only)	See ISO 8601, 4.1.2.3 b). All abstract values containing a date that represents only a year.
	`YM` (Year-Month)	See ISO 8601, 4.1.2.3 a). All abstract values containing a date that uses the year-month time-scale.
	`YMD` (Year-Month-Day)	See ISO 8601, 4.1.2.2. All abstract values containing a date that uses the year-month-day time-scale.
	`YD` (Year-Day)	See ISO 8601, 4.1.3.2. All abstract values containing a date that uses the year-day time-scale.
	`YW` (Year-Week)	See ISO 8601, 4.1.4.3. All abstract values containing a date that uses the year-week time-scale.
	`YWD` (Year-Week-Day)	See ISO 8601, 4.1.4.2. All abstract values containing a date that uses the year-week-day time-scale.

Table 5 *bis* – Properties and settings for time abstract values

Time property	Names of property settings	Abstract values that have this property setting
Type of associated year **Name**: `Year` **Comment**: This applies only to an abstract value that includes identification of one or more years or centuries. Its setting identifies whether the year (or century) identification is a "normal" year, a year in the proleptic Gregorian Calendar (see G *bis*.2.2), a year that is negative, or a year that requires more than four digits to represent it. NOTE – Any abstract value involving more than one year (for example, an interval) has a single setting for `Year` that applies to both years.	`Basic`	All abstract values containing a year in the range 1582 to 9999 (or a century in the range 15 to 99).
	`Proleptic`	All abstract values containing a year in the range 0 to 1581 (or a century in the range 00 to 14). NOTE – In the proleptic Gregorian calendar, a year value of zero has a meaning which roughly corresponds to the year 1 BC (see G *bis*.2.2).
	`Negative`	All abstract values containing a year in the range –9999 to –0001 (or a century in the range –99 to –01).
	`L5`, `L6`, `L7`, etc., to infinity (Large)	All abstract values containing a year whose decimal representation requires 5, 6, 7, etc., digits (or a century whose decimal representation requires 3, 4, 5, etc., digits) respectively, whether positive or negative.
Accuracy for a time **Name**: `Time` **Comment**: This applies only to an abstract value that includes identification of a time-of-day. It identifies the accuracy of that time-of-day. NOTE – Any abstract value identifying more than one time-of-day (for example, an interval) has a single setting for `Time` that applies to both the time-of-days.	`H` (Hour)	See ISO 8601, 4.2.2.3 b). All abstract values containing a time-of-day to an accuracy of hours.
	`HM` (Hour-Minute)	See ISO 8601, 4.2.2.3 a). All abstract values containing a time-of-day to an accuracy of minutes.
	`HMS` (Hour-Minute-Second)	See ISO 8601, 4.2.2.2. All abstract values containing a time-of-day to an accuracy of seconds.
	`HF1`, `HF2`, `HF3`, etc., to infinity (Hour-decimal-fraction)	See ISO 8601, 4.2.2.4 c). All abstract values containing a time-of-day to an accuracy of hours to 1, 2, 3, etc., decimal places.
	`HMF1`, `HMF2`, `HMF3`, etc., to infinity (Hour-Minute-fraction)	See ISO 8601, 4.2.2.4 b). All abstract values containing a time-of-day to an accuracy of minutes to 1, 2, 3, etc., decimal places.
	`HMSF1`, `HMSF2`, `HMSF3`, etc., to infinity Hour-Minute-Second-Fraction	See ISO 8601, 4.2.2.4 a). All abstract values containing a time-of-day to an accuracy of seconds to 1, 2, 3, etc., decimal places.

Table 5 *bis* – Properties and settings for time abstract values

Time property	Names of property settings	Abstract values that have this property setting
Local or UTC time-scale for a time **Name**: `Local-or-UTC` **Comment**: This applies only to an abstract value that includes identification of a time. It identifies the time-scale of that time (local time, UTC, or local time plus the difference from UTC). Time differences are determined by local administrations. ASN.1 supports time differences in the range –15 hours to +16 hours. The difference is positive if the local time is ahead of or equal to UTC (see ISO 8601, 5.2.4.1). See also G *bis*.2.11. NOTE – Any abstract value identifying more than one time (for example, an interval) has a single setting for `Local-or-UTC` that applies to both times.	`L` (Local time only)	See 34 *bis*.2.2 and ISO 8601, 4.2.2 and 4.2.3. All abstract values containing a time-of-day that specifies local time only.
	`Z` (UTC only)	See ISO 8601, 4.2.4. All abstract values containing a time-of-day that specifies UTC and not local time.
	`LD` (Local time and the difference from UTC)	See ISO 8601, 4.2.5. All abstract values containing a time-of-day that specifies local time and the time (which may be negative) added to UTC to obtain local time.
Form of interval specification **Name**: `Interval-type` **Comment**: This applies only to an abstract value that is an interval or a recurring interval. It identifies the form of interval specification (a start and an end point, a duration, a start point and a duration, or a duration with an end point).	`SE` (Start and end points)	See ISO 8601, 4.4.1 a). All abstract values that specify an interval using a start and an end point.
	`D` (Duration only)	See ISO 8601, 4.4.1 b) and 4.4.3. All abstract values that specify an interval using only a duration.
	`SD` (Start point and duration)	See ISO 8601, 4.4.1 c). All abstract values that specify an interval using a start point and a duration.
	`DE` (Duration and end point)	See ISO 8601, 4.4.1 d). All abstract values that specify an interval using a duration and an end point.
Nature of the start and/or end point specification **Name**: `SE-point` **Comment**: This applies only to intervals or recurring intervals using a start point or an end point or both. The setting of this property identifies the nature of the start point and/or end point that forms part of this abstract value. NOTE – All interval abstract values with both a start point and an end point have a single setting for this property, and for any associated properties related to date or time-of-day. There are no interval abstract values that have different forms of start point and end point. Thus all abstract values with both an interval start point and an interval end point have the same set of time components for the start point and the end point (but see Table 5 *ter* for value notation for the end-point). This is a difference from ISO 8601.	`Date`	See ISO 8601, 4.1. All abstract values that specify start and/or end points using dates only.
	`Time`	See ISO 8601, 4.2. All abstract values that specify start and/or end points using time-of-day only.
	`Date-Time`	See ISO 8601, 4.3. All abstract values that specify start and/or end points using a date and a time-of-day.
Recurrence specification **Name**: `Recurrence` **Comment**: This applies only to an abstract value that is a recurring interval. It identifies the agreed limits on the number of recurrences (or unlimited).	`Unlimited` (No limit on the number of recurrences, expressed with an empty string for the number of recurrences)	See ISO 8601, 4.5. All abstract values representing an unlimited number of recurrences of an interval.
	`R1`, `R2`, `R3`, etc., to infinity (Number of recurrence digits)	See ISO 8601, 4.5. All abstract values representing recurrences of an interval that can be expressed in 1, 2, 3, etc., digits respectively.

Table 5 *bis* – Properties and settings for time abstract values

Time property	Names of property settings	Abstract values that have this property setting
Midnight start or end of a day **Name**: `Midnight` **Comment**: This applies only to an abstract value that contains a time that represents midnight. It identifies whether this midnight value is the start of a day (often represented as 00:00:00) or the end of a day (often represented as 24:00:00).	`Start` (Start-of-day)	See ISO 8601, 4.2.3 a). An abstract value containing a time that represents midnight at the start of a day.
	`End` (End-of-day)	See ISO 8601, 4.2.3 b). An abstract value containing a time that represents midnight at the end of a day.
NOTE – ASN.1 does not support the use of start and end points of intervals that have different time properties, as there is only a single `SE-point` setting that governs the syntax of both the start point and the end-point. The start and end points are required to use the same time format. This is a difference from ISO 8601.		

34 *bis*.2.2 ISO 8601 provides two basic representations for midnight: "2400" for midnight at the end of a day and "0000" for midnight at the start of a day (with any second or fractional part of a second containing only zero digits). These are not considered different representations for a single abstract value, but as distinct abstract values.

NOTE 1 – This is because as a stand-alone time, they are clearly distinct and represent start of a day and end of a day. When used in conjunction with a day, "2400" on day x should be considered less than "0000" on day x+1, despite having exactly the same position on the time axis.

NOTE 2 – They have, respectively, the time property setting `"Midnight=End"` and `"Midnight=Start"`.

NOTE 3 – As with other time points, there are infinitely many distinct abstract values that are midnight at the start and end of any particular day, depending on the accuracy of the seconds and fractional part of seconds. There are also further infinite sets of midnight abstract values based on the use of fractions of an hour or of a minute rather than of seconds. (All these fractional parts will be zero to various different accuracies if the abstract value is a midnight value.)

34 *bis*.2.3 ISO 8601 provides two basic representations for duration (either weeks, or some combination of years, months, days, hours, minutes and seconds) as a component of time intervals and recurring time intervals. Different strings representing durations in ISO 8601 are considered to represent different abstract values in ASN.1, except where the only difference is the omission or inclusion of a zero time component that does not change the duration (including the accuracy of the duration) being represented. Inclusion or omission of zero time components is fully specified in canonical encoding rules, and in all the encoding rules of ITU-T Rec. X.691 | ISO/IEC 8825-2. There are no time properties (other than `"Basic=Interval Interval-type=D"`) associated with a duration, but restrictions can be applied to the time components of a duration, requiring them to be absent or limiting their value (see 34 *bis*.4.4).

NOTE 1 – There is an ISO 8601 requirement for prior agreement on the size of components (and particularly of fractional parts). This is normally handled by property settings for the different accuracies. However, in the case of `DURATION`, for simplicity, property settings were not introduced to determine the accuracy of the components. Instead, inner subtyping constraints on the equivalent sequence type can be applied, as specified in 34 *bis*.4.4, to record prior agreements on the components of a `DURATION`.

NOTE 2 – ISO 8601 requires that use of a weeks component shall not be combined with the use of any other date component (years, months, days), nor with the use of an hours, minutes, or seconds time component. This restriction is also applied in ASN.1 for consistency with ISO 8601.

34 *bis*.2.4 There is no defined order relation between the different `DURATION` abstract values unless they are expressed using a single time element (for example, weeks or months or days only), as there is no agreed international definition of a duration of one month or one year in terms of seconds.

34 *bis*.3 Basic value notation and XML value notation for time abstract values with specified property settings

34 *bis*.3.1 All time abstract values with the same time property settings have the same value notation, varied only by the values of year, month, week, day, hour, minute, second, etc. (on the associated time-scale) that are used to distinguish that abstract value from others with the same property settings.

34 *bis*.3.2 The value notations for the time type shall be "TimeValue" and "XMLTimeValue":

TimeValue ::= tstring

XMLTimeValue ::= xmltstring

The content of the "tstring" and of the "xmltstring" is defined in 34 *bis*.3.4 using the time component syntax that is defined in column 3 of Table 5 *ter*. Table 5 *ter* defines a number of possible notations for the different components (for example, the year component). The precise notation to be used depends on the property settings of the abstract value specified in column 2. Properties not listed in column 2 have no effect on the notation to be used for the component.

These time component notations are normally defined by reference to an ISO 8601 representation (with which they are conformant), but in order to avoid ambiguity in value notation, an additional C character is added to time components that designate a century and not a full year, as specified in column 3 of Table 5 *ter*.

34 *bis*.3.3 Table 5 *ter* specifies (in column 3) the value notation and XML value notation for time components (listed in column 1). Column 1 identifies a time component. Column 2 specifies the conditions in which a particular row is applicable, in terms of the settings of properties associated with abstract values. Column 3 specifies the notation to be used for that time component. The notation used in column 3 is that defined in ISO 8601, 3.4, with the addition of C as a century designator.

NOTE 1 – The ISO 8601 notation used in column 3 can be summarized as: Y is a year digit, M is a month digit or month designator, D is a day digit, w is a week digit, h is an hour digit, m is a minute digit, s is a second digit, n is any of 0 to 9, ± is plus or minus, and underline represents zero or more repetitions (for example "±YYYYY"). The ISO 8601 notation is used in preference to any other notation used in this Recommendation | International Standard in order to make the linkage to ISO 8601 clear.

NOTE 2 – Annex G *bis*.2 provides a tutorial on ISO 8601 key concepts that will help in understanding this notation. See also Annex E.2 *bis* for examples of the resulting value notation.

Table 5 *ter* – Value notation for time abstract values with specific properties and settings

Time component	Property	Value notation syntax
Year component	"Year=Basic" and "Date=C" or "Year=Proleptic" and "Date=C"	ISO 8601, 4.1.2.3 c) followed by the character LATIN CAPITAL LETTER C: [YYC]
Year component	"Year=Negative" and "Date=C" or "Year=L*n*" and "Date=C"	ISO 8601, 4.1.2.4 d) followed by the character LATIN CAPITAL LETTER C: [±YYYC] The number of repetitions of Y shall be zero for "Year=Negative" and equal to *n*-2 for "Year=L*n*".
Year component	"Year=Basic" and Date is not C or "Year=Proleptic" and Date is not C	ISO 8601, 4.1.2.2: [YYYY]
Year component	"Year=Negative" and Date is not C or "Year=L*n*" and Date is not C	ISO 8601, 4.1.2.4 c): [±YYYYY] The number of repetitions of Y shall be zero for "Y=Negative" and equal to *n*-4 for "Y=L*n*".
Month component	Any	ISO 8601, 4.1.2.3 a): [-MM]
Week component	Any	ISO 8601, 4.1.4.3: [-Www]
Day component	"Year=YMD"	ISO 8601, 4.1.2.2 Extended format: [-DD]
Day component	"Year=YD"	ISO 8601, 4.1.3.2 Extended format: [-DDD]
Day component	"Year=YWD"	ISO 8601, 4.1.4.2 Extended format: [-D]
Hours component	"Basic=Time" or "Basic=Interval" and "SE-point=Time" or "Basic=Rec-Interval" and "SE-point=Time"	ISO 8601, 4.2.2.3 b): [hh] The hours component value notation 24 shall always be used for the abstract value "midnight at end of day" and the hours component value notation 00 for "midnight at start of day".
Hours component	"Basic=DateTime" or "Basic=Interval" and "SE-point=DateTime" or "Basic=Rec-Interval" and "SE-point=DateTime"	ISO 8601, 4.3.2 Extended format: [Thh] The value notation T24 shall always be used for the hours component of the abstract value "midnight at end of day" and the value notation T00 for "midnight at start of day".

Table 5 *ter* – Value notation for time abstract values with specific properties and settings

Time component	Property	Value notation syntax
Minutes component	Any	ISO 8601, 4.3.2 Extended format: [:mm]
Seconds component	Any	ISO 8601, 4.3.2 Extended format: [:ss]
Decimal fraction component of hour, minute, or second	Any	ISO 8601, 4.2.2.4: [,hh] or [.hh], [,mm] or [.mm], or [,ss] or [.ss] NOTE – It is recommended that in any given ASN.1 module, the comma or full stop be consistently used for the decimal sign.

Decimal fraction component of year, month, week, or day in a duration (see G *bis*.2.6, Note)	`"Basic=Interval"` and `"Interval-type=D"`
	or
	`"Basic=Interval"` and `"Interval-type=SD"`
	or
	`"Basic=Interval"` and `"Interval-type=DE"`
		ISO 8601, 4.4.3.2: [,nn] or [.nn]
		NOTE – It is recommended that in any given ASN.1 module, the comma or full stop be consistently used for the decimal sign.

UTC designator component	`"Local-or-UTC=Z"`	ISO 8601, 4.2.4: [Z]
Time difference component	`"Local-or-UTC=LD"`	ISO 8601, 4.2.5.2 Extended format: [±hh] or [±hh:mm] The time difference component shall be the exact time difference in minutes if it is not an exact multiple of hours. NOTE – This means that the minutes component has to be present unless the difference between local time and UTC is an integral number of hours.

Duration component	`"Interval-type=D"`
	or
	`"Interval-type=SD"`
	or
	`"Interval-type=DE"`
		ISO 8601, 4.4.3.2:
		see 34 *bis*.3.6

Time interval	`"Interval-type=SE"`
	or
	`"Interval-type=SD"`
	or
	`"Interval-type=DE"`
		ISO 8601, 4.4 Extended formats:
		Start point component (`"Interval-type=SE"` or `"Interval-type=SD"`) or duration component (`"Interval-type=DE"`), followed by [/], followed by duration component (`"Interval-type=SD"`) or end point component (`"Interval-type=SE"` or `"Interval-type=DE"`).

Start point component	Depends on `SE-point` setting	This is determined by the setting of `SE-point`, which shall be interpreted as a setting of the `Basic` property for representing this component. The `Date`, `Year`, `Time`, and `Local-or-UTC` property settings shall then be used to determine the format of the start point component.
End point component	Depends on `SE-point` setting	This is determined by the setting of `SE-point`, which shall be interpreted as a setting of the `Basic` property for representing this component. The `Date`, `Year`, `Time`, and `Local-or-UTC` property settings shall then be used to determine the format of the end point component. It is permissible (optionally) to omit the time difference component if the difference between UTC and local time for the end point is the same as the difference for the start point. NOTE – This is not as general as ISO 8601, but is restricted to these cases for simplicity.
Recurring time intervals	`"Recurrence=Unlimited"`	ISO 8601, 4.5 Extended format: [R/] followed by the time interval component.
Recurring time intervals	`"Recurrence=R1"`, `"Recurrence=R2"`, `"Recurrence=R3"`, etc.	ISO 8601, 4.5 Extended format: [Rnn/] followed by the time interval component.

34 *bis*.3.4 The value of the "tstring" shall be the concatenation of the character encodings of the time components (determined by the settings of their properties in accordance with Table 5 *bis*), preceded and followed by a QUOTATION MARK (34) character (") as specified in 11.15 *ter*. The value of the "xmltstring" shall be the concatenation of the character encodings of the time components (determined by the settings of their properties in accordance with Table 5 *bis*), without surrounding QUOTATION MARK characters.

NOTE 1 – The value notation and XML value notation are canonical except for:

a) the varying representations of duration; and

b) the varying use of comma or full stop for the decimal separator; and

c) the varying use of hours and minutes or hours only for time difference components that are an integral number of hours; and

d) the inclusion or omission of a time difference component in the end point of an interval (with both a start point and an end point) when the time difference in the end point is the same as the time difference in the start point.

NOTE 2 – Examples of the value notation are provided in E.2 *bis*.

34 *bis*.3.5 The notations for the time components shall be concatenated in the order specified in ISO 8601.

NOTE – This means the most significant time component first and the zone designator (time difference component or Z) last.

34 *bis*.3.6 The basic value notation and the XML value notation for the duration component are specified in the following subclauses.

34 *bis*.3.6.1 The value notation shall be [P] (see ISO 8601, 4.4.3) followed by either:

a) a year-month-day designation (see 34 *bis*.3.6.2) optionally followed by an hours-mins-sec designation (see 34 *bis*.3.6.3); or

b) a week designation (see 34 *bis*.3.6.4); or

c) an hours-mins-sec designation (see 34 *bis*.3.6.3).

34 *bis*.3.6.2 A year-month-day designation shall be one or more (in order) of:

a) a year designation (see 34 *bis*.3.6.5);

b) a month designation (see 34 *bis*.3.6.6);

c) a day designation (see 34 *bis*.3.6.7).

34 *bis*.3.6.3 An hours-mins-secs designation shall be [T] followed by one or more (in order) of:

a) an hours designation (see 34 *bis*.3.6.8); or

b) a minutes designation (see 34 *bis*.3.6.9); or

c) a seconds designation (see 34 *bis*.3.6.10).

34 *bis*.3.6.4 A week designation shall consist of one or more digits optionally followed by a fractional part (see 34 *bis*.3.6.12) followed by [W].

34 *bis*.3.6.5 A year designation shall consist of one or more digits optionally followed by a fractional part (see 34 *bis*.3.6.12) followed by [Y].

34 *bis*.3.6.6 A month designation shall consist of one or more digits optionally followed by a fractional part (see 34 *bis*.3.6.12) followed by [M].

34 *bis*.3.6.7 A day designation shall consist of one or more digits optionally followed by a fractional part (see 34 *bis*.3.6.12) followed by [D].

34 *bis*.3.6.8 An hours designation shall consist of one or more digits optionally followed by a fractional part (see 34 *bis*.3.6.12) followed by [H].

34 *bis*.3.6.9 A minutes designation shall consist of one or more digits optionally followed by a fractional part (see 34 *bis*.3.6.12) followed by [M].

34 *bis*.3.6.10 A seconds designation shall consist of one or more digits optionally followed by a fractional part (see 34 *bis*.3.6.12) followed by [S].

34 *bis*.3.6.11 The integral part of a designation shall not contain leading zeros unless it is the single digit zero, optionally followed by a fractional part. There shall be at least one digit in the integral part if there is a following fractional part.

34 *bis*.3.6.12 A fractional part shall consist of a decimal separator (which shall be either a full stop or a comma), followed by one or more decimal digits.

34 *bis*.3.6.13 If a designation contains a fractional part, there shall be no following designation.

34 *bis*.3.6.14 Value notations expressing a duration to different accuracies represent different abstract values.

EXAMPLE 1: The following value notations all represent different abstract values:

a) `P29M` (or `P0Y29M`) -- 0 years, 29 months to an accuracy of 1 month.

b) `P29M0D` (or `P0Y29M0D`) -- 0 years, 29 months, 0 days to an accuracy of 1 day.

c) `P29MT0S` (or `P0Y29M0DT0H0M0S`) -- 0 years, 29 months, 0 days, 0 hours, 0 minutes, 0 seconds, to an accuracy of 1 second.

d) `P29MT0.00H` (or `P0Y29M0DT0,00H`) -- 0 years, 29 months, 0 days, 0 hours, to an accuracy of one-hundredth of an hour.

e) `P29MT0.000S` (or `P0Y29M0DT0H0M0.000S`) -- 0 years, 29 months, 0 days, 0 hours, 0 minutes, 0 seconds, to an accuracy of 1 millisecond.

EXAMPLE 2: The following value notations all represent the same abstract value (0 years, 29 months, 0 days, 0 hours, 0 minutes) to an accuracy of one-hundredth of a minute:

a) `P0Y29M0DT0H0.00M`

b) `P0Y29M0DT0.00M`

c) `P0Y29MT0H0.00M`

d) `P0Y29MT0.00M`

e) `P29M0DT0H0.00M`

f) `P29M0DT0.00M`

g) `P29MT0H0.00M`

h) `P29MT0.00M`

34 *bis*.4 Useful time types

The following useful time types are defined, and are expected to cover most normal requirements of application designers.

NOTE – These definitions use the property setting subtype notation specified in clause 47. Where alternative time-scales are required, for example, use of a Year and Day calendar, defined time types (see Annex A *bis*) can be used, or the property setting subtype notation can be used to define additional subtypes of the `TIME` type (see E.2 *bis* for examples of properties and settings that can be used).

34 *bis*.4.1 The date type shall be referenced by the notation:

DateType ::= `DATE`

and is defined as:

```
DATE ::= [UNIVERSAL 31] IMPLICIT TIME
      (SETTINGS "Basic=Date Date=YMD Year=Basic")
```

34 *bis*.4.2 The time-of-day type shall be referenced by the notation:

TimeOfDayType ::= `TIME-OF-DAY`

and is defined as:

```
TIME-OF-DAY ::= [UNIVERSAL 32] IMPLICIT TIME
      (SETTINGS "Basic=Time Time=HMS Local-or-UTC=L")
```

NOTE – This type allows midnight at start of day (`00:00:00`) as well as midnight at end of day (`24:00:00`).

34 *bis*.4.3 The date-time type shall be referenced by the notation:

DateTimeType ::= `DATE-TIME`

and is defined as:

```
DATE-TIME ::= [UNIVERSAL 33] IMPLICIT TIME
      (SETTINGS "Basic=Date-Time Date=YMD Year=Basic Time=HMS
       Local-or-UTC=L")
```

NOTE – This type allows midnight at start of day (`00:00:00`) as well as midnight at end of day (`24:00:00`).

34 *bis*.4.4 The duration type shall be referenced by the notation:

DurationType ::= `DURATION`

and is defined as:

```
DURATION ::= [UNIVERSAL 34] IMPLICIT TIME
      (SETTINGS "Basic=Interval Interval-type=D")
```

Any subset of the `TIME` type, all of whose abstract values have the property settings `"Basic=Interval Interval-type=D"` (whether `UNIVERSAL 34` or `UNIVERSAL 14`), is called a duration subtype. This type can be constrained in accordance with the following subclauses.

34 *bis*.4.4.1 Inner subtyping constraints can be applied to any duration subtype using an equivalent sequence type (see 34 *bis*.4.4.2).

NOTE – The inner subtyping constraint applied to the equivalent sequence type can be used to forbid or to require particular time components in the duration type, or to place range constraints on the values of some or all time components of the duration type (see also 47.11.2).

34 *bis*.4.4.2 The `DURATION-EQUIVALENT` equivalent sequence type is:

```
DURATION-EQUIVALENT ::= SEQUENCE {
      years       INTEGER (0..MAX) OPTIONAL,
      months      INTEGER (0..MAX) OPTIONAL,
      weeks       INTEGER (0..MAX) OPTIONAL,
      days        INTEGER (0..MAX) OPTIONAL,
      hours       INTEGER (0..MAX) OPTIONAL,
      minutes     INTEGER (0..MAX) OPTIONAL,
      seconds     INTEGER (0..MAX) OPTIONAL,
      fractional-part  SEQUENCE {
                  number-of-digits  INTEGER(1..MAX),
                  fractional-value  INTEGER(0..MAX) } OPTIONAL }
```

where the years component of the equivalent sequence type corresponds to the years time component of the abstract value of the duration type, and so on.

34 *bis*.4.4.3 Constraints placed on the components of the equivalent sequence type are constraints on the corresponding time components of the duration type.

NOTE 1 – The rules for duration types require that at least one of the time components be present (see 34 *bis*.2.3), but that no other time components be present when the week is present. Use of an inner subtyping constraint that violated these rules would be an illegal specification.

NOTE 2 – The fractional-part always applies to the least significant time component that is present in the abstract value.

34 *bis*.4.5 The basic value notation and the XML value notation for all the useful time types shall be the value notation for the `TIME` type (see 34 *bis*.3.2), restricted to notation for those abstract values that are present in the useful time type.

14) Clause 42

Replace clause 42 (retaining the heading) with:

NOTE – Earlier versions of this Recommendation | International Standard used different text (due to the evolution of the ISO time standards), but the technical content is unchanged from the first version of this Recommendation | International Standard.

42.1 This type shall be referenced by the name:

`GeneralizedTime`

42.2 The type consists of ~~values representing:~~

~~a)~~ a calendar date, ~~as defined in ISO 8601; and~~together with:

~~b) a time of day, to any of the precisions defined in ISO 8601, except for the hours value 24 which shall not be used; and~~

~~c) the local time differential factor as defined in ISO 8601.~~

a) a local time of day, including midnight at the start of a day, but excluding midnight at the end of a day, to an accuracy of:

1) hours, minutes, and seconds (or seconds and fractions of a second to any number of decimal places); or

2) hours and minutes (or minutes and fractions of a minute to any number of decimal places); or

3) hours (or hours and fractions of an hour to any number of decimal places); or

b) a UTC time of day, including midnight at the start of a day, but excluding midnight at the end of a day, to any of the accuracies listed in a) above; or

c) a local time of day as specified in a) above, together with the difference between local time and UTC.

NOTE – The time difference component is positive if the local time is ahead of UTC.

42.3 The type is defined, using ASN.1, as follows:

```
GeneralizedTime~~ ::=    ~~ ::= [UNIVERSAL 24] IMPLICIT VisibleString
```

with the values of the `VisibleString` restricted to strings of characters which are either:

a) a specification of a calendar date followed by a local time, consisting of:

1) a string representing the calendar date (as specified in ISO 8601,~~ with~~ 4.1.2.2 – Basic format); followed by:

NOTE 1 – This specifies a four-digit representation of the year, a two-digit representation of the month and a two-digit representation of the day, without use of separators.~~, followed by a string representing the time of day, as specified in ISO 8601, without separators other than decimal comma or decimal period (as provided for in ISO 8601), and with no terminating Z (as provided for in ISO 8601); or~~

~~b) the characters in a) above followed by an upper-case letter `Z`; or~~

~~c) the characters in a) above followed by a string representing a local time differential, as specified in ISO 8601, without separators.~~

~~In case a), the time shall represent the local time. In case b), the time shall represent coordinated universal time. In case c), the part of the string formed as in case a) represents the local time (t_1), and the time differential (t_2) enables coordinated universal time to be determined as follows:~~

~~coordinated universal time~~ 2) a string representing the time of day to an accuracy of one hour, one minute, one second, or fractions of a second (to any degree of accuracy), using either comma or full stop as the decimal sign (as specified in ISO 8601, 4.2.2.2 and 4.2.2.3 – Basic format); optionally followed by:

3) a decimal fraction of a minute if seconds are omitted, or a decimal fraction of an hour if minutes and seconds are omitted (as specified in ISO 8601, 4.2.2.4); or

NOTE 2 – ISO 8601 specifies the use of either a comma or a full stop as the decimal sign. There are no other separators present. It is recommended that in any given ASN.1 specification, either comma or full stop be consistently used as the decimal sign.

b) a specification of a calendar date and a UTC time consisting of the characters in a) above followed by an upper-case letter Z; or

c) a specification of a calendar date, the local time, and the exact difference between local time and UTC as specified in ISO 8601, with the minutes component optionally omitted if the difference is an integral number of hours.

NOTE 3 – Early work on ASN.1 canonical encoding rules assumed that there was no actual concept of accuracy, so that an abstract value that might be represented with a seconds component of 3.000 was regarded as the same abstract value as one that was represented with a seconds component of 3, and forbade the use of trailing zeros in canonical encoding fractional parts, and forbade the omission of seconds or minutes and seconds. It also supported only the use of UTC time, not local time or local time with a time difference component. This has not been changed in later editions of the ASN.1 standards, for backwards compatibility. The `TIME` type (introduced into ASN.1 in 2004) recognizes that abstract values can have an associated accuracy, and that (e.g.) the representations of seconds as 3.000 and 3 produces different abstract values, and that local time and UTC specifications represent different abstract values. The canonical encoding rules for `TIME` encode the full range of its abstract values, so use of `TIME` may be preferred in new specifications to the use of `GeneralizedTime`.

In case c), the part of the string formed as in case a) represents the local time (t_1), and the (signed) time difference (t_2) enables UTC to be determined. If t_2 is positive, local time is ahead of UTC. We can thus determine UTC as:

UTC is $t_1 - t_2$

EXAMPLES

Case a)

```
"19851106210627.3"
```

~~local~~Local time 6 minutes, 27.3 seconds after 9 pm on 6 November 1985.

Case b)

`"19851106210627.3Z"`
~~coordinated~~Coordinated universal time as above.

Case c)

`"19851106210627.3-0500"`
~~local~~Local time as in example a), with local time 5 hours ~~retarded in relation to~~ coordinated universal time.

Case d)

`"198511062106.456"`
Local time 6.456 minutes after 9 pm on 6 November 1985.

Case e)

`"1985110621.14159"`
Local time 0.14159 hours after 9 pm on 6 November 1985.

42.4 The tag shall be as defined in 42.3.

42.5 The value notation shall be the value notation for the `VisibleString` defined in 42.3.

15) Subclause 47.1

In 47.1, replace the first paragraph with:

A number of different forms of notation for "SubtypeElements" are provided. They are identified below, and their syntax and semantics are defined in the following subclauses. Table 9 and Table 9 *bis* summarize~~s~~ which notations can be applied to which parent types. "SubtypeElements" not present in one of the tables means that the corresponding subtype element cannot be applied to any of the parent types listed in that table.

16) Subclause 47.1

*In 47.1, add the following lines to the production "***SubtypeElements***":*

After "**PatternConstraint**":

| **PropertySettings**
| **DurationRange**
| **TimePointRange**
| **RecurrenceRange**

17) Table 9

Replace Table 9 heading with:

Table 9 – Applicability of ~~subtype value sets~~ "SubtypeElements" to types other than the Time type

18) Table 9

*Replace Table 9 row heading "***Time types***" with:*

GeneralizedTime and UTCTime types

19) New Table 9 *bis*

Add a new Table 9 bis after Table 9 as follows:

Table 9 bis – Applicability of "SubtypeElements" to the Time type

Type (or derived from such a type by tagging or subtyping)	Single value	Contained subtype	Property settings	Duration range	Time point range	Recurrence range	Inner subtyping
Time type	Yes	Yes	Yes	Yes	Yes	Yes	(Note)
NOTE – Only allowed if all the abstract values of the parent type have the property settings `"Basic=Interval Interval-type=D"` (see 34 *bis*.4.4).							

20) New clauses 47.10, 47.11, 47.12, and 47.13

Insert new clauses 47.10, 47.11, 47.12, and 47.13 after subclause 47.9.2 as follows:

47.10 Property settings

47.10.1 The "PropertySettings" notation shall be:

PropertySettings ::= `SETTINGS` simplestring

47.10.2 The contents of the "simplestring" shall be "PropertySettingsList":

PropertySettingsList ::=
PropertyAndSettingPair
| PropertySettingsList PropertyAndSettingPair

PropertyAndSettingPair ::= PropertyName "=" SettingName

PropertyName ::= psname

SettingName ::= psname

47.10.3 The "PropertyName" shall be one of the time property names listed in column 1 of Table 5 *bis*, and shall appear at most once in the "PropertySettingsList".

47.10.4 The "SettingName" of a "PropertyAndSettingPair" shall be one of the property setting names that are listed in column 2 of Table 5 *bis* in the row that contains (in column 1) the "PropertyName" of that "PropertyAndSettingPair".

47.10.5 An abstract value shall be included in the subtype if, and only if, it satisfies the following condition for all of the "PropertyAndSettingPair"s. Either:

a) the abstract value does not have a property setting for the "PropertyName" (see columns 2 and 3 of Table 5 *bis* for the abstract values that have a property setting for a given "PropertyName"); or

b) the abstract value has a property setting that is the same as the "SettingName".

NOTE – To assist with human readability, it is recommended, but not required, that the setting of the `Basic` time property be always included as the first "PropertyAndSettingPair".

EXAMPLE: `TIME(SETTINGS "Midnight=Start")` would produce a subset of the `TIME` type in which all abstract values are present (including those that represent dates only) except those that have the property setting `"Midnight=End"`.

47.10.6 All abstract values of the `TIME` type have settings for the `Basic` time property (this is not true for other time properties). In order to prevent misleading notation in which a "PropertyAndSettingPair" has no effect on the resulting set of abstract values, some restrictions are placed on the "PropertyName"s that can be used with a specific setting of the `Basic` time property. The restrictions are listed in Table 9 *ter*.

NOTE – Table 9 *ter* is not an exhaustive set of rules for preventing the use of "PropertyAndSetting" pairs, some of which are redundant (which is not in general illegal).

Table 9 *ter* – Restrictions on use of property names with `Basic` property settings

`Basic` property setting	Prohibited property names with this `Basic` property setting
`Date`	`Time`, `Local-or-UTC`, `Midnight, Interval-type`, `SE-point`, `Recurrence`
`Time`	`Date`, `Year`, `Interval-type`, `SE-point`, `Recurrence`
`Date-Time`	`Interval-type`, `SE-point`, `Recurrence`
`Interval`	`Recurrence`
`Rec-Interval`	*No restriction*

47.11 Duration range

47.11.1 The "DurationRange" subtype notation shall be:

DurationRange ::= ValueRange

47.11.2 Both the "Value"s in the "ValueRange" shall identify a time abstract value (either by value notation or by a value reference) that is present in the subtype:

```
TIME(SETTINGS "Basic=Interval Interval-type=D")
```

47.11.3 Both the "Value"s in the "ValueRange" shall specify the duration using either:

a) the same single time component to the same accuracy (no fractional part, or the same number of digits in the fractional part); or

b) multiple time components that have identical values apart from the least significant time component (which may have different values, but shall have the same accuracy).

47.11.4 The selected duration abstract values are those that:

a) have the same values for the identical time components of the two "Value"s in the "ValueRange"; and

b) are within the specified range for the least significant time component of the two "Value"s in the "ValueRange"; and

c) have the same accuracy as the least significant time component of the two "Value"s in the "ValueRange".

NOTE – This provides an alternative to the use of inner subtyping (see 34 *bis*.4.4) as a means of specifying a duration that uses only a single time component to a specified accuracy.

EXAMPLE: `TIME("PT2M0.000S".."PT2M59.000S")` defines a `TIME` subtype that consists only of abstract values representing durations of 2 minutes and zero to 59 seconds, to an accuracy of one millisecond.

47.12 Time point range

47.12.1 The "TimePointRange" notation shall be:

TimePointRange ::= ValueRange

47.12.2 Both the "Value"s in the "ValueRange" shall identify a time abstract value (either value notation or a value reference) that is present in the subtype:

```
TIME ( (SETTINGS "Basic=Date")
|(SETTINGS "Basic=Time")
|(SETTINGS "Basic=DateTime") )
```

47.12.3 The two "Value"s in the "ValueRange" shall have identical settings for all time properties except the `Midnight` time property.

NOTE – This allows subtyping using, for example:

```
TIME ("00:00".."09:00")
```

or:

```
TIME ("21:00".."24:00").
```

47.12.4 This subtype notation selects from the parent type those abstract values that have identical settings for all time properties (except the `Midnight` time property) to those of the "Value"s in the "ValueRange", and that have values within the specified "ValueRange" (see 47.4).

NOTE – The requirement for all relevant abstract values to have identical settings for all time properties (except the `Midnight` time property) ensures that they are all using the same time-scale, and hence that an order relationship exists among them.

47.13 Recurrence range

47.13.1 The "RecurrenceRange" notation shall be:

RecurrenceRange ::= ValueRange

47.13.2 Both the "Value"s in the "ValueRange" shall be integer values.

47.13.3 For the purposes of ordering only, a time value with a property setting of `"Recurrence=Unlimited"` shall be treated as specifying an infinite number of repetitions (an integer value of `MAX`).

47.13.4 This subtype notation selects from the parent type those abstract values that are also present in the subtype:

```
TIME(SETTINGS "Basic=Rec-interval")
```

and that have recurrence values within the specified "ValueRange" (see 47.4).

21) New Annex A *bis*

Insert a new Annex A bis:

Annex A *bis*

The defined time types

(This annex forms an integral part of this Recommendation | International Standard)

A *bis*.1 General

A *bis*.1.1 This annex contains an ASN.1 module that specifies the defined time types. These types can be imported into an ASN.1 specification and used in that specification, or can be used as a model for the definition of additional time types. They cannot be used without importation.

A *bis*.1.2 In some cases, the defined time types are only useful if subtyped with one of the date or time-of-day subsets (or both) specified in this module. Where this is the case, it is clearly stated in the definition of the type.

EXAMPLE: Use

```
APPLICATION-DATE-TIME ::= DATE-TIME(YEAR-MONTH-DAY-SUBSET)(SECONDS-SUBSET)
```

to define a date-time that is a year, month, day, hours, minutes, seconds. To use this, the type and the two subtypes have to be imported.

A *bis*.2 The ASN.1 defined time types module

```
DefinedTimeTypes {joint-iso-itu-t asn1(1) specification(0) modules(0) defined-types-
module(3)}
DEFINITIONS AUTOMATIC TAGS ::= BEGIN

EXPORTS ALL;

-- Date types

CENTURY ::= TIME((SETTINGS "Basic=Date Date=C Year=Basic")|
                          (SETTINGS "Basic=Date Date=C Year=Proleptic"))

ANY-CENTURY ::= TIME((SETTINGS "Basic=Date Date=C Year=Negative")|
                              (SETTINGS "Basic=Date Date=C Year=L5"))
                          -- This allows only a 3-digit century if positive.
                          -- A type with a greater number of digits can be
                          -- defined as an additional time type.

YEAR ::= TIME((SETTINGS "Basic=Date Date=Y Year=Basic")|
                       (SETTINGS "Basic=Date Date=Y Year=Proleptic"))
```

```
ANY-YEAR ::= TIME((SETTINGS "Basic=Date Date=Y Year=Negative")|
                         (SETTINGS "Basic=Date Date=Y Year=L5"))
                        -- This allows only a 5-digit year if positive.
                        -- A type with a greater number of digits can be
                        -- defined as an additional time type.

YEAR-MONTH ::= TIME((SETTINGS "Basic=Date Date=YM Year=Basic")|
                           (SETTINGS "Basic=Date Date=YM Year=Proleptic"))

ANY-YEAR-MONTH ::= TIME((SETTINGS "Basic=Date Date=YM Year=Negative")|
                               (SETTINGS "Basic=Date Date=YM Year=L5"))
                        -- This allows only a 5-digit year if positive.
                        -- A type with a greater number of digits can be
                        -- defined as an additional time type.

YEAR-MONTH-DAY ::= TIME((SETTINGS "Basic=Date Date=YMD Year=Basic")|
                               (SETTINGS "Basic=Date Date=YMD Year=Proleptic"))

ANY-YEAR-MONTH-DAY ::= TIME((SETTINGS "Basic=Date Date=YMD Year=Negative")|
                                   (SETTINGS "Basic=Date Date=YMD Year=L5"))
                        -- This allows only a 5-digit year if positive.
                        -- A type with a greater number of digits can be
                        -- defined as an additional time type.

YEAR-WEEK ::= TIME((SETTINGS "Basic=Date Date=YW Year=Basic")|
                          (SETTINGS "Basic=Date Date=YW Year=Proleptic"))

ANY-YEAR-WEEK ::= TIME((SETTINGS "Basic=Date Date=YW Year=Negative")|
                              (SETTINGS "Basic=Date Date=YW Year=L5"))
                        -- This allows only a 5-digit year if positive.
                        -- A type with a greater number of digits can be
                        -- defined as an additional time type.

YEAR-WEEK-DAY ::= TIME((SETTINGS "Basic=Date Date=YWD Year=Basic")|
                              (SETTINGS "Basic=Date Date=YWD Year=Proleptic"))

ANY-YEAR-WEEK-DAY ::= TIME((SETTINGS "Basic=Date Date=YWD Year=Negative")|
                                  (SETTINGS "Basic=Date Date=YWD Year=L5"))
                        -- This allows only a 5-digit year if positive.
                        -- A type with a greater number of digits can be
                        -- defined as an additional time type.

-- Types related to time-of-day

HOURS ::= TIME(SETTINGS "Basic=Time Time=H Local-or-UTC=L")

HOURS-UTC ::= TIME(SETTINGS "Basic=Time Time=H Local-or-UTC=Z")

HOURS-AND-DIFF ::= TIME(SETTINGS "Basic=Time Time=H Local-or-UTC=LD")

MINUTES ::= TIME(SETTINGS "Basic=Time Time=HM Local-or-UTC=L")

MINUTES-UTC ::= TIME(SETTINGS "Basic=Time Time=HM Local-or-UTC=Z")

MINUTES-AND-DIFF ::= TIME(SETTINGS "Basic=Time Time=HM Local-or-UTC=LD")

SECONDS ::= TIME(SETTINGS "Basic=Time Time=HMS Local-or-UTC=L")

SECONDS-UTC ::= TIME(SETTINGS "Basic=Time Time=HMS Local-or-UTC=Z")

SECONDS-AND-DIFF ::= TIME(SETTINGS "Basic=Time Time=HMS Local-or-UTC=LD")

HOURS-AND-FRACTION ::= TIME(SETTINGS "Basic=Time Time=HF3 Local-or-UTC=L")
             -- 3 digit fraction

HOURS-UTC-AND-FRACTION ::= TIME(SETTINGS "Basic=Time Time=HF3 Local-or-UTC=Z")
                 -- 3-digit fraction

HOURS-AND-DIFF-AND-FRACTION ::= TIME(SETTINGS "Basic=Time Time=HF3
                   Local-or-UTC=LD")
                   -- 3-digit fraction

MINUTES-AND-FRACTION ::= TIME(SETTINGS "Basic=Time Time=HMF3 Local-or-UTC=L")
                -- 3-digit fraction

MINUTES-UTC-AND-FRACTION ::= TIME(SETTINGS "Basic=Time Time=HMF3 Local-or-UTC=Z")
                 -- 3-digit fraction
```

```
MINUTES-AND-DIFF-AND-FRACTION ::= TIME(SETTINGS "Basic=Time Time=HMF3
                    Local-or-UTC=LD")
                   -- 3-digit fraction

SECONDS-AND-FRACTION ::= TIME(SETTINGS "Basic=Time Time=HMSF3 Local-or-UTC=L")
                                       -- 3-digit fraction

SECONDS-UTC-AND-FRACTION ::= TIME(SETTINGS "Basic=Time Time=HMSF3 Local-or-UTC=Z")
                                           -- 3-digit fraction

SECONDS-AND-DIFF-AND-FRACTION ::= TIME(SETTINGS "Basic=Time Time=HMSF3
                    Local-or-UTC=LD")
                    -- 3-digit fraction

-- Interval types (DURATION is not included as this is a useful type).

START-END-DATE-INTERVAL ::= TIME(SETTINGS "Basic=Interval Interval-type=SE
                 SE-point=Date")
           -- This is only useful if subtyped with a DATE subset (see below).

START-END-TIME-INTERVAL ::= TIME(SETTINGS "Basic=Interval Interval-type=SE
                 SE-point=Time")
           -- This is only useful if subtyped with a TIME-OF-DAY subset
           -- (see below).

START-END-DATE-TIME-INTERVAL ::= TIME(SETTINGS "Basic=Interval Interval-type=SE
                 SE-point=Date-Time")
           -- This is only useful if subtyped with a DATE subset and a
           -- TIME-OF-DAY subset (see below).

START-DATE-DURATION-INTERVAL ::= TIME(SETTINGS "Basic=Interval Interval-type=SD
                 SE-point=Date")
           -- This is only useful if subtyped with a DATE subset (see below).

START-TIME-DURATION-INTERVAL ::= TIME(SETTINGS "Basic=Interval Interval-type=SD
                 SE-point=Time")
           -- This is only useful if subtyped with a TIME-OF-DAY subset
           -- (see below).

START-DATE-TIME-DURATION-INTERVAL ::= TIME(SETTINGS "Basic=Interval
                      Interval-type=SD
                      SE-point=Date-Time")
           -- This is only useful if subtyped with a DATE subset and a
           -- TIME-OF-DAY subset (see below).


DURATION-END-DATE-INTERVAL ::= TIME(SETTINGS "Basic=Interval Interval-type=DE
                 SE-point=Date")
           -- This is only useful if subtyped with a DATE subset (see below).

DURATION-END-TIME-INTERVAL ::= TIME(SETTINGS "Basic=Interval Interval-type=DE
                 SE-point=Time")
           -- This is only useful if subtyped with a TIME-OF-DAY subset
           -- (see below).

DURATION-END-DATE-TIME-INTERVAL ::= TIME(SETTINGS "Basic=Interval Interval-type=DE
                 SE-point=Date-Time")
           -- This is only useful if subtyped with a DATE subset and a
           -- TIME-OF-DAY subset (see below).

-- Recurring interval types.

REC-START-END-DATE-INTERVAL ::= TIME(SETTINGS "Basic=Rec-Interval Interval-type=SE
                 SE-point=Date")
           -- This is only useful if subtyped with a DATE subset (see below).

REC-START-END-TIME-INTERVAL ::= TIME(SETTINGS "Basic=Rec-Interval Interval-type=SE
                 SE-point=Time")
           -- This is only useful if subtyped with a TIME-OF-DAY subset
           -- (see below).

REC-START-END-DATE-TIME-INTERVAL ::= TIME(SETTINGS "Basic=Rec-Interval
                     Interval-type=SE
                     SE-point=Date-Time")
           -- This is only useful if subtyped with a DATE subset and a
           -- TIME-OF-DAY subset (see below).
```

```
REC-DURATION-INTERVAL ::= TIME(SETTINGS "Basic=Rec-Interval Interval-type=D")

REC-START-DATE-DURATION-INTERVAL ::= TIME(SETTINGS "Basic=Rec-Interval
                     Interval-type=SD
                     SE-point=Date")
           -- This is only useful if subtyped with a DATE subset (see below).

REC-START-TIME-DURATION-INTERVAL ::= TIME(SETTINGS "Basic=Rec-Interval
                     Interval-type=SD
                     SE-point=Time")
           -- This is only useful if subtyped with a TIME-OF-DAY subset
           -- (see below).

REC-START-DATE-TIME-DURATION-INTERVAL ::= TIME(SETTINGS "Basic=Rec-Interval
                     Interval-type=SD
                     SE-point=Date-Time")
           -- This is only useful if subtyped with a DATE subset and a
           -- TIME-OF-DAY subset (see below).


REC-DURATION-END-DATE-INTERVAL ::= TIME(SETTINGS "Basic=Rec-Interval
                     Interval-type=DE
                     SE-point=Date")
           -- This is only useful if subtyped with a DATE subset (see below).

REC-DURATION-END-TIME-INTERVAL ::= TIME(SETTINGS "Basic=Rec-Interval
                     Interval-type=DE
                     SE-point=Time")
           -- This is only useful if subtyped with a TIME-OF-DAY subset
           -- (see below).

REC-DURATION-END-DATE-TIME-INTERVAL ::= TIME(SETTINGS "Basic=Rec-Interval
                     Interval-type=DE
                     SE-point=Date-Time")
           -- This is only useful if subtyped with a DATE subset and a
           -- TIME-OF-DAY subset (see below).

-- Date subsets

CENTURY-SUBSET ::= TIME((SETTINGS "Date=C Year=Basic")|
                             (SETTINGS "Date=C Year=Proleptic"))

ANY-CENTURY-SUBSET ::= TIME((SETTINGS "Date=C Year=Negative")|
                                 (SETTINGS "Date=C Year=L5"))

YEAR-SUBSET ::= TIME((SETTINGS "Date=Y Year=Basic")|
                          (SETTINGS "Date=Y Year=Proleptic"))

ANY-YEAR-SUBSET ::= TIME((SETTINGS "Date=Y Year=Negative")|
                              (SETTINGS "Date=Y Year=L5"))

YEAR-MONTH-SUBSET ::= TIME((SETTINGS "Date=YM Year=Basic")|
                         (SETTINGS "Date=YM Year=Proleptic"))

ANY-YEAR-MONTH-SUBSET ::= TIME((SETTINGS "Date=YM Year=Negative")|
                                    (SETTINGS "Date=YM Year=L5"))

YEAR-MONTH-DAY-SUBSET ::= TIME((SETTINGS "Date=YMD Year=Basic")|
                                    (SETTINGS "Date=YMD Year=Proleptic"))

ANY-YEAR-MONTH-DAY-SUBSET ::= TIME((SETTINGS "Date=YMD Year=Negative")|
                                        (SETTINGS "Date=YMD Year=L5"))

YEAR-WEEK-SUBSET ::= TIME((SETTINGS "Date=YW Year=Basic")|
                               (SETTINGS "Date=YW Year=Proleptic"))

ANY-YEAR-WEEK-SUBSET ::= TIME((SETTINGS "Date=YW Year=Negative")|
                                   (SETTINGS "Date=YW Year=L5"))

YEAR-WEEK-DAY-SUBSET ::= TIME((SETTINGS "Date=YWD Year=Basic")|
                                   (SETTINGS "Date=YWD Year=Proleptic"))

ANY-YEAR-WEEK-DAY-SUBSET ::= TIME((SETTINGS "Date=YWD Year=Negative")|
                                       (SETTINGS "Date=YWD Year=L5"))
```

```
-- Time subsets

HOURS-SUBSET ::= TIME(SETTINGS "Time=H Local-or-UTC=L")

HOURS-UTC-SUBSET ::= TIME(SETTINGS "Time=H Local-or-UTC=Z")

HOURS-AND-DIFF-SUBSET ::= TIME(SETTINGS "Time=H Local-or-UTC=LD")

MINUTES-SUBSET ::= TIME(SETTINGS "Time=HM Local-or-UTC=L")

MINUTES-UTC-SUBSET ::= TIME(SETTINGS "Time=HM Local-or-UTC=Z")

MINUTES-AND-DIFF-SUBSET ::= TIME(SETTINGS "Time=HM Local-or-UTC=LD")

SECONDS-SUBSET ::= TIME(SETTINGS "Time=HMS Local-or-UTC=L")

SECONDS-UTC-SUBSET ::= TIME(SETTINGS "Time=HMS Local-or-UTC=Z")

SECONDS-AND-DIFF-SUBSET ::= TIME(SETTINGS "Time=HMS Local-or-UTC=LD")

HOURS-AND-FRACTION-SUBSET ::= TIME(SETTINGS "Time=HF3 Local-or-UTC=L")

HOURS-UTC-AND-FRACTION-SUBSET ::= TIME(SETTINGS "Time=HF3
                                               Local-or-UTC=Z")

HOURS-AND-DIFF-AND-FRACTION-SUBSET ::= TIME(SETTINGS "Time=HF3
                       Local-or-UTC=LD")

MINUTES-AND-FRACTION-SUBSET ::= TIME(SETTINGS "Time=HMF3
                       Local-or-UTC=L")

MINUTES-UTC-AND-FRACTION-SUBSET ::= TIME(SETTINGS "Time=HMF3
                       Local-or-UTC=Z")

MINUTES-AND-DIFF-AND-FRACTION-SUBSET ::= TIME(SETTINGS "Time=HMF3
                       Local-or-UTC=LD")

SECONDS-AND-FRACTION-SUBSET ::= TIME(SETTINGS "Time=HMSF3
                       Local-or-UTC=L")

SECONDS-UTC-AND-FRACTION-SUBSET ::= TIME(SETTINGS "Time=HMSF3
                       Local-or-UTC=Z")

SECONDS-AND-DIFF-AND-FRACTION-SUBSET ::= TIME(SETTINGS "Time=HMSF3
                       Local-or-UTC=LD")

END
```

22) New clause E.2 *bis*

Insert new clause E.2 bis before clause E.3 as follows:

E.2 *bis* Value notation and property settings (`TIME` type and useful time types)

This subclause provides examples of value notation for the time type. The same value notation is used for the useful time types, but is restricted to denotation of abstract values that are present in those types. Each example gives a time abstract value in normal human notation, then a value assignment for that value, using a useful time type if there is one that contains it, otherwise using the **TIME** type. The following comment gives the settings needed to define a subtype of the **TIME** type that contains all similar abstract values.

E.2 *bis*.1 Date

EXAMPLES

Calendar date – 12 April 1985:

```
date1 DATE ::= "1985-04-12" -- Basic=Date Date=YMD Year=Basic
```

Ordinal date – 12 April 1985:

```
date2 TIME ::= "1985-102" -- Basic=Date Date=YD Year=Basic
```

Week date – Friday 12 April 1985:

```
date3 TIME ::= "1985-W15-5" -- Basic=Date Date=YWD Year=Basic
```

Calendar week – 15th week of 1985:

```
date4 TIME ::= "1985-W15" -- Basic=Date Date=YW Year=Basic
```

Calendar month – April 1985:

```
date5 TIME ::= "1985-04" -- Basic=Date Date=YM Year=Basic
```

Calendar year – 1985:

```
date6 TIME ::= "1985" -- Basic=Date Date=Y Year=Basic
```

Calendar date – 12 April 11985:

```
date7 TIME ::= "+011985-04-12" -- Basic=Date Date=YMD Year=L6
```

The 12th April in the 2nd year before the year 0000:

```
date8 TIME ::= "-0002-04-12" -- Basic=Date Date=YMD Year=Negative
```

The 20th century:

```
date9 TIME ::= "19C" -- Basic=Date Date=C Year=Basic
```

E.2 *bis*.2 Time of day

EXAMPLES

27 minutes and 46 seconds past 15 hours:

```
time1 TIME-OF-DAY ::= "15:27:46"
             -- Basic=Time Time=HMS Local-or-UTC=L
```

To the nearest minute:

```
time2 TIME ::= "15:28"
             -- Basic=Time Time=HM Local-or-UTC=L
```

Local time with decimal fractions using comma – 27 minutes and 35 and a half second past 15 hours:

```
time3 TIME ::= "15:27:35,5"
             -- Basic=Time Time=HMSF1 Local-or-UTC=L
```

UTC – 20 minutes and 30 seconds past 23 hours:

```
time4 TIME ::= "23:20:30Z"
             -- Basic=Time Time=HMS Local-or-UTC=Z
```

To the nearest hour:

```
time5 TIME ::= "23Z"
             -- Basic=Time Time=H Local-or-UTC=Z
```

Local time of the day and the difference from UTC – 27 minutes 46 seconds past 15 hours locally in Geneva (one hour ahead of UTC):

```
time6 TIME ::= "15:27:46+01:00"
             -- Basic=Time Time=HMS Local-or-UTC=LD
```

Alternative value notation for the same abstract value:

```
time7 TIME ::= "15:27:46+01"
             -- Basic=Time Time=HMS Local-or-UTC=LD
```

27 minutes 46 seconds past 15 hours locally in New York (five hours behind UTC):

```
time8 TIME ::= "15:27:46-05:00"
             -- Basic=Time Time=HMS Local-or-UTC=LD
```

E.2 *bis*.3 Date and time of day

EXAMPLES

Combination of calendar date and local time of day:

```
date-time1 DATE-TIME ::= "1985-04-12T10:15:30"
             -- Basic=Date-Time Date=YMD Year=Basic Time=HMS
             -- Local-or-UTC=L
```

Combination of ordinal date and UTC:

```
date-time2 TIME ::= "1985-102T23:50:30Z"
             -- Basic=Date-Time Date=YD Year=Basic Time=HMS Local-or-UTC=Z
```

Combinations of week date and local time of the day:

```
date-time3 TIME ::= "1985-W14-5T23:50:30"
             -- Basic=Date-Time Date=YWD Year=Basic Time=HMS
             -- Local-or-UTC=L
```

E.2 *bis*.4 Time interval

EXAMPLES

A time interval starting at 20 minutes and 50 seconds past 23 hours on 12 April 1985 and ending at 30 minutes past 10 hours on 25 June 1985:

```
interval1 TIME ::= "1985-04-12T23:20:50/1985-06-25T10:30:00"
             -- Basic=Interval Interval-type=SE SE-point=Date-Time
             -- Date=YMD Year=Basic Time=HMS Local-or-UTC=L
```

A time interval starting at 12 April 1985 and ending on 25 June 1985:

```
interval2 TIME ::= "1985-04-12/1985-06-25"
             -- Basic=Interval Interval-type=SE SE-point=Date
             -- Date=YMD Year=Basic
```

Alternative value notation for the same abstract value:

```
interval3 TIME ::= "1985-04-12/1985-06-25"
             -- Basic=Interval Interval-type=SE SE-point=Date
             -- Date=YMD Year=Basic
```

A time interval of 2 years, 10 months, 15 days, 10 hours, 20 minutes and 30 seconds:

```
duration1 DURATION ::= "P2Y10M15DT10H20M30S"
             -- Basic=Interval Interval-type=D
```

A time interval of 1 year and 6 months:

```
duration2 DURATION ::= "P1Y6M"
             -- Basic=Interval Interval-type=D
```

A time interval of seventy-two hours:

```
duration3 DURATION ::= "PT72H"
             -- Basic=Interval Interval-type=D
```

A time interval of 1 year, 2 months, 15 days and 12 hours, beginning on 12 April 1985 at 20 minutes past 23 hours:

```
interval4 TIME ::= "1985-04-12T23:20:00/P1Y2M15DT12H"
             -- Basic=Interval Interval-type=SD SE-point=Date-Time
             -- Date=YMD Year=Basic Time=HMS Local-or-UTC=L
```

A time interval of 1 year, 2 months, 15 days and 12 hours, ending on 12 April 1985 at 20 minutes past 23 hours:

```
interval5 TIME ::= "P1Y2M15DT12H/1985-04-12T23:20:00"
             -- Basic=Interval Interval-type=DE SE-point=Date-Time
             -- Date=YMD Year=Basic Time=HMS Local-or-UTC=L
```

E.2 *bis*.5 Recurring interval

EXAMPLES

Fifteen recurrences of a time interval of 2 years, 10 months, 15 days, 10 hours, 20 minutes and 30 seconds:

```
rec-int1 TIME ::= "R15/P2Y10M15DT10H20M30S"
             -- Basic=Rec-Interval Recurrence=Unlimited Interval-type=D
```

An unbounded number of recurrences of a time interval of 2 years, 15 days, 10 hours, 20 minutes and 30 seconds:

```
rec-int2 TIME ::= "R/P2Y15DT10H20M30S"
             -- Basic=Rec-Interval Recurrence=Unlimited Interval-type=D
```

Two recurrences of a time interval of 1 year and 6 months:

```
rec-int3 TIME ::= "R2/P1Y6M"
             -- Basic=Rec-Interval Recurrence=Unlimited Interval-type=D
```

An unbounded number of occurrences of a time interval of 1 year, 2 months, 15 days and 12 hours of which the last occurrence ends at 12 April 1985 at 20 minutes and 50 seconds past 23 hours:

```
rec-int4 TIME ::= "R/P1Y2M15DT12H/1985-04-12T23:20:50"
             -- Basic=Rec-Interval Recurrence=Unlimited Interval-type=DE
             -- SE-point=Date-Time Date=YMD Year=Basic Time=HMS
             -- Local-or-UTC=L
```

23) New clause E.4.8

Insert new clause E.4.8 after clause E.4.7 as follows:

E.4.8 Examples of subtyping the time type are present in 34 *bis*.4, and several useful settings are given in the comments in E.2 *bis*. Additional examples follow, with comments. Note that all examples of subtyping can also be applied to the useful time types, but will only select abstract values that are already present in those types. The main use of this notation is to provide variations on the useful time types.

EXAMPLES

```
My-Date ::= TIME
             (SETTINGS "Basic=Date Year=Basic Date=YD")
             -- A date type that uses years and days

My-Date1 ::= TIME
             (SETTINGS "Basic=Date Year=Basic Date=YD")
             ("2000-001" .. < "2011-001")
             -- A date type that uses years and days restricted to the
             -- period from the 1st Jan. AD 2000 to Dec. 31st AD 2010, inclusive.


My-Date2 ::= TIME
             ("2000-001" .. < "2011-001")
             -- The same date type as My-Date1, but this is probably less
             -- clear to a human user.  It relies on the property settings
             -- being deduced from the value notation (see Annex G ter).

My-Illegal-Date ::= TIME
             ("1500-00" .. < "2011-00")
             -- The lower bound is a proleptic date, and the upper bound
             -- is a basic date, so they do not have the same properties,
             -- and this is illegal.

My-time-of-day-1 ::= TIME
             (SETTINGS "Basic=Time Time=HMS Local-or-UTC=L
              Midnight=Start")
             -- This is the same as TIME-OF-DAY, but midnight at the end of
             -- the day is excluded, with the only midnight being represented
             -- by the value notation "00:00:00".

My-time-of-day-2 ::= TIME
             (SETTINGS "Basic=Time Time=HMS Local-or-UTC=L
              Midnight=End")
             -- This is the same as TIME-OF-DAY, but midnight at the start of
             -- the day is excluded, with the only midnight being represented
             -- by the value notation "24:00:00".

My-time-of-day-3 ::= TIME
             (SETTINGS "Basic=Time Time=HMS Local-or-UTC=Z")
             -- This is the same as TIME-OF-DAY, but the time is UTC, not
             -- local time.
```

24) New Annexes G *bis* and G *ter*

Insert new Annexes G bis and G ter as follows:

Annex G *bis*

Tutorial annex on the TIME type

(This annex does not form an integral part of this Recommendation | International Standard)

G *bis*.1 The collections of ASN.1 types for times and dates

G *bis*.1.1 Historically, ASN.1 defined its own time type, **UTCTime**, as a "UsefulType", because it was based on specifying the contents of a **VisibleString** type. Later, **GeneralizedTime** was added, allowing a four-digit year, and defined by reference to a set of standards that were the predecessors of the first (1988) version of ISO 8601, to specify the contents of a **VisibleString**. (The other "UsefulType", defined by specifying the contents of a **GraphicString**, was **ObjectDescriptor**.) Traditionally, the so-called "UsefulType"s have used mixed upper-lower case letters for their type reference names, with the other built-in ASN.1 types using only upper-case letters. The useful types have, however, their own **UNIVERSAL** class tags, and can be referenced independently in Encoding Rules specifications.

G *bis*.1.2 While these types (**UTCTime**, **GeneralizedTime** and **ObjectDescriptor**) are undoubtedly useful, the separation of them from other types by the term "UsefulType"s (simply because they are defined in terms of other – character string – types), and the use of mixed upper-lower case in their type reference names has been increasingly recognized as a historical accident.

G *bis*.1.3 With the introduction of time types to support the 2004 version of ISO 8601, it was recognized that a primary time type (**TIME**) was needed, but that a number of commonly useful time types (**DATE**, **TIME-OF-DAY**, **DATE-TIME** and **DURATION**), defined as subsets of the basic **TIME** type (using the ASN.1 subtype notation), were needed. A decision was taken to call these "Useful time types", and to give them names that were all upper case, in order to minimize backwards compatibility problems, as they are new reserved words. They all have distinct **UNIVERSAL** class tags that are distinct from the tag of the **TIME** type (to enable optimized BER encodings), and are all listed under the production "BuiltinType" (see 16.2).

G *bis*.2 ISO 8601 key concepts

G *bis*.2.1 ISO 8601 provides the definitive reference for identification of instants of time and for their character representation. It forms the basis for the specification of the ASN.1 **TIME** type, both in terms of time-related concepts and in terms of actual representations used in ASN.1 value notation and in the Basic Encoding Rules (BER).

G *bis*.2.2 ISO 8601 is based entirely on the Gregorian calendar introduced in 1582, together with the so-called proleptic Gregorian calendar that extends the Gregorian calendar sequentially backwards in time from 1582, using the normal rules for the definition of common (non-leap) years and leap years. There is in general no easy way to determine a date AD or BC using the Julian calendar from a date specified using the proleptic Gregorian calendar, but in particular the year 1 AD is roughly (but not exactly) in alignment with year 1 proleptic Gregorian, and the year 1 BC (the preceding year) is roughly (but not exactly) in alignment with year 0 proleptic Gregorian.

G *bis*.2.3 Key definitions and concepts in ISO 8601 include the concept of multiple time-scales for the time axis. Each time-scale consists of an ordered set of marks on the time axis. Each mark represents a time point (an instant of time).

G *bis*.2.4 Three main time-scales are defined in ISO 8601.

G *bis*.2.4.1 The first is called the calendar date time-scale. This has marks corresponding to calendar years, calendar months, and the ordinal number of a day within its calendar month (days are numbered 01 to 28, 29, 30 or 31, depending on the month).

G *bis*.2.4.2 The second is called the ordinal date time-scale. This has marks corresponding to calendar years, and the ordinal number of a day within its calendar year (days are numbered 001 for Jan. 1st to 365 or 366, depending on the year).

G *bis*.2.4.3 The third is called the week date time-scale. This has marks corresponding to calendar years, the ordinal number of a week within that calendar year, and the ordinal number of a day within that week (with day 1 being Monday). Weeks are numbered 01 to 52 or 53 (depending on the year), with week 01 being defined as the week containing Jan. 4th, and the last week of the previous year being defined as the previous week to that (which is why some years contain 53 weeks).

G *bis*.2.5 Between the day marks on each time-scale are hour, minute, and second marks. However, the time axis is a continuum of instants of time, and all three of the time-scales also contain marks that are everywhere dense on the time axis.

NOTE – Another way of expressing this is to say that between any two marks there are infinitely more other marks, each identifying a decimal fraction of a second to arbitrarily large accuracy.

G *bis*.2.6 A variation on the calendar date time-scale is a time-scale in which seconds are not represented, but between each minute time point are infinitely more other marks representing decimal fractions of that minute to arbitrarily large accuracy. The same is true for decimal fractions of an hour.

NOTE – There is no concept in ISO 8601 of specifying a time point using decimal fractions of a day or any larger unit of time, although decimal fractions of a year, a month, a week or a day can be used in specifying a duration.

G *bis*.2.7 Because the rational number 1/60 does not have a terminating decimal representation, there are some time points on the time-scale using seconds that cannot be expressed as time points on the time-scale using fractions of a minute, in any finite representation.

G *bis*.2.8 Similarly, it is not possible to identify which mark for a day on one time-scale corresponds to the mark for a day on a different time-scale, without knowledge of which years are leap years. A similar problem arises with leap seconds with the identification of time intervals using scales based on a start point and an end point, or on a start point and a duration (in seconds, say), or on a duration and an end point.

G *bis*.2.9 ISO 8601 also recognizes the concept of identification of marks with varying accuracy. Thus on any given time-scale there can be different marks at the same time point, one specifying it as (for example) 3.100 seconds and the other specifying it as 3.1 seconds.

NOTE – In earlier ASN.1 work on time types (**`UTCTime`** and **`GeneralizedTime`**), the issue of having separate abstract values for the same time point expressed with different accuracies was not addressed. In the case of the **`TIME`** type, marks on the time axis that have different accuracy, but that are placed at the same time point, are firmly identified as distinct abstract values. Thus an abstract value that might be represented by 3.100 is distinct from one that might be represented by 3.1, and may carry different application semantics.

G *bis*.2.10 Control of the accuracy used, and of some other aspects of ISO 8601, is stated in ISO 8601 to be "by mutual agreement". In general, where ISO 8601 identifies areas requiring mutual agreement, notations are provided in ASN.1 for an application designer to specify in the ASN.1 type definition the mutual agreements that are to be assumed. This is done by selecting subsets of the multiple infinities of abstract values in the **`TIME`** type, using time properties associated with each time abstract value.

G *bis*.2.11 ISO 8601 recognizes the concept of time difference. This is the difference between local time and UTC for a particular World Time Zone. There is no international authority for agreement on or recording of time differences for different World Time Zones. This is a matter for local administrations, although HM Nautical Almanac Office (UK) attempts to maintain an authoritative record of currently assigned time differences for all parts of the world. As at 2005, time differences in the range –12 to +14 have been defined by various local administrations. To allow for possible future changes, ASN.1 supports time differences in the range –15 to +16 (only).

G *bis*.3 Abstract values of the `TIME` type

G *bis*.3.1 Each mark on each time-scale, with each accuracy, is identified as a distinct abstract value of the **`TIME`** type, and thus has a distinct ASN.1 value notation and distinct encodings in all ASN.1 Encoding Rules.

G *bis*.3.2 ISO 8601 is predominantly concerned with the identification of time points, but distinguishes between identification of date only, of time of day only, and of date and time. These different identifications also produce distinct abstract values in the **`TIME`** type.

G *bis*.3.3 Within the identification of time of day, local time or UTC or both can be used. Again, these different identifications produce distinct and unrelated abstract values because of the different time-scales that are being used (and will generally carry different application semantics).

G *bis*.3.4 Another feature of ISO 8601 is the identification of time intervals using either a start and end point (which can be identified using any of the various time-scales), a duration, or a duration with either a start or an end point. Again, these provide four main sets of abstract values that are distinct from abstract values representing time points, but with many subsets of those main sets of time intervals, depending on the abstract values used in the specification of the start and end point of the time interval, or the time components used in specifying a duration.

NOTE – In ASN.1, it is not possible to use different time-scales for the start point and end point of a time interval. This is for simplicity of specification, and is not expected to be a problem for application designers.

G *bis*.3.5 Finally, ISO 8601 has the concept of specifying a recurring time interval. Recurring time intervals map into abstract values that are distinct from those representing time intervals and time points.

G *bis*.4 Time properties of the time abstract values

G *bis*.4.1 It is possible to identify sets of time abstract values that have common time properties. Some time properties (such as whether it is a time point, a time interval, or a recurring time interval) apply to all time abstract values. Other time properties, such as whether a time interval is expressed as a start point and a duration, or by a duration and an end point (for example), apply only to time abstract values that are time intervals. Similarly, the time property indicating whether a time abstract value represents local time, UTC or both, applies only to time abstract values with at least one component related to time-of-day.

G *bis*.4.2 Clause 34 *bis*.2 and Table 5 *bis* specify the complete set of time properties that can be associated with a time abstract value, and the possible settings for each of those time properties. The setting of a time property can be used in the subtype notation to select subsets of the **TIME** type, all of whose abstract values have the same setting for a given time property.

NOTE – The term "setting of a time property" rather than "value of a time property" is used to avoid confusion with the use of "value" in the term "abstract value".

G *bis*.4.3 The presence of some time properties on a time abstract value is dependent on the setting of other time properties, as outlined above.

G *bis*.4.4 In the subtype notation, abstract values of the **TIME** type are specified using a list of time property and setting pairs. There are restrictions on the combinations of time properties and settings that can be specified (see 47.10.6), but the order of the property and setting pairs does not matter (but see G *bis*.4.5). An abstract value of the **TIME** type is included in the subtype if and only if it has the specified setting for all the listed properties that are applicable to it. As usual, it is an illegal ASN.1 specification if the resulting set of abstract values assigned to a type reference is empty (although empty sets are not prohibited in set arithmetic).

G *bis*.4.5 To provide clarity for human readers, and to avoid errors, it is recommended, but not required, that the order of specification of property and setting pairs proceeds from major properties (such as **"Basic=Date-Time"**) to more detailed properties (such as **"TIME=HMS"**). This would generally mean a specification of time property and setting pairs in the order of Table 5 *bis*. This convention is used in all examples in this Recommendation | International Standard (see 34 *bis*.4, E.2 *bis* and E.4.8).

G *bis*.4.6 It is important for interval specification, and for subtyping using value ranges, to have an order relationship on the abstract values of the **TIME** type. In general, there is an order relationship between abstract values representing time points (based on their position on the time axis) if, and only if, they all have the same time property settings. Similarly, the number of seconds between two time point abstract values can in general only be determined if they have the same property settings. This means that time points used in some of the subtype notations and in interval specification are required to have the same property settings. An order relationship is defined for durations only if they have the same accuracy and differ only in a single time component (see 47.11).

G *bis*.5 Value notation

G *bis*.5.1 The value notation (and encoding) for an abstract value depends on its associated time properties and their settings. The value notation is specified in 34 *bis*.3.

G *bis*.5.2 ISO 8601 in general specifies two separate (character-based) representations, called a basic format and an extended format, for identifying marks in a time-scale.

G *bis*.5.3 In general, the basic format is a simple string of digits, with non-digit separators (such as a decimal separator) only where needed to provide unambiguous representations within commonly useful subsets of the abstract values implied by ISO 8601. This format is not used by ASN.1 value notation.

NOTE – For example, in the basic format, the string `2020` represents both the year `2020` and also the time 8:20 pm.

G *bis*.5.4 The extended format contains additional non-digit separators designed to make the representation more readable for human users, and is generally (but with one exception – see Note 1 below) unambiguous over all the abstract values implied by ISO 8601. The extended format is recommended by ISO 8601 for use in plain text.

NOTE 1 – The exception is the representation of a date that is four digits representing a century, which can be confused with four digits representing a year. In the ASN.1 value notation, a `C` is added to the century notation, for all century representations, including two-digit representations, to resolve the ambiguity.

NOTE 2 – For example, in the extended format, the year `2020` would be represented by `2020`, but the time 8:20 pm would be represented by `20:20`.

G *bis*.5.5 The use of the extended format (with the added uppercase `C` for centuries) enables the property settings of the abstract value being represented to be determined from the value notation, and that notation unambiguously identifies an abstract value, knowing only that its type is the `TIME` type.

G *bis*.5.6 The basic format requires knowledge of some of the property settings of the abstract value that is being represented, in order to resolve ambiguity in the representations, and is not used in ASN.1.

G *bis*.5.7 The basic ASN.1 value notation and the XML value notation (specified in 34 *bis*.3), and the XML Encoding Rules specified in ITU-T Rec. X.693 | ISO/IEC 8825-4, use the ISO 8601 extended format. The Basic Encoding Rules specified in ITU-T Rec. X.690 | ISO/IEC 8825-1 use the ISO 8601 extended format (but with the removal of some designators and separators – such as P for duration, colons in time-of-day, and hyphens in dates). Different ASN.1 tags are assigned to the useful types to enable BER to identify property settings that are needed to resolve what would otherwise be ambiguity in the encoding of the useful time types. The Packed Encoding Rules specified in ITU-T Rec. X.691 | ISO/IEC 8825-2 use a binary encoding that is unrelated to (and out of the scope of) ISO 8601. PER encodings provide very compact representations of date, time and duration (typically 17 bits for a date, 15 bits for a time, 32 bits (4 octets) for a date-time, and often less than 16 bits for a duration).

G *bis*.6 Use of the ASN.1 subtype notation

G *bis*.6.1 Six forms of subtype notation (plus inner subtyping in restricted cases – see G *bis*.6.8) are permitted for this type (see clause 47 and Table 9 *bis*).

NOTE – Examples of subtype notation for the `TIME` type and the useful time types can be found in 34 *bis*.4, E.2 *bis* and E.4.8.

G *bis*.6.2 A property settings subtype notation allows the selection of all abstract values with a given setting for one or more listed time properties. This is the normal means of producing additional customized time types for applications, and is discussed more fully in G *bis*.7.

G *bis*.6.3 Single value subtypes are permitted, but are not expected to be generally useful.

G *bis*.6.4 Contained subtypes are permitted, and are expected to be commonly used in the specification of customized time types by application designers.

G *bis*.6.5 Duration range subtypes (containing an ordered pair of durations) can be applied. They select from the parent type only those abstract values that are time intervals specified as a duration, and constrain the duration to the specified range (see 47.11).

G *bis*.6.6 Time point range subtypes (containing an ordered pair of time points) can be applied. They select from the parent type only those abstract values that are time points with the same property settings as the two ends of the time point range (which are required to have the same property settings), and constrain the time point to be within the specified range.

NOTE – This subtype constraint restricts the range of values of a time point and is totally separate from the direct use of value notation to identify a single time abstract value that is a time interval.

G *bis*.6.7 Recurrence range subtypes (containing an ordered pair of integers) can be applied. They select from the parent type only those abstract values that are recurring time intervals and constrain the number of recurrences to be within the specified range.

G *bis*.6.8 Inner subtyping can be applied if the time type has already been restricted to a duration (typically by use of the `DURATION` useful time type). This enables restrictions to be placed on the form of a duration specification.

G *bis*.6.9 No other forms of subtype constraint are permitted.

G *bis*.7 The property settings subtype notation

G *bis*.7.1 Time abstract values that have the same setting for a given time property form natural subsets of the time type. The property settings subtype notation enables the selection of abstract values by listing the setting of one or more time properties. An abstract value is included in the resulting subtype if, and only if, it has the specified setting for all the listed properties that are applicable to it.

EXAMPLE: The following notation can be used to define a time subtype that contains all abstract values that are a date only, specified using a four-digit year, week-of-the-year, and day:

```
My-time ::= TIME (SETTINGS "Basic=Date Date=YWD Year=Basic")
```

A more extensive set of examples of the subtype notation is given in 34 *bis*.4, E.2 *bis* and E.4.8.

G *bis*.7.2 ASN.1 set arithmetic (or simply application of multiple constraints) can be used in the normal way to define combinations (using **INTERSECTION**, **UNION** and **EXCEPT**) of time subtypes, to produce types that are appropriate for use in a particular application.

G *bis*.7.3 The time properties, their names, possible settings, and the abstract values that have this setting are specified in Table 5 *bis*.

G *bis*.7.4 A small number of useful time subtypes are specified using the property settings subtype notation (see 34 *bis*.4), and are given human-friendly names. It is expected that these useful subtypes (**DATE**, **TIME-OF-DAY**, **DATE-TIME**, and **DURATION**) will be sufficient for many applications. A more extensive set of defined time types of general utility are specified in the ASN.1 defined time types module in Annex A *bis*. These types can be imported and used, either directly, or to define application-specific time types. They support the full functionality of ISO 8601. Additionally, where necessary, designers can define additional types as subtypes of the **TIME** type or the useful or defined time types using the property setting subtype notation. These types can be further combined using ASN.1 set arithmetic.

NOTE – The useful time types have been given ASN.1 **UNIVERSAL** class tags that are different from the **TIME** type in order to permit efficient encodings in BER. They should be regarded as independent types rather than as subtypes, but can also be used in a contained subtype constraint if the parent type is **TIME**.

Annex G *ter*

Analysing `TIME` type value notation

(This annex does not form an integral part of this Recommendation | International Standard)

G *ter*.1 General

G *ter*.1.1 The body of this Recommendation | International Standard specifies the value notation for abstract values with given time properties.

G *ter*.1.2 Every instance of this value notation unambiguously identifies a single abstract value of the time type, and its properties.

G *ter*.1.3 This informative annex describes one possible algorithm for determining the time property settings of the abstract value that is represented by an instance of the value notation. There are many alternative (and probably better) algorithms, and this annex is provided simply as a demonstration that such algorithms exist.

> NOTE – If this algorithm is applied to a random string, it will identify that the string can only represent an abstract value with a given set of time property settings. It is then necessary to check that the syntax of the string conforms to that required for an abstract value with those property settings before the notation can be accepted and the abstract value identified.

G *ter*.1.4 If two abstract values have the same property settings, then their value notation differs only in the values of the digits present in the notation, with the following exceptions:

a) there are several different representations of duration abstract values, depending on which time units are being used, and on whether decimal fractions are being used;

b) either comma or full stop can be used as decimal separators;

c) a time difference component that is an integral number of hours may be expressed with hours only or with hours and minutes;

d) the end point of an interval expressing UTC may omit the time difference component if the time difference is the same as the time difference on the start point;

e) abstract values that differ only by having a plus or a minus for the time difference component nonetheless have the same time properties.

G *ter*.1.5 If two strings differ only in the actual value of the digits present in the strings, then they have the same property settings with the exceptions that the use of year dates in the range 0000 to 1581 means that the date has the property setting `"Year=Proleptic"` and not `"Year=Basic"` (similarly for century notation).

G *ter*.2 Analysing the full string

G *ter*.2.1 If the string commences with an LATIN CAPITAL LETTER R, then it has the property setting `"Basic=Rec-Interval"`. The "`R`" will be followed by a number of recurrences (empty string for unbounded), then a SOLIDUS ("/"). If the portion of the string after "`R`" and before "/" is empty, then it has the property setting `"Recurrence=Unlimited"`. Otherwise, if the number of digits in the portion of the string after "`R`" and before "/" is 1, 2, 3, etc., then it has the property setting `"Recurrence=R1"`, `"Recurrence=R2"`, `"Recurrence=R3"`, etc. The remainder of the string can be analysed as a string containing an interval (see G *ter*.3) to determine additional property settings.

G *ter*.2.2 Otherwise, if the string contains a solidus ("/"), then it has the property setting `"Basic=Interval"`, and can be analysed as a string containing an interval (see G *ter*.3) to determine additional property settings.

G *ter*.2.3 Otherwise, if the string commences with a LATIN CAPITAL LETTER P, then it has the settings `"Basic=Interval"` and `"Interval-type=D"`, completing the analysis of properties.

G *ter*.2.4 Otherwise, if the string contains a LATIN CAPITAL LETTER T, then it has the property setting `"Basic=Date-Time"`, and the portion of the string before the "`T`" can be analysed as a string containing a date (see G *ter*.4) and the portion following the "`T`" can be analysed as a string containing a time (see G *ter*.7) to determine further property settings.

G *ter*.2.5 Otherwise, if the string ends in a LATIN CAPITAL LETTER C, then it has the property setting `"Basic=Date"` and `"Date=C"` and can be analysed as a string containing a century (see G *ter*.6) to determine further property settings.

G *ter*.2.6 Otherwise, if the string contains a COLON (":"), or has less than four characters, then it has the property setting **`"Basic=Time"`** and can be analysed as a string containing a time (see G *ter*.7) to determine further property settings.

G *ter*.2.7 Otherwise, it has the property setting **`"Basic=Date"`** and can be analysed as a string containing a date (see G *ter*.4) to determine further property settings.

G *ter*.3 Analysis of a string containing an interval

G *ter*.3.1 If the string begins with a LATIN CAPITAL LETTER P and does not contain a SOLIDUS ("/"), then it has the property setting **`"Interval-type=D"`**, completing the analysis of properties.

G *ter*.3.2 If the string contains a SOLIDUS, then either the portion of the string before the solidus or the portion after the solidus will commence with a LATIN CAPITAL LETTER P, or neither will commence with a "P" (but not both). If:

a) the portion before the SOLIDUS begins with a "P" then it has the property setting **`"Interval-type=DE"`**, and may have further properties by analysing the portion after the SOLIDUS as specified in subsequent subclauses of this G *ter*.3;

b) the portion after the SOLIDUS begins with a "P" then it has the property setting **`"Interval-type=SD"`**, and may have further properties by analysing the portion before the SOLIDUS as specified in subsequent subclauses of this G *ter*.3;

c) if neither portion begins with a "P" then it has the property setting **`"Interval-type=SE"`**, and may have further properties by analysing the portion before the SOLIDUS as specified in subsequent subclauses of this G *ter*.3.

NOTE – There is a requirement in ASN.1 (but not in ISO 8601) that the end point have the same time property settings as the start point. This means that the end point portion of the string does not need to be analysed in the determination of property settings. It should, however, be noted that there are permitted representations of the end point that omit the time difference component if it is the same as the time difference in the start point. This needs to be considered when determining the abstract value of the end point.

G *ter*.3.3 If the portion contains a LATIN CAPITAL LETTER T, then the part of the portion before the "T" can be analysed as a string containing a date (see G *ter*.4) and the part following the "T" can be analysed as a string containing a time (see G *ter*.7) to determine further property settings.

G *ter*.3.4 If the portion ends in a LATIN CAPITAL LETTER C, then it has the property setting **`"Date=C"`** and can be analysed as a string containing a century (see G *ter*.6) to determine further property settings.

G *ter*.3.5 Otherwise, if the portion contains a COLON (":"), then it can be analysed as a string containing a time (see G *ter*.7) to determine further property settings.

G *ter*.3.6 Otherwise, it can be analysed as a string containing a date (see G *ter*.4) to determine further property settings.

G *ter*.4 Analysis of a string containing a date

G *ter*.4.1 If the string ends in a LATIN CAPITAL LETTER C, then it has the property setting **`"Date=C"`** and the rest of the string can be analysed as a string containing a century (see G *ter*.6).

G *ter*.4.2 If the string begins with a HYPEN-MINUS ("-"), this should be ignored for the analysis in the rest of this subclause G *ter*.4.

NOTE – In this case, the hyphen represents a minus sign, not a separator.

G *ter*.4.3 Otherwise, the string will contain zero, one, or two HYPHEN-MINUS ("-") characters, and in the last two cases may or may not contain a LATIN CAPITAL LETTER W.

G *ter*.4.4 If the string does not contain a LATIN CAPITAL LETTER W, then if:

a) the string contains no HYPHEN-MINUS characters, it has the property setting **`"Date=Y"`** and can be analysed as a string containing a year (see G *ter*.5) to determine further property settings;

b) the string contains one HYPHEN-MINUS character, it has the property setting **`"Date=YM"`**; two digits for the month will follow the HYPEN-MINUS and the portion before the HYPHEN-MINUS can be analysed as a string containing a year (see G *ter*.5) to determine further property settings;

c) the string contains two HYPHEN-MINUS characters, it has the property setting **`"Date=YMD"`**; two digits for the month will follow the first HYPHEN-MINUS, two digits for the day will follow the second HYPHEN-MINUS, and the portion of the string before the first HYPHEN-MINUS can be analysed as a string containing a year (see G *ter*.5) to determine further property settings.

G *ter*.4.5 If the string contains a LATIN CAPITAL LETTER W, then if:

a) the string contains one HYPHEN-MINUS character, it has the property setting `"Date=YW"`; two digits for the week will follow the HYPHEN-MINUS and the portion before the HYPHEN-MINUS can be analysed as a string containing a year (see G *ter*.5) to determine further property settings.

b) If the string contains two HYPHEN-MINUS characters, it has the property setting `"Date=YWD"`; two digits for the week will follow the first HYPHEN-MINUS, one digit for the day will follow the second HYPHEN-MINUS, and the portion of the string before the first HYPHEN-MINUS can be analysed as a string containing a year (see G *ter*.5) to determine further property settings.

G *ter*.5 Analysis of a string containing a year

G *ter*.5.1 If the string commences with a HYPHEN-MINUS ("-") character, then it has the property setting `"Year=Negative"`, completing the analysis of properties.

G *ter*.5.2 Otherwise, if the string is more than four characters, it has the property setting `"Year=L5"`, `"Year=L6"`, `"Year=L7"`, etc., for a number of characters equal to 5, 6, 7, etc., respectively, completing the analysis of properties.

G *ter*.5.3 Otherwise, if the value of the four-digit string is less than 1582, then it has the property setting `"Year=Proleptic"`, completing the analysis of properties.

G *ter*.5.4 Otherwise, it has the property setting `"Year=Basic"`, completing the analysis of properties.

G *ter*.6 Analysis of a string containing a century

G *ter*.6.1 If the string commences with a HYPHEN-MINUS ("-") character, it has the property setting `"Year=Negative"`, completing the analysis of properties.

G *ter*.6.2 Otherwise, if the string is more than two digits, it has the property setting `"Year=L5"`, `"Year=L6"`, `"Year=L7"`, etc., for a number of digits equal to 3, 4, 5, etc., respectively, completing the analysis of properties.

G *ter*.6.3 Otherwise, if the value of the two-digit string is less than 15, it has the property setting `"Year=Proleptic"`, completing the analysis of properties.

G *ter*.6.4 Otherwise, it has the property setting `"Year=Basic"`, completing the analysis of properties.

G ter.7 Analysis of a string containing a time

G *ter*.7.1 If the string ends in a LATIN CAPITAL LETTER Z then it has the property setting `"Local-or-UTC=Z"`, and the portion of the string before the "Z" can be analysed as a string containing a simple time (see G *ter*.8) to determine further property settings.

G *ter*.7.2 Otherwise, if the string contains a plus ("+") or a minus ("-") then it has the property `"Local-or-UTC=LD"`, and the portion of the string before the plus or minus can be analysed as a simple time (see G *ter*.8) to determine further property settings.

NOTE – Analysis of the portion of the string following the plus or minus (a time differential) is not needed for the determination of property settings.

G *ter*.7.3 Otherwise, it has the property `"Local-or-UTC=L"` and can be analysed as a simple time (see G *ter*.8) to determine further property settings.

G *ter*.8 Analysis of a string containing a simple time

G *ter*.8.1 The string will contain zero, one or two colons (:) and may contain a decimal sign, which is a full stop (".") or comma (",").

G *ter*.8.2 If the string does not contain a decimal sign, then if:

a) the string does not contain a colon, it has the property setting `"Time=H"`, completing the analysis of properties;

b) the string contains one colon, it has the property setting `"Time=HM"`, completing the analysis of properties;

c) the string contains two colons, it has the property setting `"Time=HMS"`, completing the analysis of properties.

G *ter*.8.3 If the string contains a decimal sign, then if:

a) the string does not contain a colon, it has the property setting `"Time=HF1"`, `"Time=HF2"`, `"Time=HF3"`, etc., if the number of digits after the decimal sign is 1, 2, 3, etc., respectively, completing the analysis of properties;

b) the string contains one colon, it has the property setting `"Time=HMF1"`, `"Time=HMF2"`, `"Time=HMF3"`, etc., if the number of digits after the decimal sign is 1, 2, 3, etc., respectively, completing the analysis of properties;

c) the string contains two colons, then it has the property setting `"Time=HMSF1"`, `"Time=HMSF2"`, `"Time=HMSF3"`, etc., if the number of digits after the decimal sign is 1, 2, 3, etc., respectively, completing the analysis of properties.

25) Annex H

*In Annex H, add the following lines after the lexical item "***xmlcstring***":*

simplestring

tstring

xmltstring

psname

26) Annex H

In Annex H, add the following lines:

After `CONTAINING`:

```
DATE

DATE-TIME
```

After `DEFINITIONS`:

```
DURATION
```

After `SET`:

```
SETTINGS
```

After `TeletexString`:

```
TIME

TIME-OF-DAY
```

After "**ChoiceType**" in "**BuiltinType**":

| **DateType**
| **DateTimeType**
| **DurationType**

After "**PrefixedType**" in "**BuiltinType**":

| **TimeType**
| **TimeOfDayType**

After "**PrefixedValue**" in "**BuiltinValue**":

| **TimeValue**

After "**XMLPrefixedValue**" in "**XMLBuiltinValue**":

| **XMLTimeValue**

After "**ExternalType** ::= **EXTERNAL**":

TimeType ::= TIME

TimeValue ::= tstring

XMLTimeValue ::= xmltstring

DateType ::= DATE

TimeOfDayType ::= TIME-OF-DAY

DateTimeType ::= DATE-TIME

DurationType ::= DURATION

After "**PatternConstraint**" in "**SubtypeElements**":

| PropertySettings
| DurationRange
| TimePointRange
| RecurrenceRange

After "**PatternConstraint** ::= **PATTERN Value**":

PropertySettings ::= SETTINGS simplestring

PropertySettingsList ::=
PropertyAndSettingPair
| PropertySettingsList PropertyAndSettingPair

PropertyAndSettingPair ::= PropertyName "=" SettingName

PropertyName ::= psname

SettingName ::= psname

DurationRange ::= ValueRange

TimePointRange ::= ValueRange

RecurrenceRange ::= ValueRange

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Cable networks and transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Construction, installation and protection of cables and other elements of outside plant
Series M	Telecommunication management, including TMN and network maintenance
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	Switching and signalling
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	**Data networks, open system communications and security**
Series Y	Global information infrastructure, Internet protocol aspects and next-generation networks
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland
Geneva, 2006