SLOVENSKI STANDARD
SIST-TS CEN/TS 14796:2004

01-september-2004

Zdravstvena informatika – Vrste podatkov

Health Informatics - Data Types

Medizinische Informatik - Datentypen

Informatique de santé - Types de données

Ta slovenski standard je istoveten z: **CEN/TS 14796:2004**

ICS:

35.240.80	Uporabniške rešitve IT v zdravstveni tehniki	IT applications in health care technology

SIST-TS CEN/TS 14796:2004 **en**

TECHNICAL SPECIFICATION

SPÉCIFICATION TECHNIQUE

TECHNISCHE SPEZIFIKATION

CEN/TS 14796

June 2004

ICS 35.240.80

English version

Health Informatics - Data Types

Élément introductif - Élément central

Medizinische Informatik - Datentypen

This Technical Specification (CEN/TS) was approved by CEN on 9 February 2004 for provisional application.

The period of validity of this CEN/TS is limited initially to three years. After two years the members of CEN will be requested to submit their comments, particularly on the question whether the CEN/TS can be converted into a European Standard.

CEN members are required to announce the existence of this CEN/TS in the same way as for an EN and to make the CEN/TS available promptly at national level in an appropriate form. It is permissible to keep conflicting national standards in force (in parallel to the CEN/TS) until the final decision about the possible conversion of the CEN/TS into an EN is reached.

CEN members are the national standards bodies of Austria, Belgium, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Netherlands, Norway, Poland, Portugal, Slovakia, Slovenia, Spain, Sweden, Switzerland and United Kingdom.

EUROPEAN COMMITTEE FOR STANDARDIZATION
COMITÉ EUROPÉEN DE NORMALISATION
EUROPÄISCHES KOMITEE FÜR NORMUNG

Management Centre: rue de Stassart, 36 B-1050 Brussels

Ref. No. CEN/TS 14796:2004: E

Contents

Page

List of Figures

List of Tables

Foreword

This document CEN/TS 14796:2004 has been prepared by Technical Committee CEN/TC 251 “Health Informatics”, the secretariat of which is held by SIS.

According to the CEN/CENELEC Internal Regulations, the national standards organizations of the following countries are bound to announce this Technical Specification: Austria, Belgium, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Netherlands, Norway, Poland, Portugal, Slovakia, Slovenia, Spain, Sweden, Switzerland and United Kingdom.

Introduction

ISO standards have existed for data types for some time, and especially significant is ISO 11404 (1994) Language Independent Data types. Since 1994 most data typing has been based upon or harmonised with this standard. However, in healthcare information communication a different source of 'data type standardisation' has arisen, sourced by Health Level Seven (HL7), whose data types often resemble but are not wholly compatible with ISO 11404.

In developing this Technical Specification there has been the wish to harmonise with the HL7 data types so that the health informatics industry in Europe and the USA can more easily be aligned. To this end a collaboration agreement was entered into in March 2000 between CEN/TC 251 and HL7. The goal was set for a maximum degree of alignment while maintaining their independence and need to serve the business requirements of the respective markets but also to make the results available to ISO for possible international standardisation.

This Technical Specification differs from the HL7 abstract data types in two major ways:

- Primitive data types are not re-defined here – they are assumed to be available in any engineering environment that implements this specification. All other data types are defined here
- This TS says nothing about operations that are associated with any particular data type. Where a data type has been inherited from previously issued international standards the associated operations may be assumed but are not specifically referred to in this document.

In most other respects, this specification may be regarded as a sub set of the HL7 Version 3 abstract data types although partly described differently due to the fact that CEN is following the ISO rules for drafting and presentation of standards which HL7 is not.

This version attempts to include insights gained as a result of recent CEN/HL7 harmonisation, and to correct errors, where they were known to exist, particularly with respect to improving the technical quality of the underlying abstract models. There are still some incomplete specifications, which are indicated where appropriate.

The distinction between Primitive and Constructed data types has been tightened up.

Constructed types are divided into 4 packages for convenience (Basic, Text, Quantity, Time Specification)

It is intended that, following further harmonisation work with HL7, and after there has been an opportunity to gain implementation experience of using this specification, the TS will be developed into a full European Standard (EN). By that stage the UML model will be extended to include all necessary operations and formal invariants, together with attribute optionalities. For information on this process, see www.centc251.org.

1 Scope

This Technical Specification defines abstract data types for use in communicating healthcare information and for other health informatics purposes.

2 Normative references

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

ISO/IEC 11404	Information technology -- Programming languages, their environments and system software interfaces -- Language-independent datatypes
ISO 8601:2000(E)	Data elements and interchange formats – Information interchange – Representation of dates and times Second edition 2000-12-15
ISO/IEC 8824-1	Information technology -- Abstract Syntax Notation One (ASN.1): Specification of basic notation
ISO/IEC 10646-1	Information technology -- Universal Multiple-Octet Coded Character Set (UCS) -- Part 1: Architecture and Basic Multilingual Plane.
IEEE 754-1985	Standard for Binary Floating-Point Arithmetic
ISO 639:1988 (E/F)	Code for the representation of names of languages

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply

3.1
data type
a set of distinct values, characterised by properties of those values and by operations on those values

3.2
date
identification of a particular calendar day, expressed by some combination of the data elements calendar year, calendar month, calendar week, calendar day or day of the year

3.3
implementable technology specification
description of how to implement data types in a particular context (organisation, country, programming language etc)

3.4
literal form
concise character string representation of information of a certain data type

EXAMPLE: the decimal digit string “1234” as a valid literal form for some number data types

3.5
null flavor
reason for the absence of a valid data value

3.6
period of time (= time-interval)
portion of time between two time points

NOTE: A period of time is often also referred to as period

3.7
recurring time-interval
series of consecutive time-intervals of the same duration

3.8
time-point
instant in the laps of time regarded as dimensionless

3.9
value domain
set of valid data values for a data type

3.10
vocabulary domain
value domain for coded values

4 Abbreviations

BIN	Binary Data
BL	Boolean Data Value
CD	Concept Descriptor
CE	Coded with Equivalents
CR	Concept Role
CS	Coded Simple Value
CV	Coded Value
DV	Data Value
ED	Encapsulated Data
EIVL	Event Related Periodic Interval of Time
HL7	Health Level Seven
II	Instance Identifier
ITS	Implementable Technology Specification
IVL	Interval
OID	ISO object identifier
ORD	Ordinal
PIVL	Periodic Interval of Time

PQ Physical Quantity

RTO Ratio

ST Simple Text

TS Time Point

URI Universal Resource Identifier

UTC Coodinated Universal Time

XML Extensible Markup Language

5 Introduction to abstract data type definitions

The classes that form the abstract model of the data types are grouped for convenience into Packages as shown in Fig 1 below

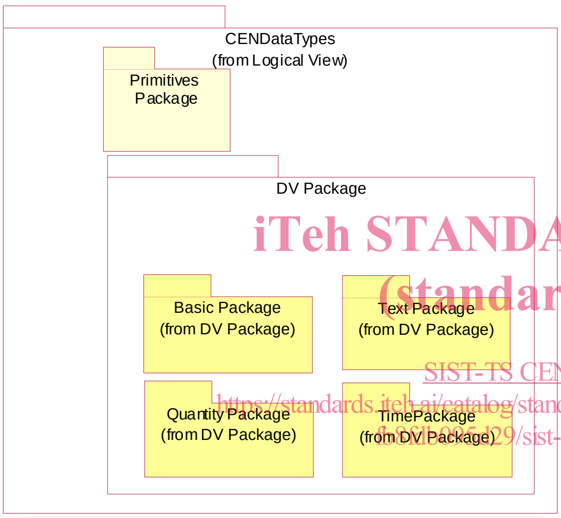

Figure 1: Packages

NOTE The names of Primitive classes (those that adopt industry standard definitions) are shown in lower case Capitalised, whereas constructed classes defined in this specification are have their names shown in CAPITALS.

5.1 Data values and data types

Data types define the meaning (semantics) of data values that can be assigned to a data element. Meaningful exchange of data requires that we have a shared understanding and agreed definitions for the types of data used in the exchange.

According to ISO 11404, a data type is "a set of distinct values, characterised by properties of those values and by operations on those values". **This TS restricts itself inasmuch as it does not yet define operations on data values for any of its data types.**

A data type defines the properties exposed by every data value of that type. Data types have a set of data values that are of that type (i.e. the type's "value set").

A semantic property of a data type is referred to by a name and has a value for each data value. The value of a data value's property must itself be a value defined by a data type - no data value exists that would not be defined by a data type.

Data types are thus the basic building blocks used to construct any higher order meaning: messages, computerised patient record documents, or business objects and their transactions.

5.2 Representation of Data Values

Data values can be represented through various symbols but the data value's meaning is not bound to any particular representation.

The number five can be represented by the word "five" by the Arabic number "5" or the Roman number "V". The representation does not matter so long as it conforms to the semantic definition of the data type.

Another example, the Boolean data type is defined by its extension, the two distinct values *true* and *false* and the rules of negation and combining these values in conjunction and disjunction. The representation of Boolean values can be the words "true" and "false," "yes" and "no," the numbers 0 and 1, any two signs that are distinct from each other. The representation of data types does not matter as long as it conforms to the semantic definition of the data type.

This specification defines the semantics, the meaning of the data types**,** independent of representational and operational concerns or specific implementation technologies.

Additional standards for representing the data values defined here are being specified for various technological approaches, e.g. for XML. These standards are called "Implementable Technology Specification" (ITS) and extend the basic specification provided in this document. A variety of ITS could therefore be defined for the set of data types specified here. These standards would define how values are represented so that they conform to the semantic definitions of this specification and may include syntaxes for character or binary representations, and computer procedures to act on the representation of data values.

5.3 Properties of Data Values

Data values have properties defined by their data type. The "fields" of "composite data types" are the most common example of such properties. The properties of a data type should be considered as logical predicates or as mathematical functions.

A property is referred to by its name. For example, the data type *integer* may have a property named "sign." A property has a domain, which is the set of possible "answer" values. The set of possible "answer" values is defined by the property's data type, but the domain of a property may be a subset of the data type's value set.

Any concrete implementation of these information model standards must ultimately use the built-in data types of their implementation technology. Therefore, we need a very flexible mapping between abstract data types and those data types built into any specific implementation technology.

This specification only requires that the properties defined for data values can somehow be inferred from whatever representation is chosen, it does not matter how these values are represented. For example, a decimal representation, a floating-point register and a scaled integer are all possible native representations of real numbers for different implementation technologies. Some of these representations have properties that others do not have. Scaled integers, for instance, have a fixed precision and a relatively small range. Floating-point values have variable precision and a large range, but floating-point values lose any information about precision. Decimal representations are of variable precision and maintain the precision information (yet are slow to process.) The data type semantics must be independent from all these accidental properties of the various representations, and must define the essential properties that any technology should be able to represent.

5.4 Characteristics of the data types

These data types may be characterised by:

- type name and possibly a short name
- informal description
- use
- specialisation of
- attributes and their semantic properties
- syntax of character string value literals (if any)

6 Primitive data types

6.1 Introduction

The following types are ‘atomic’ inasmuch as they are not defined as composites of other data types.

Each of these primitive types is already the subject of existing international standards and has very similar definitions in different implementation technologies.

This document does not attempt to redefine these primitive types but provides references to the standards where they are defined.

6.2 Primitive type set

6.2.1

The following list is set of primitive data types that are referenced and utilised within this specification. In addition to the primitive data types, three ‘primitive’ abstract types are described. These are Numeric, Integer and Real. They have no attributes of their own but are used as generalisations of certain of the concrete types which are their specialisations.

- Boolean
- String
- Integer
- Byte
- Real
- Double
- Character
- List<T>
- Set<T>
- Bag<T>
- Array<T>