

Default data structure for data carrier cession

Version 1.1

Explanations concerning the storage and description
of data within the scope of the GDPdU/GoBD



Copyright
Audicon GmbH
Meitnerstr. 6
70563 Stuttgart
Germany

Audicon did its utmost to ensure the quality of the information supplied in this document. All the information is supplied without warranty of any kind. Audicon GmbH hereby disclaims all warranties and conditions with regard to this information whether express, implied or statutory.

The document is protected by international copyright law. Translating, printing, copying images, tables or the publishing of the whole document or parts of it requires the written permission of Audicon GmbH.

IDEA is a registered trademark of CaseWare International Inc.

Default data structure – version 1.1 of 1 August 2002

Change	Date	Author(s)
Default data structure – Version 1.0	6/6/2002	Jon Chick
Default data structure – Version 1.1	1/8/2002	Jon Chick

Table 1: Default data structure change history

Document version 1.1 of 1 August 2002

Change	Date	Author(s)
Document version 1.0	6/6/2002	Jon Chick, Tankred Giese, Martin Otto
Document version 1.1 Explanations regarding references Expansions version 1.1	1/8/2002	Martin Otto, Jon Chick, Tankred Giese, Michael Schleupen

Table 2: Document change history

Contents

Target audience for this paper.....	5
Summary/objective.....	5
Text of § 147 paragraph 6 AO (German Fiscal Code).....	6
Data carrier cession.....	6
Organization of data during data carrier cession	8
The role of the software manufacturer.....	9
XML-based default data structure.....	10
Overview.....	10
General assumptions or requirements for the DTD	10
Technical sequence of the import process	11
The XML DTD.....	12
Organization of the default data structure	15
Notes regarding XML	19
Description of elements	20
Example 1 of an index.xml file	35
Example 2 regarding links.....	38
Example 3 of an index.xml file (compressed data)	40
Frequently asked questions	41
Synopsis of default data structure version 1.0 and version 1.1	44
List of figures.....	50
List of tables	51

Target audience for this paper

- Users of software instructed to provide a data medium/data package pursuant to GDPdU/GoBD.
- Developers of software providing tax relevant data for data carrier cession pursuant to GDPdU/GoBD.

Summary/objective

Pursuant to § 147 paragraph 6 AO (German Fiscal Code), internal revenue is entitled to audit the data of electronic accounting systems “digitally” either through data carrier cession and/or through indirect or alternatively direct access.

For data carrier cession, the data must be provided in a “machine-analyzable form” by the business subject to taxation (or by the commissioned tax advisor, accounting firm/subcontractor etc.) on suitable data media.

The term “machine-analyzability” from the perspective of internal revenue refers to optional access to all stored data, including the master data and links with sorting and filtering functions.

In order to achieve such analyzability and usability, it is necessary to define and standardize the file formats for data carrier cession.

This document presents a procedure to facilitate the defined and standardized data carrier cession and to outline a standard for the software industry.

Should you have any queries, please write to GDPdU@audicon.net, or obtain information from <https://www.audicon.net> (in German).

Text of § 147 paragraph 6 AO (German Fiscal Code)

In the version of the act to lower tax rates and to reform business taxation (Tax Reduction Act – StSenkG) of 23/10/2000, BGBl. (Federal Gazette) I 2000, p. 1433 (non-official translation):

- (6) In the event that documentation has been prepared pursuant to paragraph 1 with the aid of a data processing system, internal revenue, within the scope of an external audit, has the right to view stored data and to use the data processing system to audit such documentation. Within the scope of an external audit, internal revenue can also require that the data be analyzed automatically as specified by it, or that the stored data and copies be made available to it on a machine-utilizable data medium. The costs involved are borne by the party subject to taxation.

Data carrier cession

A possible scenario for a tax audit involving a data carrier cession:

An auditor is commissioned to audit a business. For specific auditing fields and auditing periods, the auditor decides to analyze the data with the aid of auditing software. Accordingly, the business concerned will be provided with notice of the audit and will be notified of the areas on which the audit will concentrate. He or she requests the business to make the tax relevant data available on a machine-analyzable data medium as specified by the “Principles of data access and auditing of digital documents” (GDPdU) pursuant to the German Federal Ministry of Finance communication of 16/07/2001 and the “Principles of duly keeping and retaining accounting books, records and documents in electronic form and of accessing data” (GoBD) pursuant to the German Federal Ministry of Finance communication of 28/11/2019.

Upon commencement of the audit, the auditor receives a data medium or data package.

In line with the focus of auditing, the auditor searches for the relevant tables and then imports these. The data medium/data package contains tax relevant data and descriptive data to allow the data to be imported without further clarifications. As soon as the data have been imported into the analysis software, the auditor carries out various analyses (in part also automated by means of macros).



1. Requests data medium/data package
2. Imports from data medium/data package
3. Carries out analysis or runs analysis macros

Figure 1: Auditor

From the perspective of the business, the scenario described above involves the following work steps:

- An employee initiates the extraction of data or exporting from the production database, during which process data are potentially (automatically) transformed in the sense of changing the way the data are organized, the structure of the tables or the structure of the objects.
- Similarly, in the case of compressed or encrypted data, the compression or encryption must be removed. It is then additionally necessary to create or copy a description in the form of a machine-analyzable file to accompany the tax relevant data. A file of this type must be created during the process of provision in a case where the software manufacturer is unable to foresee which data are to be provided during the audit. It is possible to provide a static file if the scope and structure of the data are certain beforehand.

The data are subsequently saved automatically or manually to a portable data medium/data package.

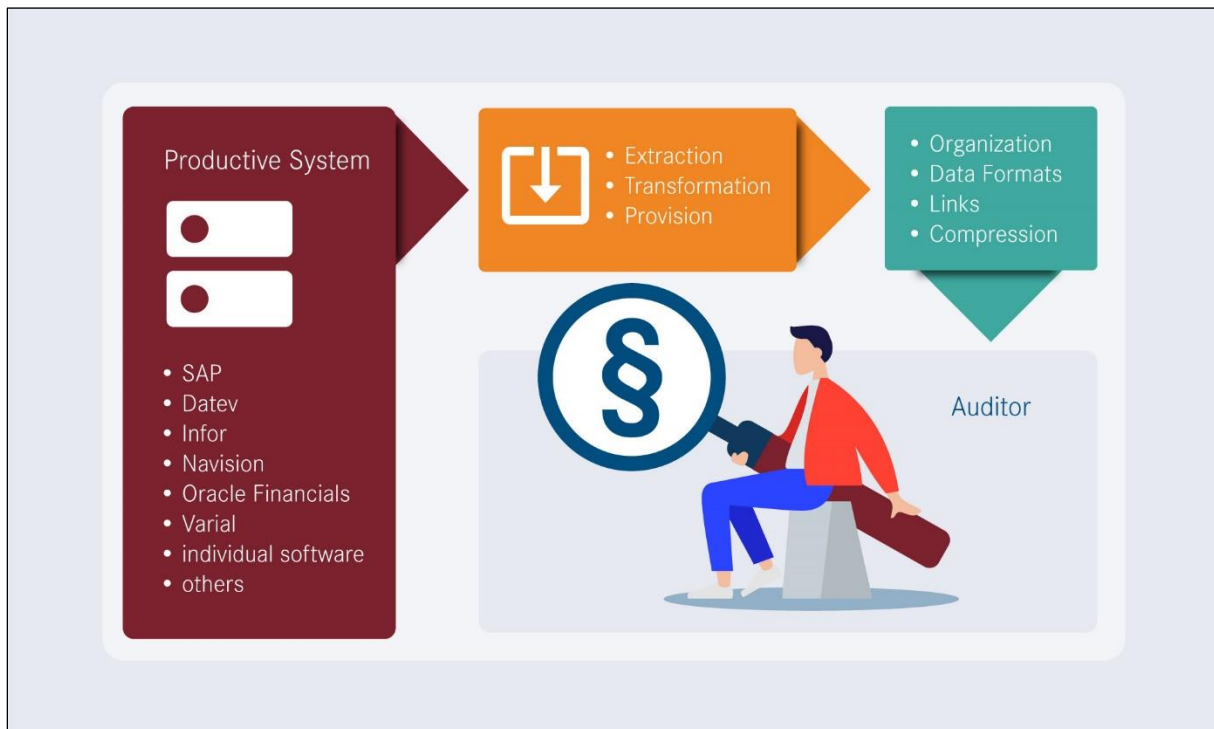


Figure 2: Data carrier session

Organization of data during data carrier cession

The German Federal Ministry of Finance communication of 16/07/2001 as a matter of principle provides audited companies substantial leeway for the technical organization of the data medium/data package to be transferred.

After the tax authorities of the German federation and federal states have procured analysis software in common use for auditing (IDEA), the auditors are in a position to read and process data mechanically (automated or manually). The financial authority does not define the data fields and contents saved to the data medium/data package during data carrier cession. It is therefore necessary that parties subject to taxation and/or manufacturers of software who process tax relevant data decide themselves which data are to be provided during data carrier cession. Depending on the individual EDP system, software structures, business structure and size, a variety of data contents may here be sensible and necessary.

In order to minimize queries by auditors regarding the content of data and formats or to render these unnecessary, a procedure is described here which describes and comments on the data, data structures and links.

In order to be able to process the various data structures, the party subject to taxation or his/her advisor makes the tax relevant data available in a largely denormalized form. Additionally, he/she provides a machine-analyzable description of the data, data formats and links. The file format for the tax relevant data has been standardized to common default formats. Both tax relevant data and description data are made available on a common data medium/in a common data package.

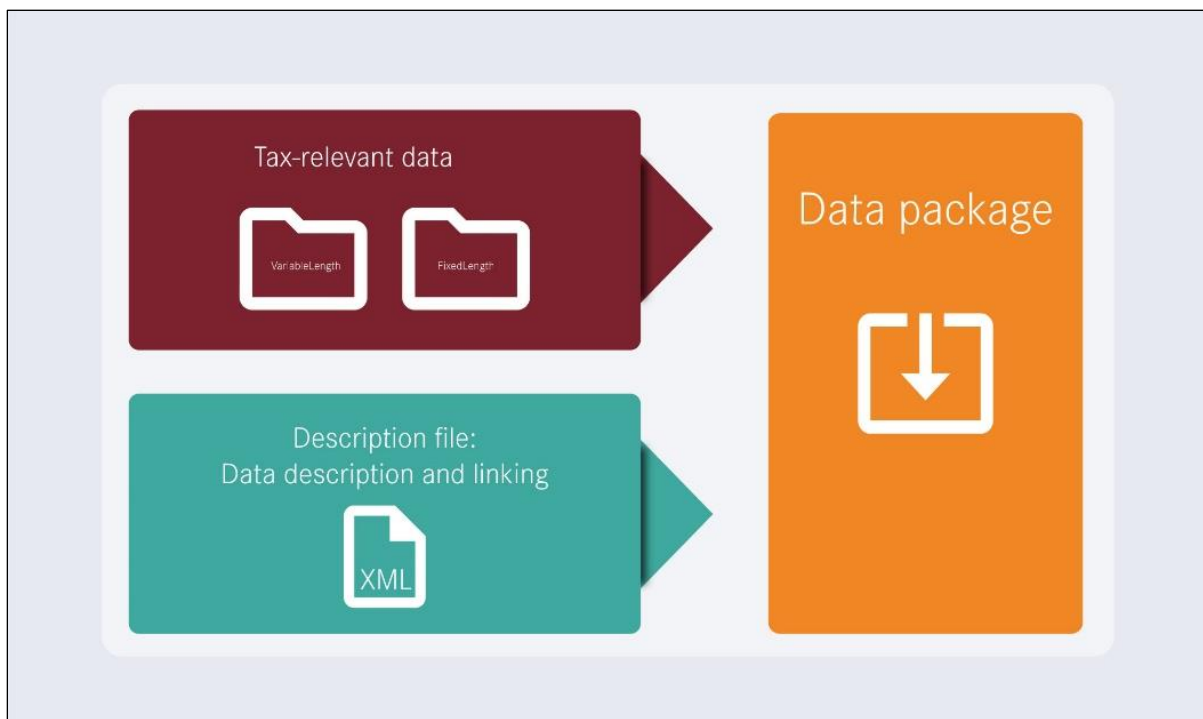


Figure 3: Data organization

The supported file formats for tax relevant data are:

- VariableLength
- FixedLength

In this way, it is possible for the software systems to use pre-existing export filters or storage mechanisms.

Internal revenue does not permit any installation of special software in order to decrypt or uncompress data on its systems. However, during conventional mailing of data media, at minimum, the encryption of data is required for data protection reasons. The default data structure therefore, for example, provides for programs for decryption or uncompressing to be started directly from the data medium without these installing themselves on the internal revenue system.

The computers of the auditors are equipped with access protection software and where applicable also with anti-virus software. The functionality of these directly accessed programs should therefore be clarified by the data provider in advance. Testing whether these routines can be executed does not form part of this default data structure.

The role of the software manufacturer

In what follows, a “software manufacturer” – from the perspective of the business subject to taxation – is assumed to refer to a supplier of software or a data processing system, including programs, or an internal department that makes relevant programs or services available.

In line with the outlined scenarios for auditing, amongst others, a machine-analyzable description of data and links needs to be made available during auditing.

The description file is read by a program that, in line with this description, controls the import function of the analysis program of the tax auditor. No operations are carried out on databases of the productive systems.

The manufacturer of the programs or procedures by means of which the business subject to taxation processes tax relevant data (software manufacturer) must in addition to extraction functionality ensure the formatting or provision of the description functionality.

XML-based default data structure

Overview

XML stands for “eXtensible Markup Language”. XML only prescribes a generic set of language elements, so that an individual language needs to be defined based on each individual instance of application. The in each instance newly formulated language can be stored in a machine-readable form by means of structure information and element definitions in a DTD (Document Type Definition). In this way, an XML parser can check whether an XML document corresponds to the agreed language (validation). A distinction is therefore drawn between the language description in the DTD and the contents in the XML instance.

Within the DTD itself, no descriptive data are therefore stored for the data carrier session, but instead in the XML file called index.xml. The DTD describes the structure of the index.xml file.

Additional information regarding XML is available at <https://www.w3.org/>.

General assumptions or requirements for the DTD

In the DTD, it is assumed that the tax relevant data are organized into files. Per table, exactly one file is assumed, so that organization according to file and table coincide physically.

The descriptive data for the data carrier session are saved into a file called index.xml.

The DTD needs to be located in the same directory on the data medium/in the data package as the index.xml file.

In the event that more than one data medium is provided, the index.xml file must only be located on the first data medium.

Technical sequence of the import process

The import process is controlled by a component that interprets the index.xml file and transfers the export tables from the individual data media/data packages to the analysis software. This component checks the index.xml file for correctness and serves as a table of contents for the delivered data for the auditor.

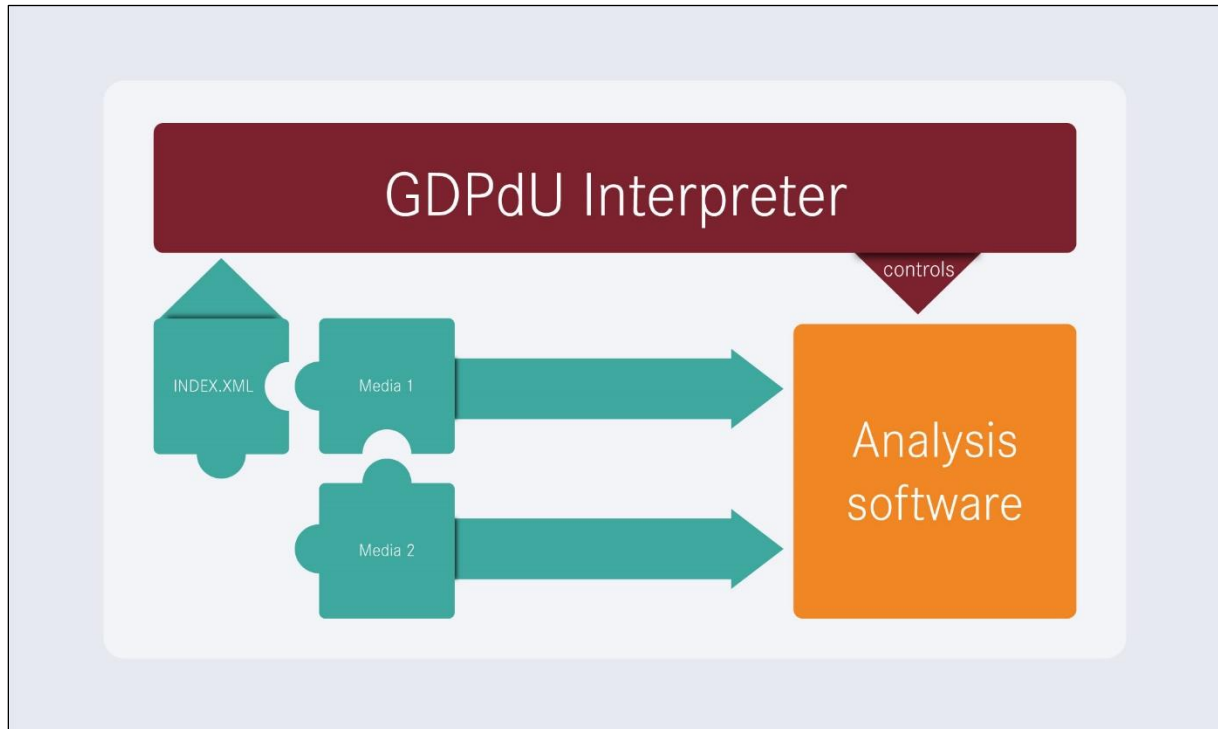


Figure 4: Import process

The XML DTD

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Versions available:
1.1 (August-01-2002)
-->
<!-- Start Simple Types -->
<!-- Supplementary Vocabulary -->
<ELEMENT Version (#PCDATA)>
<ELEMENT Location (#PCDATA)>
<ELEMENT Comment (#PCDATA)>
<ELEMENT Length (#PCDATA)>
<ELEMENT References (#PCDATA)>
<ELEMENT From (#PCDATA)>
<ELEMENT To (#PCDATA)>
<!-- Specifying a maximum length for a VariableLength column can
reduce a VariableLength tables' import time. If MaxLength
is not specified then we parse URL to determine the MaxLength
for each column.

* Only applies to VariableLength tables. -->
<ELEMENT MaxLength (#PCDATA)>
<!-- Specifies which character (if any) encapsulates a
VariableLength AlphaNumeric column.

Doublequote is the default TextEncapsulator "

* Only applies to VariableLength tables. (Optional) -->
<ELEMENT TextEncapsulator (#PCDATA)>
<!-- Specifies how many    digits appear to the right of the decimal symbol.

CAUTION: Results are undefined when importing numeric data with
greater Accuracy than the Accuracy defined in index.xml

For example trying to import the value 1000,25 with an
accuracy of 0 might result in 1000 or an error. This
behavior is specific to the implementation.

Zero is the default Accuracy '0' (Optional)
-->
<ELEMENT Accuracy (#PCDATA)>
<!-- The decimal place is not always stored with numbers. If each number
is supposed to have decimal places use ImpliedAccuracy -->
<ELEMENT ImpliedAccuracy (#PCDATA)>
<!-- Enables you to change how GDPdU displays dates.
DD.MM.YYYY is the default Format -->
<ELEMENT Format (#PCDATA)>
<!-- Specifies the symbol that indicates decimal values.
Comma is the default DecimalSymbol. ','
Specified once per Table. -->
<ELEMENT DecimalSymbol (#PCDATA)>
<!-- Specifies the symbol that groups the digits in large numbers.
Dot is the default DigitGroupingSymbol or ThousandsSeparator. '.'
Specified once per Table -->
<ELEMENT DigitGroupingSymbol (#PCDATA)>
<!-- Command(s) are executed in the following manner
* before the import process
* after the import process
* before a Media is imported
* after a Media is imported
-->
<ELEMENT Command (#PCDATA)>
<!-- Only the file protocol is supported at this time.

* The standard uses relative URLs.

Absolute URLs are not allowed. The following are all invalid:
* http://www.somewhere.com/data/Accounts.dat
* ftp://ftp.somewhere.com/data/Accounts.dat
* file://localhost/Accounts.dat
* file:///Accounts.dat
```

```

The following are valid examples
* Accounts.dat
* data/Accounts.dat
* data/january/Accounts.dat
* ../Accounts.dat

-->
<!ELEMENT URL (#PCDATA)>
<!-- Textual description of specified element (Optional) -->
<!ELEMENT Description (#PCDATA)>
<!-- The logical name of specified element.
Sometimes referred to business name.

If missing, URL will be used in place of Name. -->
<!ELEMENT Name (#PCDATA)>
<!-- Y2K Window Any year before Epoch is 2000+
Default value 30. -->
<!ELEMENT Epoch (#PCDATA)>
<!-- Element(s) that separate columns or records.
Semicolon is the default ColumnDelimiter. ';'
CRLF or &#13;&#10; is the default RecordDelimiter. -->
<!ELEMENT ColumnDelimiter (#PCDATA)>
<!ELEMENT RecordDelimiter (#PCDATA)>
<!-- The number of bytes skipped before reading of URL commences.
Zero is the default when not specified. '0'

-->
<!ELEMENT SkipNumBytes (#PCDATA)>
<!-- End Simple Types -->
<!-- Start Complex Types -->
<!-- Self-explanatory -->
<!ELEMENT Range (From, (To | Length)?)>
<!ELEMENT FixedRange (From, (To | Length))>
<!-- The document element -->
<!ELEMENT DataSet (Version, DataSupplier?, Command*, Media+, Command*)>
<!-- Supported datatypes (mandatory) -->
<!ELEMENT AlphaNumeric EMPTY>
<!ELEMENT Date (Format?)>
<!ELEMENT Numeric ((ImpliedAccuracy | Accuracy)?)>
<!-- Supported codepages:
Be careful to explicitly define RecordDelimiter when using
a non-default codepage.

ANSI is the default codepage when not specified -->
<!ELEMENT ANSI EMPTY>
<!ELEMENT Macintosh EMPTY>
<!ELEMENT OEM EMPTY>
<!ELEMENT UTF16 EMPTY>
<!ELEMENT UTF7 EMPTY>
<!ELEMENT UTF8 EMPTY>
<!-- Supported file formats:
FixedLength
VariableLength -->
<!ELEMENT FixedLength ((Length | RecordDelimiter)?, ((FixedPrimaryKey+, FixedColumn*) | (FixedColumn+)),
ForeignKey*)>
<!ELEMENT FixedColumn (Name, Description?, (Numeric | AlphaNumeric | Date), Map*, FixedRange)>
<!ELEMENT FixedPrimaryKey (Name, Description?, (Numeric | AlphaNumeric | Date), Map*, FixedRange)>
<!ELEMENT VariableLength (ColumnDelimiter?, RecordDelimiter?, TextEncapsulator?, ((VariablePrimaryKey+,
VariableColumn*) | (VariableColumn+)), ForeignKey*)>
<!ELEMENT VariableColumn (Name, Description?, (Numeric | (AlphaNumeric, MaxLength?) | Date), Map*)>
<!ELEMENT VariablePrimaryKey (Name, Description?, (Numeric | (AlphaNumeric, MaxLength?) | Date), Map*)>
<!-- Description of the entity supplying the data. (Optional) -->
<!ELEMENT DataSupplier (Name, Location, Comment)>
<!-- The first Media will contain index.xml. Importing will process each media listed -->
<!ELEMENT Media (Name, Command*, Table+, Command*)>
<!-- Elements common to FixedLength & VariableLength are propagated to Table. -->
<!ELEMENT Table (URL, Name?, Description?, Validity?, (ANSI | Macintosh | OEM | UTF16 | UTF7 | UTF8)?,
(DecimalSymbol, DigitGroupingSymbol)?, SkipNumBytes?, Range?, Epoch?, (VariableLength | FixedLength))>
<!-- ForeignKeys denote joins or relationships between tables.
To successfully join two tables make sure both the PrimaryKey
and the referenced column (foreignkey) are of the same datatype.
Results are undefined when joining two tables with different
key datatypes. Most likely an error will occur -->
<!ELEMENT ForeignKey (Name+, References)>
<!-- Maps AlphaNumeric columns from 'From' to 'To'
i.e. From To
=====

```

```
True    1
True    -1
False   0
```

Basically, a map is an associative container.

The standard implementation only supports
AlphaNumeric datatypes. The following
conversions are NOT supported.

```
Numeric    -> AlphaNumeric
Date       -> AlphaNumeric
AlphaNumeric -> Date
AlphaNumeric -> Numeric
```

-->

<!-- ELEMENT Map (Description?, From, To)>

<!-- Documentation for table validity. -->

<!-- ELEMENT Validity (Range, Format?)>

<!-- End Complex Types -->

Figure 5: XML DTD

Organization of the default data structure

XML

version1.0

encodingUTF-8

DOCTYPEDataSet

IDSYSTEM"gdpu-01-08-2002.dtd"

DataSet

Version1.0

DataSupplier

NameGlaswerk AG

LocationSingen/Deutschland

CommentDatenträgerüberlassung nach GDPdU vom 14.August 2002, Dr. Grässle, Tel: 07544 934266

Media (2)

NameTable

1CD Nummer 1

Table

Table

2CD Nummer 2

Table

Table

Table

Table

Table

URLbestellungen.csv

NameBestellungen

DescriptionBestellungen-Bewegungsdaten

VariableLength

VariableColumn (6)

NameAlphaNumericDateNumeric

1Artikel-Nr

2Kunden-Code

3Bestelldatum

Date

FormatYYYYMMDD

4Lieferdatum

Date

FormatYYYYMMDD

5Bestelldatum

Date

FormatYYYYMMDD

6Frachtkosten

Comment

weitere columns...

ForeignKey (2)

NameReferences

1Kunden-CodeKunden

2Artikel-IDArtikel

Figure 6: Default data structure – organization

The basic element of the default data structure is the DataSet element. This contains the elements of Version, DataSupplier, Command and Media.

- Version contains the version number of the data media provision.
- DataSupplier contains details concerning the origin of data.
- The media containers contain details concerning the provided data media and tables stored on these. Here, for each data medium provided, an individual media entry is carried out. It is therefore also possible to combine a number of data media within a DataSet.

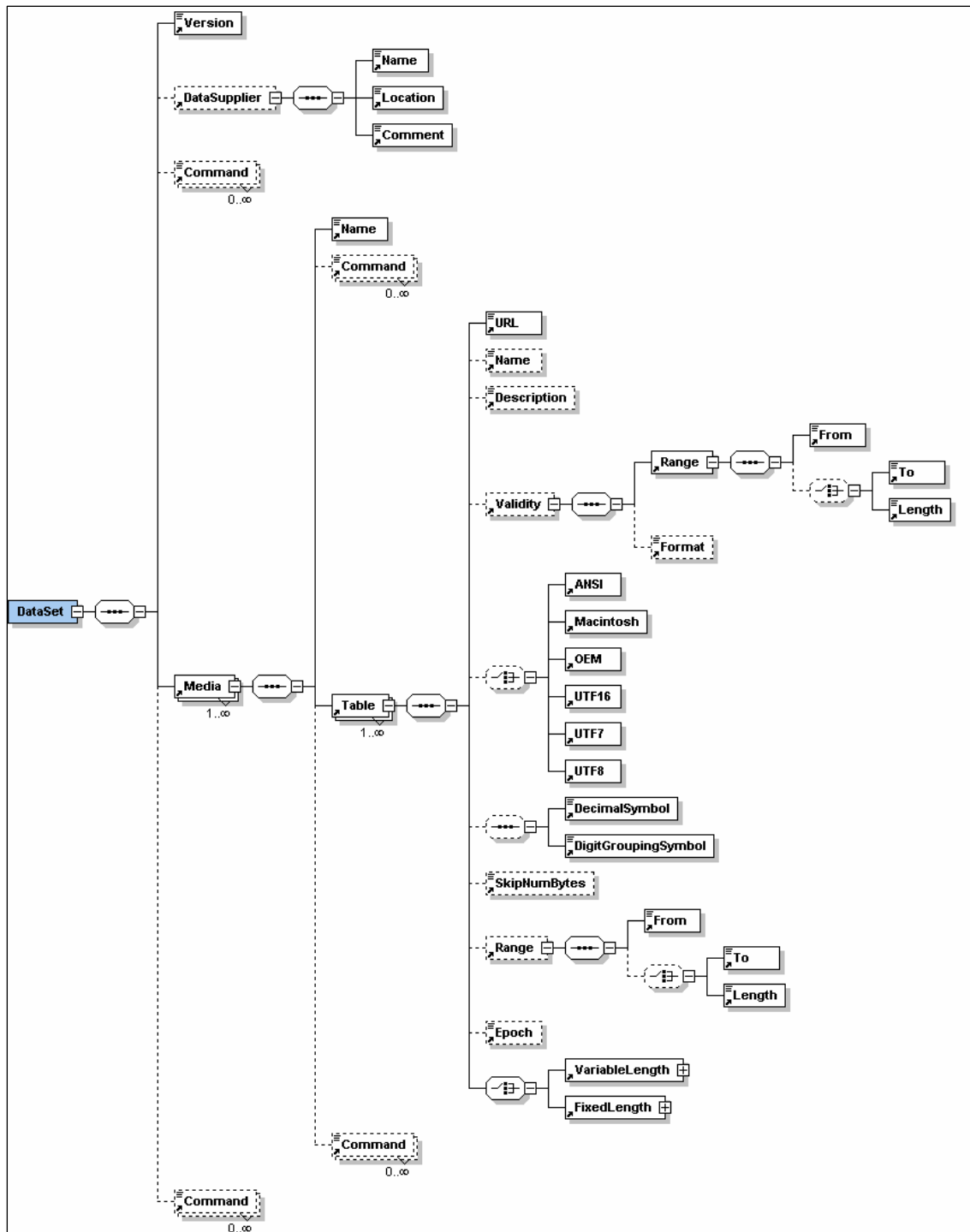


Figure 7: Default data structure – DataSet

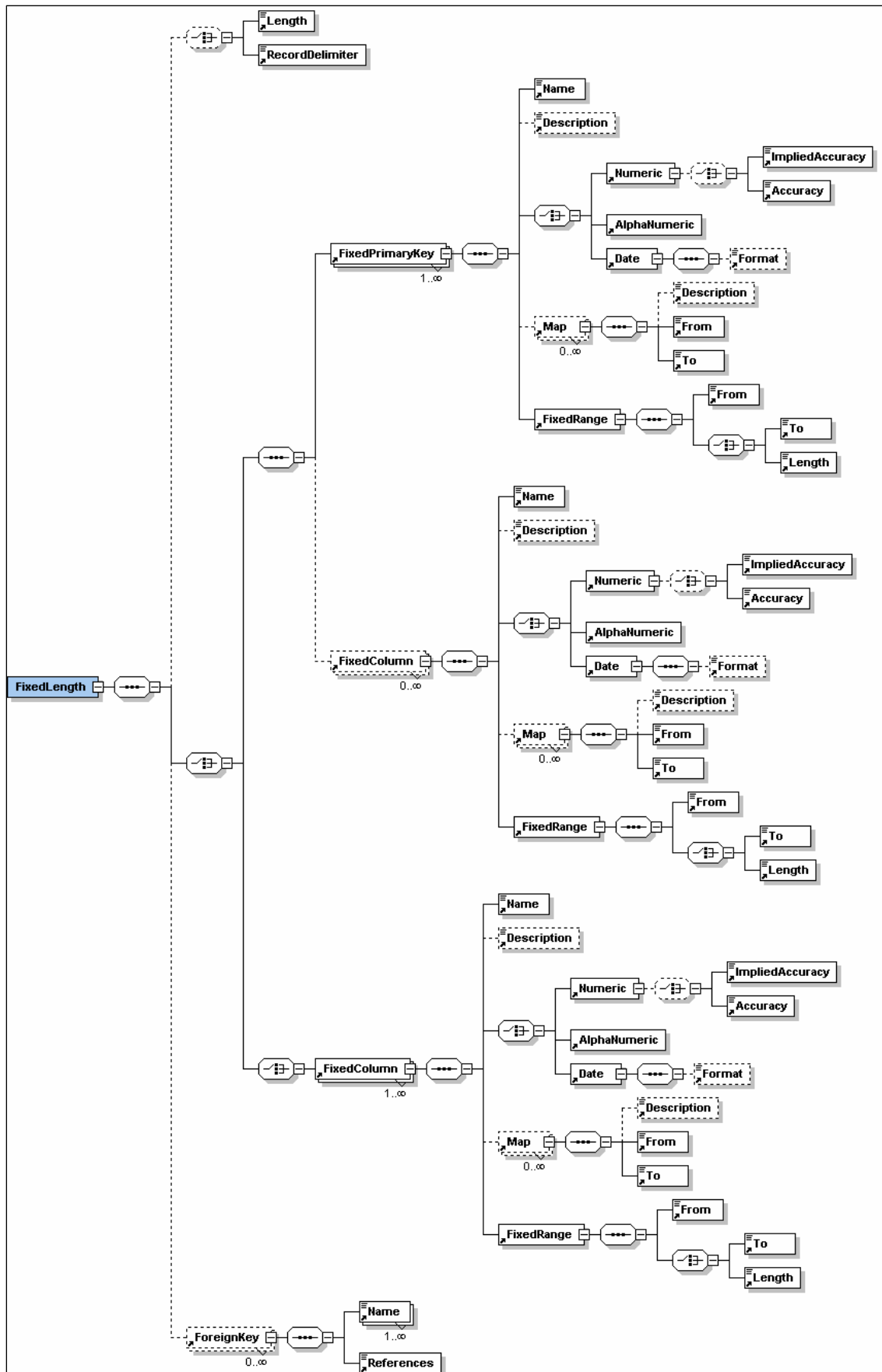


Figure 8: Default data structure – FixedLength

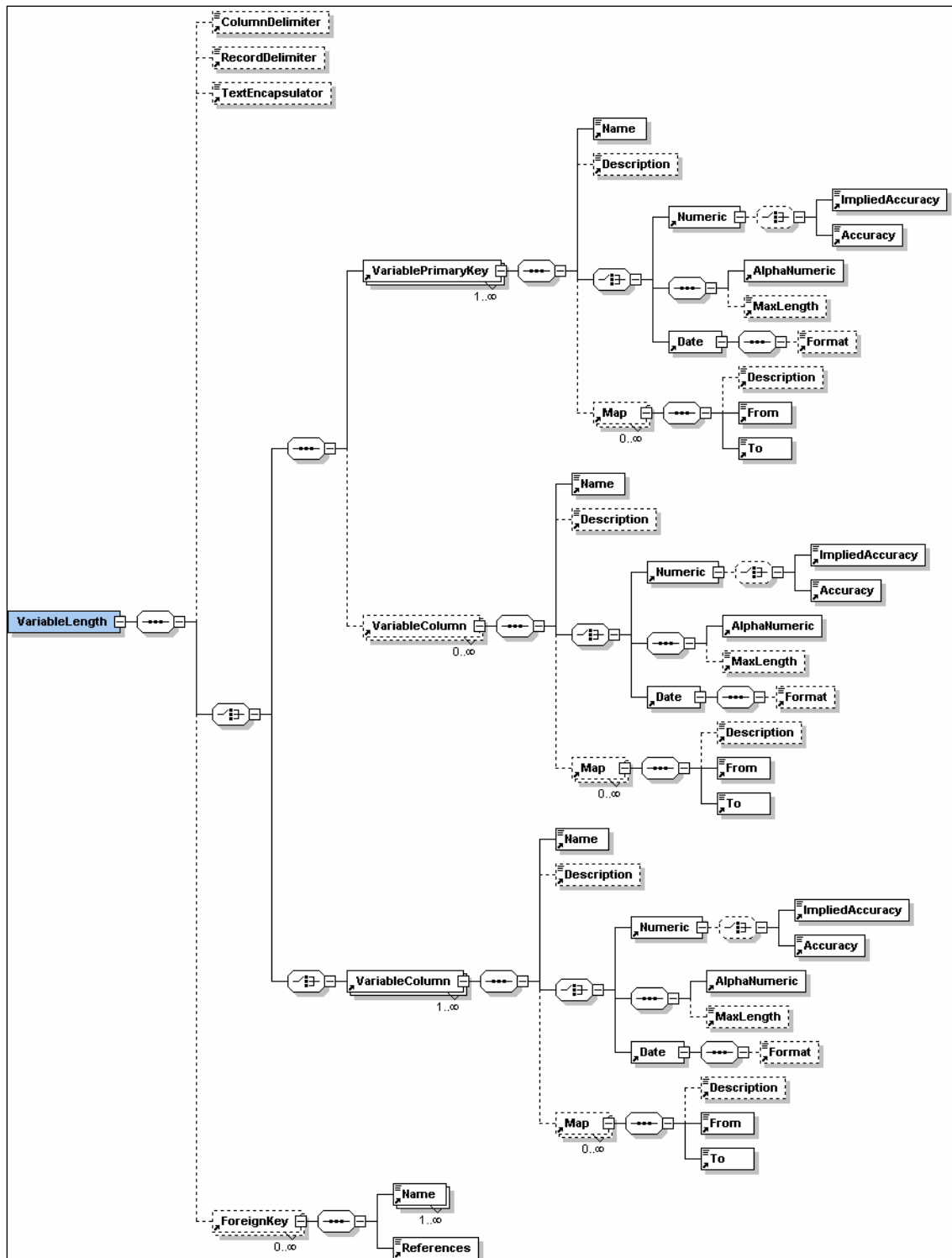


Figure 9: Default data structure – VariableLength

Notes regarding XML

- Note on operators common in XML:

Operator	Element type may occur
?	0 times or 1 time
*	0 times or as often as desired
+	1 time or as often as desired

Table 3: Overview of common operators

- #PCDATA stands for parsed character data and means that the element is provided for a text entry.
- In the case of enumerations, the possible values are separated by means of vertical lines ("|").
- XML is case sensitive.
- Please do not use any of the following characters in the descriptive fields and/or the table or column names: " & < >

Instead use names or numbers from the list below:

Character	Name	Number
"	"	"
&	&	&
<	<	<
>	>	>

Table 4: List of names and numbers for the characters " & < >

Description of elements

What follows is a description of the individual ELEMENT declarations.

What generally applies is that the texts stored in the relevant “description” attributes are displayed to the auditor as commentary. The length of the text in the description attributes should not exceed 255 characters.

<!ELEMENT Version (#PCDATA)>

Contains the version number of the data media provision. This element has no technical effect but serves as a description.

Used in container	Context-dependent description
DataSet	

Table 5: ELEMENT Version

<!ELEMENT Location (#PCDATA)>

Describes the location of the data supplier.

Used in container	Context-dependent description
DataSupplier	

Table 6: ELEMENT Location

<!ELEMENT Comment (#PCDATA)>

Comment field for additional information concerning the data supplier.

Used in container	Context-dependent description
DataSupplier	

Table 7: ELEMENT Comment

<!ELEMENT Length (#PCDATA)>

Contains information concerning the length of attributes or datasets.

Used in container	Context-dependent description
Range	

Table 8: ELEMENT Length

<!ELEMENT References (#PCDATA)>

Contains information concerning links.

Used in container	Context-dependent description
ForeignKey	referenced table

Table 9: ELEMENT References

In this regard also see example on page 38.

<!ELEMENT From (#PCDATA)>

Contains information concerning the starting value of a value range.

Used in container	Context-dependent description
Range	Starting value within a value range
Map	Redefinition of data

Table 10: ELEMENT From

<!ELEMENT To (#PCDATA)>

Contains information concerning the end value of a value range.

Used in container	Context-dependent description
Range	End value within a value range
Map	Redefinition of data

Table 11: ELEMENT To

<!ELEMENT MaxLength (#PCDATA)>

Contains information concerning the maximum length of an alpha-numeric field in a VariableLength table.

Used in container	Context-dependent description
VariableColumn	Maximum length of the field
VariablePrimaryKey	Maximum length of the field

Table 12: ELEMENT MaxLength

The specification of MaxLength for the VariableLength file type speeds up the import process, because otherwise the file would be tested before the import process and the maximum lengths of fields would be determined.

<!ELEMENT TextEncapsulator (#PCDATA)>

In the case of VariableLength files, it is possible to encapsulate text fields using an “Encapsulator” character, e.g. in case the field separator occurs in the data.

Standard value: "

Used in container	Context-dependent description
VariableLength	

Table 13: ELEMENT TextEncapsulator

<!ELEMENT Accuracy (#PCDATA)>

Number of decimal places. Be careful with data the accuracy of which is more precise than specified under Accuracy.

Standard value: 0

Used in container	Context-dependent description
Numeric	

Table 14: ELEMENT Accuracy

<!ELEMENT ImpliedAccuracy (#PCDATA)>

Number of decimal places, even when no decimal places are specified in the source data.

Example:

Source data: “100; 200; 102; 5000; 6587890”

ImpliedAccuracy 3: “0,100; 0,200; 0,102; 5,000; 6587,890”

Used in container	Context-dependent description
Numeric	

Table 15: ELEMENT ImpliedAccuracy

<IELEMENT Format (#PCDATA)>

Details concerning format definitions.

Standard value: DD.MM.YYYY

Used in container	Context-dependent description
Date	<p>This is where the mask for the representation of the date details is stored.</p> <p>The following specifications for the format are valid:</p> <ul style="list-style-type: none"> • DD for day, • MM for month, • YY or alternatively YYYY for year. <p>Example (30 September 2001)</p> <ul style="list-style-type: none"> • MM/DD/YY: 09/30/01 • DD.MM.YY: 30.09.01 • MM/DD/YYYY: 09/30/2001 • YYYY-MM-DD: 2001-09-30 • DDMMYY: 300901 • DDMMYYYY: 30092001
Validity	Masking for the date format used (see Date)

Table 16: ELEMENT Format

<IELEMENT DecimalSymbol (#PCDATA)>

Specification concerning the decimal place separator

Standard value: ,

Used in container	Context-dependent description
Table	

Table 17: ELEMENT DecimalSymbol

<IELEMENT DigitGroupingSymbol (#PCDATA)>

Defines the separator for thousands.

Standard value: .

Used in container	Context-dependent description
Table	

Table 18: ELEMENT DigitGroupingSymbol

<!ELEMENT Command (#PCDATA)>

Command defines an operating system command. These can be carried out at the following points:

- prior to the overall import process
- after the overall import process
- prior to an individual data medium being imported
- after an individual data medium being imported

Used in container	Context-dependent description
DataSet	Command for the overall process
Media	Command for an individual data medium

Table 19: ELEMENT Command

In this regard also see example on page 40.

<!ELEMENT URL (#PCDATA)>

This element contains the physical file name of the file provided.

In this version of the default data structure, relative URLs are supported exclusively. The path to the index.xml file serves as the base directory.

Examples of **invalid** entries:

<http://www.somewhere.com/data/Accounts.csv>

<ftp://ftp.somewhere.com/data/Accounts.csv>

Examples of **valid** entries:

Accounts.dat

data/Accounts.dat

data/january/Accounts.dat

../Accounts.dat

Used in container	Context-dependent description
Table	

Table 20: ELEMENT URL

<IELEMENT Description (#PCDATA)>

Description field

Used in container	Context-dependent description
FixedColumn	Description field for columns in FixedLength files
FixedPrimaryKey	Description field for keys in FixedLength files
VariableColumn	Description field for columns in VariableLength files
VariablePrimaryKey	Description field for keys in VariableLength files
Table	Description field for tables
Map	Description field for redefinition of data

Table 21: ELEMENT Description

<IELEMENT Name (#PCDATA)>

Description field

Standard value: Specification from the URL element

Used in container	Context-dependent description
FixedColumn	Column name in FixedLength files
FixedPrimaryKey	Key name in FixedLength files
VariableColumn	Column name in VariableLength files
VariablePrimaryKey	Key name in VariableLength files
DataSupplier	Name of the data supplier (company name, ...)
Media	Data media name
URL	Physical table name
ForeignKey	Name of the foreign key

Table 22: ELEMENT Name

<!ELEMENT Epoch (#PCDATA)>

For dates with a two-digit year format, it is possible to specify a standard placeholder that separates 19xx from 20xx. This value is pre-populated with 30.

In some files, the date 25 November 55 can mean either 1955 or 2055. Using Epoch, it is specified up to which two-digit value the year is to be interpreted as referring to the next century. For instance, if the century separation value field contains the value 50, all year indications from 0 to 49 will be interpreted as the years 2000 to 2049, and the year indications 50 to 99 as 1950 to 1999.

Standard value: 30

Used in container	Context-dependent description
Table	

Table 23: ELEMENT Epoch

<!ELEMENT ColumnDelimiter(#PCDATA)>

Defines the separator for data fields.

Standard value: ; (semicolon “;”)

Used in container	Context-dependent description
VariableLength	Data field separator

Table 24: ELEMENT ColumnDelimiter

Example of conventional separators:

Separator	XML notation
Comma “,”	,
Semicolon “;”	;
Tab		

Table 25: ELEMENT ColumnDelimiter – list of common separators

<IELEMENT RecordDelimiter (#PCDATA)>

Defines separators for datasets.

Standard value: 
 (CRLF)

Used in container	Context-dependent description
FixedLength	Dataset separator
VariableLength	Dataset separator

Table 26: ELEMENT RecordDelimiter

Example of conventional separators:

Separator	XML notation
CR Carriage Return	
CRLF	

LF Linefeed	

Table 27: ELEMENT RecordDelimiter – list of common separators

<IELEMENT SkipNumBytes (#PCDATA)>

If a file contains undesirable data at the beginning, e.g. file header, it needs to be specified here how many bytes are to be skipped and not read.

Standard value: 0

Used in container	Context-dependent description
Table	

Table 28: ELEMENT SkipNumBytes

<IELEMENT Range (From, (To | Length)?)>

Defines a range or a starting value.

Starting value means: if no To or Length is specified, this range only defines a starting value (e.g. a dataset position).

Used in container	Context-dependent description
Table	Start dataset
Validity	Validity range

Table 29: ELEMENT Range

<!ELEMENT FixedRange (From, (To | Length))>

Defines a range specifying From and To or From and Length.

Used in container	Context-dependent description
FixedColumn	Field position in FixedLength files
FixedPrimaryKey	Key position in FixedLength files

Table 30: ELEMENT FixedRange

<!ELEMENT DataSet (Version, DataSupplier?, Command*, Media+, Command*)>

The DataSet is the upper element (document element) in the hierarchy of the XML document. The element DataSet is the medium for the version, data origin, preceding and subsequent processes and the media containing the tables.

<!ELEMENT AlphaNumeric EMPTY>

Defines the data type as alpha-numeric.

Used in container	Context-dependent description
FixedColumn	Type of field in FixedLength file
FixedPrimaryKey	Type of key in FixedLength file
VariableColumn	Type of field in VariableLength files
VariablePrimaryKey	Type of key in VariableLength files

Table 31: ELEMENT AlphaNumeric

<!ELEMENT Date (Format?)>

Defines the data type as date.

Used in container	Context-dependent description
FixedColumn	Type of field in FixedLength file
FixedPrimaryKey	Type of key in FixedLength file
VariableColumn	Type of field in VariableLength files
VariablePrimaryKey	Type of key in VariableLength files

Table 32: ELEMENT Date

<!ELEMENT Numeric ((ImpliedAccuracy | Accuracy)?)>

Defines the data type as numeric.

Used in container	Context-dependent description
FixedColumn	Type of field in FixedLength file
FixedPrimaryKey	Type of key in FixedLength file
VariableColumn	Type of field in VariableLength files
VariablePrimaryKey	Type of key in VariableLength files

Table 33: ELEMENT Numeric

<!ELEMENT ANSI EMPTY>

Specifies the use of the ANSI code page.

<!ELEMENT Macintosh EMPTY>

Specifies the use of the Macintosh code page.

<!ELEMENT OEM EMPTY>

Specifies the use of the IBM-PC-ASCII code page.

<!ELEMENT UTF16 EMPTY>

Specifies the use of the UTF16 code page.

<!ELEMENT UTF7 EMPTY>

Specifies the use of the UTF7 code page.

<IELEMENT UTF8 EMPTY>

Specifies the use of the UTF8 code page.

Standard value: ANSI

Platform	Character set
Unix computers	7-bit ASCII character set (7-bit ISO code, US variant), only characters with ASCII code below 128, in other words letters, numbers and special characters (full-stop, comma, etc.) without country-specific special characters (umlauts, etc.) and without graphics characters. Line break is LF (Line Feed), character 10.
PC under DOS	8-bit "IBM PC ASCII" character set that in terms of the lower 127 characters corresponds to 7-bit ASCII and from character 128 contains country-specific special characters and (semi-)graphics characters. Line break is the sequence CR and LF, representing characters 12 and 10 (Carriage Return, Line Feed).
PC under Windows	8-bit ANSI character set, which up to 127 corresponds to ASCII and above 127 contains country-specific special characters that do not correspond to those of IBM PC ASCII. Line breaks, as under DOS, the sequence CR, LF. (Special applications are also able to use a 16-bit code, Unicode.)
Apple Mac	Mac-specific "Mac-ASCII" 8-bit character set, which up to 127 corresponds to ASCII and above 127 contains country-specific special characters. Line break is CR.

Table 34: ELEMENT UTF8

The selection of code page does not set a default for the RecordDelimiter. You will always have to specify the RecordDelimiter explicitly if the standard value (CRLF) is not suitable.

<!ELEMENT FixedLength ((Length | RecordDelimiter)?, ((FixedPrimaryKey+, FixedColumn*) | (FixedColumn+)), ForeignKey*)>

Defines a FixedLength type file.

Used in container	Context-dependent description
Table	

Table 35: ELEMENT FixedLength

<!ELEMENT FixedColumn (Name, Description?, (Numeric | AlphaNumeric | Date), Map*, FixedRange)>

Defines a column (=column) in a FixedLength type file.

Used in container	Context-dependent description
FixedLength	

Table 36: ELEMENT FixedColumn

<!ELEMENT FixedPrimaryKey (Name, Description?, (Numeric | AlphaNumeric | Date), Map*, Range)>

Defines a key field (=key) in a FixedLength type file.

Used in container	Context-dependent description
FixedLength	

Table 37: ELEMENT FixedPrimaryKey

In this regard also see example on page 38.

<!ELEMENT VariableLength (ColumnDelimiter?, RecordDelimiter?, TextEncapsulator?, ((VariablePrimaryKey+, VariableColumn*) | (VariableColumn+))?, ForeignKey*)>

Defines a VariableLength type file.

Used in container	Context-dependent description
Table	

Table 38: ELEMENT VariableLength

The fields (VariableColumn, VariablePrimaryKey) must be defined in the sequence in which they are delivered in the file.

<!ELEMENT VariableColumn (Name, Description?, (Numeric | AlphaNumeric, MaxLength ? | Date), Map*)>

Defines a column (=column) in a VariableLength type file.

Used in container	Context-dependent description
VariableLength	

Table 39: ELEMENT VariableColumn

The fields (VariableColumn, VariablePrimaryKey) must be defined in the sequence in which they are delivered in the file.

<!ELEMENT VariablePrimaryKey (Name, Description?, (Numeric | AlphaNumeric, MaxLength ? | Date), Map*)>

Defines a key field (=key) in a VariableLength type file.

Used in container	Context-dependent description
VariableLength	

Table 40: ELEMENT VariablePrimaryKey

The fields (VariableColumn, VariablePrimaryKey) must be defined in the sequence in which they are delivered in the file.

Concerning links also see example on page 38.

<!ELEMENT DataSupplier (Name, Location, Comment)>

Contains details concerning the data supplier.

Used in container	Context-dependent description
DataSet	supplier (company, ...)

Table 41: ELEMENT DataSupplier

<!ELEMENT Media (Name, Command*, Table+, Command*)>

Defines the content of a data medium.

Used in container	Context-dependent description
DataSet	data media transfer

Table 42: ELEMENT Media

<!ELEMENT Table (URL, Name?, Description?, Validity?, (ANSI | Macintosh | OEM | UTF16 | UTF7 | UTF8)?, (DecimalSymbol, DigitGroupingSymbol)?, SkipNumBytes?, Range?, (VariableLength | FixedLength)?)>

Defines a table.

Used in container	
Media	table

Table 43: ELEMENT Table

<!ELEMENT ForeignKey (Name+, References)>

Defines a column as a foreign key.

The column referenced by name must be defined as a FixedColumn or VariableColumn in the same table.

Used in container	Context-dependent description
FixedLength	Field in FixedLength file, referenced table
VariableLength	Field in VariableLength file, referenced table

Table 44: ELEMENT ForeignKey

In this regard also see example on page 38.

<!ELEMENT Map (Description?, From, To)>

Redefinitions at the data level can be carried out by means of the Map element.

Example:

From (original data)	To (target data)
1	true
0	false
Hr	Herr (Mister)
Fr	Frau (Mrs/Miss)
01	married
02	single
03	tax category 3

Table 45: ELEMENT Map – redefinitions

Used in container	Context-dependent description
FixedColumn	Field in FixedLength file
FixedPrimaryKey	Key in FixedLength file
VariableColumn	Field in VariableLength files
VariablePrimaryKey	Key in VariableLength files

Table 46: ELEMENT Map

Map may only be used for alpha-numeric fields.

You should avoid delivering coded errors as part of the data carrier cession and instead convert all codes into textual descriptions. In the event that this conversion is not already carried out by the export program, Map represents a possibility for converting codes using data description.

<!ELEMENT Validity (Range, Format?)>

Defines the validity period for the data.

Used in container	Context-dependent description
Table	Validity period

Table 47: ELEMENT Validity

Example 1 of an index.xml file

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE DataSet SYSTEM "gdpdu-01-08-2002.dtd">
<DataSet>
  <Version>1.0</Version>
  <DataSupplier>
    <Name>Glaswerk AG</Name>
    <Location>Singen/Deutschland</Location>
    <Comment>Datentraegerüberlassung nach GDPdU vom 14.August 2002, Dr. Grässle, Tel: 07544
934266</Comment>
  </DataSupplier>
  <Media>
    <Name>CD Nummer 1</Name>
    <Table>
      <URL>Account.csv</URL>
      <Name>Account</Name>
      <Description>Accounts Receivables</Description>
      <!-- Diese Tabelle gilt von 1.Januar 2000- 31.Dezember 2002 -->
      <Validity>
        <Range>
          <From>20000101</From>
          <To>20021231</To>
        </Range>
        <Format>YYYYMMDD</Format>
      </Validity>
      <!-- Spezifiziert den Dezimaltrenner -->
      <DecimalSymbol>.</DecimalSymbol>
      <!-- Spezifiziert den Tausender-Separator -->
      <DigitGroupingSymbol>,</DigitGroupingSymbol>
      <VariableLength>
        <!-- ein zusammengesetzter Primaerschlüssel -->
        <VariablePrimaryKey>
          <Name>RegionId</Name>
          <AlphaNumeric/>
        </VariablePrimaryKey>
        <VariablePrimaryKey>
          <Name>Id</Name>
          <AlphaNumeric/>
        </VariablePrimaryKey>
        <VariableColumn>
          <Name>Account Description</Name>
          <AlphaNumeric/>
        </VariableColumn>
        <VariableColumn>
          <Name>Balance</Name>
          <Numeric/>
        </VariableColumn>
        <VariableColumn>
          <Name>Old Balance</Name>
          <Numeric/>
        </VariableColumn>
      </VariableLength>
    </Table>
    <Table>
      <URL>Region.csv</URL>
      <VariableLength>
        <VariablePrimaryKey>
          <Name>RegionId</Name>
          <AlphaNumeric/>
        </VariablePrimaryKey>
        <VariableColumn>
          <Name>Description</Name>
          <AlphaNumeric/>
        </VariableColumn>
      </VariableLength>
    </Table>
  </Media>
  <Media>
    <Name>CD Nummer 2</Name>
    <Table>
      <URL>Sales.csv</URL>
```

```

    <Range>
      <From>56</From>
    </Range>
  </FixedLength>
  <FixedLength>
    <FixedPrimaryKey>
      <Name>SalesId</Name>
      <AlphaNumeric/>
      <FixedRange>
        <!-- Position des Feldes: -->
        <From>1</From>
        <To>10</To>
      </FixedRange>
    </FixedPrimaryKey>
    <FixedColumn>
      <Name>RegionId</Name>
      <AlphaNumeric/>
      <FixedRange>
        <From>11</From>
        <Length>10</Length>
      </FixedRange>
    </FixedColumn>
    <FixedColumn>
      <Name>Id</Name>
      <AlphaNumeric/>
      <FixedRange>
        <From>21</From>
        <To>30</To>
      </FixedRange>
    </FixedColumn>
    <FixedColumn>
      <Name>SalesComplete</Name>
      <AlphaNumeric/>
      <!-- Redefinition des Wertes "1" nach "True" und "0" nach "False" -->
      <Map>
        <From>1</From>
        <To>True</To>
      </Map>
      <Map>
        <From>0</From>
        <To>False</To>
      </Map>
      <FixedRange>
        <From>31</From>
        <To>32</To>
      </FixedRange>
    </FixedColumn>
    <!-- ein zusammengesetzter Fremdschlüssel der den Primaerschlüssel in Tabelle Account
referenziert -->
    <ForeignKey>
      <Name>RegionId</Name>
      <Name>Id</Name>
      <References>Account</References>
    </ForeignKey>
  </FixedLength>
</Table>
<Table>
  <URL>kunden.csv</URL>
  <Name>Kunden</Name>
  <Description>Kunden-Stammdaten</Description>
  <VariableLength>
    <VariablePrimaryKey>
      <Name>Kunden-Code</Name>
      <AlphaNumeric/>
    </VariablePrimaryKey>
    <VariableColumn>
      <Name>Firma</Name>
      <AlphaNumeric/>
    </VariableColumn>
    <VariableColumn>
      <Name>Kontaktperson</Name>
      <AlphaNumeric/>
    </VariableColumn>
    <!-- weitere columns... -->
  </VariableLength>

```

```

</Table>
<Table>
  <URL>artikel.csv</URL>
  <Name>Artikel</Name>
  <Description>Artikel-Stammdaten</Description>
  <VariableLength>
    <VariablePrimaryKey>
      <Name>Artikel-Nr</Name>
      <AlphaNumeric/>
    </VariablePrimaryKey>
    <VariableColumn>
      <Name>Artikelname</Name>
      <AlphaNumeric/>
    </VariableColumn>
    <VariableColumn>
      <Name>Lieferanten-Nr</Name>
      <AlphaNumeric/>
    </VariableColumn>
    <!-- weitere columns... -->
  </VariableLength>
</Table>
<Table>
  <URL>bestellungen.csv</URL>
  <Name>Bestellungen</Name>
  <Description>Bestellungen-Bewegungsdaten</Description>
  <VariableLength>
    <VariableColumn>
      <Name>Artikel-Nr</Name>
      <AlphaNumeric/>
    </VariableColumn>
    <VariableColumn>
      <Name>Kunden-Code</Name>
      <AlphaNumeric/>
    </VariableColumn>
    <VariableColumn>
      <Name>Bestelldatum</Name>
      <Date>
        <Format>YYYYMMDD</Format>
      </Date>
    </VariableColumn>
    <VariableColumn>
      <Name>Lieferdatum</Name>
      <Date>
        <Format>YYYYMMDD</Format>
      </Date>
    </VariableColumn>
    <VariableColumn>
      <Name>Bestelldatum</Name>
      <Date>
        <Format>YYYYMMDD</Format>
      </Date>
    </VariableColumn>
    <VariableColumn>
      <Name>Frachtkosten</Name>
      <Numeric/>
    </VariableColumn>
    <!-- weitere columns... -->
    <ForeignKey>
      <Name>Kunden-Code</Name>
      <References>Kunden</References>
    </ForeignKey>
    <ForeignKey>
      <Name>Artikel-ID</Name>
      <References>Artikel</References>
    </ForeignKey>
  </VariableLength>
</Table>
</Media>
</DataSet>

```

Figure 10: Example 1 – index.xml file

Example 2 regarding links

It may make sense in terms of data carrier cession not to deliver transaction data and master data in one broad table, but instead to save these into separate tables:

Example:

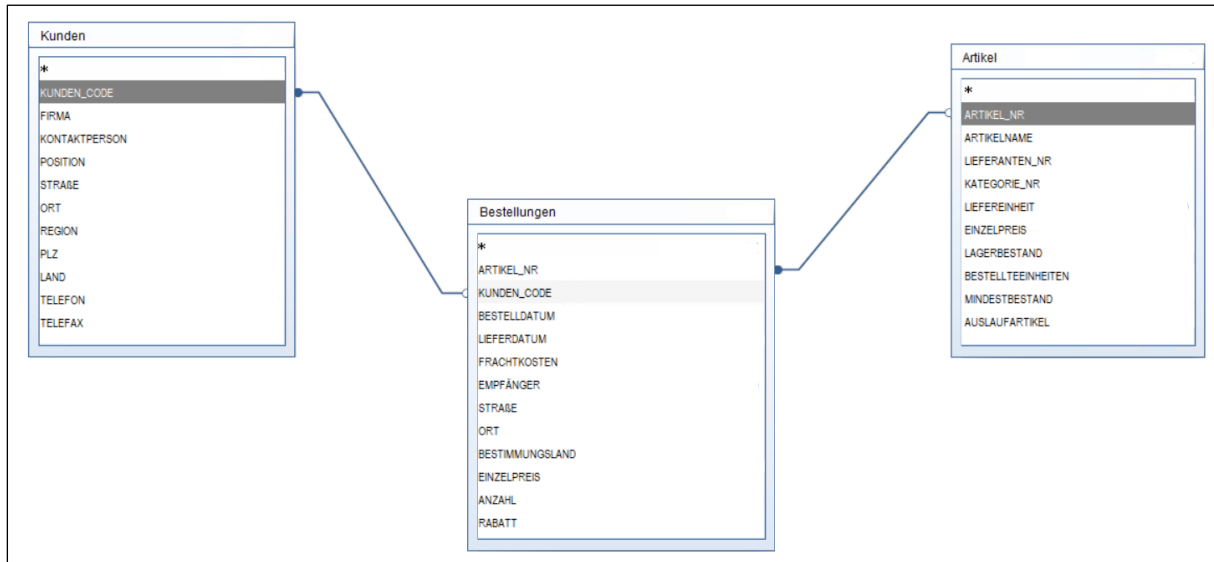


Figure 11: Example 2 – tables with transaction data and master data

The primary keys are described in the master data:

<pre> <Table> <URL>kunden.csv</URL> <Name>Kunden</Name> <Description>Kunden-Stammdaten</Description> <VariableLength> <VariablePrimaryKey> <Name>Kunden-Code</Name> <AlphaNumeric/> </VariablePrimaryKey> <VariableColumn> <Name>Firma</Name> <AlphaNumeric/> </VariableColumn> <VariableColumn> <Name>Kontaktperson</Name> <AlphaNumeric/> </VariableColumn> <!-- weitere columns... --> </VariableLength> </Table> </pre>	<pre> <Table> <URL>artikel.csv</URL> <Name>Artikel</Name> <Description>Artikel-Stammdaten</Description> <VariableLength> <VariablePrimaryKey> <Name>Artikel-Nr</Name> <AlphaNumeric/> </VariablePrimaryKey> <VariableColumn> <Name>Artikelname</Name> <AlphaNumeric/> </VariableColumn> <VariableColumn> <Name>Lieferanten-Nr</Name> <AlphaNumeric/> </VariableColumn> <!-- weitere columns... --> </VariableLength> </Table> </pre>
---	--

Figure 12: Example 2 – master data

The transaction data describe the links to the master data via the specification of the foreign key relationship:

```
<Table>
  <URL>bestellungen.csv</URL>
  <Name>Bestellungen</Name>
  <Description>Bestellungen-Bewegungsdaten</Description>
  <VariableLength>
    <VariableColumn>
      <Name>Artikel-Nr</Name>
      <AlphaNumeric/>
    </VariableColumn>
    <VariableColumn>
      <Name>Kunden-Code</Name>
      <AlphaNumeric/>
    </VariableColumn>
    <VariableColumn>
      <Name>Bestelldatum</Name>
      <Date>
        <Format>YYYYMMDD</Format>
      </Date>
    </VariableColumn>
    <VariableColumn>
      <Name>Lieferdatum</Name>
      <Date>
        <Format>YYYYMMDD</Format>
      </Date>
    </VariableColumn>
    <VariableColumn>
      <Name>Bestelldatum</Name>
      <Date>
        <Format>YYYYMMDD</Format>
      </Date>
    </VariableColumn>
    <VariableColumn>
      <Name>Frachtkosten</Name>
      <Numeric/>
    </VariableColumn>
    <!-- weitere columns... -->
  <ForeignKey>
    <Name>Kunden-Code</Name>
    <References>Kunden</References>
  </ForeignKey>
  <ForeignKey>
    <Name>Artikel-Nr</Name>
    <References>Artikel</References>
  </ForeignKey>
</VariableLength>
</Table>
```

Figure 13: Example 2 – transaction data

Note:

Although the description of a VariablePrimaryKey fully describes the column, the ForeignKey does not define a new column. The column used in the ForeignKey must already have been defined as a VariableColumn or FixedColumn in advance.

Example 3 of an index.xml file (compressed data)

index.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE DataSet SYSTEM "gdpdu-01-08-2002.dtd">
<DataSet>
  <Version>1.2</Version>
  <DataSupplier>
    <Name>Audicon GmbH</Name>
    <Location>Germany</Location>
    <Comment>A compressed sample DataSet</Comment>
  </DataSupplier>
  <Command>uncompress.bat</Command>
  <Media>
    <Name>Disk 1</Name>
    <Table>
      <URL>dummy.csv</URL>
      <VariableLength>
        <VariablePrimaryKey>
          <Name>dummy</Name>
          <Numeric/>
        </VariablePrimaryKey>
      </VariableLength>
    </Table>
  </Media>
</DataSet>
```

Figure 14: Example 3 – index.xml file

uncompress.bat

```
rem -----
rem uncompress.bat
rem expandiert gdpdu.zip in das windows-temp-verzeichnis, unterverzeichnis gdpdu
rem -----
mkdir %temp%\gdpdu
pkunzip gdpdu.zip %temp%\gdpdu
```

Figure 15: Example 3 – uncompress.bat file

%temp% is an operating system environment variable that represents the path to the temporary directory.

The file gdpdu.zip contains the compressed data media provision (including the index.xml file and the gdpdu-01-08-2002.dtd file). After decompression the auditor therefore opens the new data provision in the directory %TEMP%\gdpdu.

Note:

The <Version> specification does not relate to the version of the default data structure but the version of the data carrier session.

Frequently asked questions

- **Do we need to deliver data in XML format?**
No. Only the description of the data (the index.xml file) is in XML format. The reference data are in FixedLength or VariableLength format.
- **Can your system handle data in DOS/Unix/Mac formats (line breaks)?**
Yes, because the line breaks can be specified explicitly.
- **How do negative figures need to be indicated (in front of/behind the figure, space character between preceding character and digit permissible)?**
Negative figures must be preceded or followed by a minus sign without a space character (e.g. "-1782,90" or "1782,90-").
- **When a transferred file is too large for a specific data medium, may it then be split across a number of data media?**
This default data structure does not provide for the splitting up of a file. However, in an instance like this, you may compress data or select a data medium with a higher capacity (e.g. DVD).
- **Which XML parser do you use?**
We use Xerces (<https://xerces.apache.org/xerces-c/>).
- **Is it an error when the ForeignKey references a table that is not contained in the DataSet, or would the ForeignKey then be ignored?**
The description of the links must match the delivered data. Otherwise, your data carrier session will not be consistent. The import process will indicate this inconsistency to the auditor.
- **What happens when the index.xml file describes a table that is, however, subsequently not present?**
The description of the files must match the delivered data. Otherwise, your data carrier session will not be consistent. The import process will indicate this inconsistency to the auditor.

- **Are commands compulsory? Do you offer environment variables for commands?**
Commands are not compulsory. When a .BAT or .CMD file is accessed, the conventional operating system environment variables are available. The import process itself does not set any environment variables.
- **How can it finally be checked whether the data of the data medium/data package can be imported by your system?**
The safe method of testing is importing with IDEA for GDPdU/GoBD (IDEA + SmartX). SmartV will help you to validate the index.xml file and the data to be imported regarding their conformity with the default data structure.

A simple option for testing the XML file for its syntactic correctness is to open it with an XML-capable browser.
- **Since I am not familiar with XML standards, I am not sure how to construct the files index.xml and gdpdu.dtd concretely. I have sent you a VariableLength file. Would you be able to create the corresponding XML or DTD file for me?**
You should acquire some basic XML knowledge. You do not have to and may not modify the DTD file.
The description of the data (in the index.xml file) should always be carried out from the point that has the data ownership, in other words the software manufacturer or the business subject to taxation.
Audicon offers assistance in verifying the compliance of data carrier session with the default data structure and in the construction of the index.xml file (see <https://www.audicon.net/consulting/dienstleistungspakete/> – in German).
- **In some press releases, apart from VariableLength and FixedLength file formats, various others are mentioned. Which formats apply to data carrier sessions?**
The file formats referred to in some articles relate to the original import options of the software IDEA.
The default data structure is, however, product independent and in its current version only supports the two generally applicable formats of VariableLength and FixedLength.
- **Must the gdpdu-01-08-2002.dtd file also be written to the data medium?**
Yes. A data medium serving data carrier session purposes must contain the gdpdu-01-08-2002.dtd file, the index.xml file and all reference data (e.g. T1.CSV, T2.CSV, T3.CSV, T4.ASC).

- In a VariableLength file, the field names are stored in the first dataset. How is it possible to peruse the first set?

Use the Range specification in the Table element:

```
<Table>
  <Range>
    <From>2</From>
  </Range>
  <URL>Region.csv</URL>
  <VariableLength>
    <VariablePrimaryKey>
      <Name>RegionId</Name>
      <AlphaNumeric/>
    </VariablePrimaryKey>
    <VariableColumn>
      <Name>Description</Name>
      <AlphaNumeric/>
    </VariableColumn>
  </VariableLength>
</Table>
```

Figure 16: Range specification

- By specifying a character set, e.g. UTF7 or Macintosh, is the specification of the line delimiter overridden automatically?

No. Only by explicitly specifying the line delimiter (RecordDelimiter) is the standard value (CRLF) overridden.

Synopsis of default data structure version 1.0 and version 1.1

Renaming is indicated in **blue** and new or deleted elements in **red**.

Version 1.0 of 10 May 2002 (6/6/2002)	Version 1.1 of 1 August 2002
<pre> <?xml version="1.0" encoding="UTF-8"?> <!--Versions available: 1.0 (May-10-2002) --> <!-- Start Simple Types --> <!-- Supplementary Vocabulary --> <!ELEMENT Version (#PCDATA)> <!ELEMENT Location (#PCDATA)> <!ELEMENT Comment (#PCDATA)> <!ELEMENT Length (#PCDATA)> <!ELEMENT References (#PCDATA)> <!ELEMENT From (#PCDATA)> <!ELEMENT To (#PCDATA)> <!-- Specifies which character (if any) encapsulates a CSV text column --> <!ELEMENT TextEncapsulator (#PCDATA)> <!-- Specifies how many digits appear to the right of the decimal symbol. --> </pre>	<pre> <?xml version="1.0" encoding="UTF-8"?> <!--Versions available: 1.1 (August-01-2002) --> <!-- Start Simple Types --> <!-- Supplementary Vocabulary --> <!ELEMENT Version (#PCDATA)> <!ELEMENT Location (#PCDATA)> <!ELEMENT Comment (#PCDATA)> <!ELEMENT Length (#PCDATA)> <!ELEMENT References (#PCDATA)> <!ELEMENT From (#PCDATA)> <!ELEMENT To (#PCDATA)> <!-- Specifying a maximum length for a VariableLength column can reduce a VariableLength tables' import time. If MaxLength is not specified then we parse URL to determine the MaxLength for each column. * Only applies to VariableLength tables. --> <!ELEMENT MaxLength (#PCDATA)> <!-- Specifies which character (if any) encapsulates a VariableLength AlphaNumeric column. Doublequote is the default TextEncapsulator " * Only applies to VariableLength tables. (Optional) -- > <!ELEMENT TextEncapsulator (#PCDATA)> <!-- Specifies how many digits appear to the right of the decimal symbol. </pre>

Version 1.0 of 10 May 2002 (6/6/2002)	Version 1.1 of 1 August 2002
<pre> <!-- The decimal place is not always stored with numbers. If each number is supposed to have decimal places use ImpliedAccuracy --> <!-- The first line in CSV URL contains column headers. A value of False is interpreted if the element is missing. Any columns specified in the XML file take precedence over the file. This lets you override some columns in the file and leave the others as they are. --> <!-- Enables you to change how gdpdu displays dates. The default is implementation dependant. --> <!-- Specifies the symbol that indicates decimal values. The default is implementation dependant. Specified once per Table. --> <!-- Specifies the symbol that groups the digits in large numbers. The default is implementation dependant. Specified once per Table --> <!-- Command(s) are executed in the following manner </pre>	<pre> CAUTION: Results are undefined when importing numeric data with greater Accuracy than the Accuracy defined in index.xml For example trying to import the value 1000,25 with an accuracy of 0 might result in 1000 or an error. This behavior is specific to the implementation. Zero is the default Accuracy '0' (Optional) --> <!-- The decimal place is not always stored with numbers. If each number is supposed to have decimal places use ImpliedAccuracy --> <!-- Enables you to change how gdpdu displays dates. DD.MM.YYYY is the default Format --> <!-- Specifies the symbol that indicates decimal values. Comma is the default DecimalSymbol. ',' Specified once per Table. --> <!-- Specifies the symbol that groups the digits in large numbers. Dot is the default DigitGroupingSymbol or ThousandsSeperator. '.' Specified once per Table --> <!-- Command(s) are executed in the following manner </pre>

Version 1.0 of 10 May 2002 (6/6/2002)	Version 1.1 of 1 August 2002
<pre> * before the import process * after the import process * before a Media is imported * after a Media is imported --> <!ELEMENT Command (#PCDATA)> <!-- Only the file protocol is supported at this time. The root path is the path of Media The following are all invalid * http://www.somewhere.com/data/Accounts.csv * ftp://ftp.somewhere.com/data/Accounts.csv * ... The following is valid. * file://Accounts.csv --> <!ELEMENT URL (#PCDATA)> <!-- Textual description of specified element --> <!ELEMENT Description (#PCDATA)> <!-- The logical names of specified element --> <!ELEMENT Name (#PCDATA)> <!-- Y2K Window Any year before Epoch is 2000+ Default value 50. --> </pre>	<pre> * before the import process * after the import process * before a Media is imported * after a Media is imported --> <!ELEMENT Command (#PCDATA)> <!-- Only the file protocol is supported at this time. * The standard uses relative URLs. Absolute URLs are not allowed. The following are all invalid: * http://www.somewhere.com/data/Accounts.dat * ftp://ftp.somewhere.com/data/Accounts.dat * file://localhost/Accounts.dat * file:///Accounts.dat The following are valid examples * Accounts.dat * data/Accounts.dat * data/january/Accounts.dat * ../Accounts.dat --> <!ELEMENT URL (#PCDATA)> <!-- Textual description of specified element (Optional) --> <!ELEMENT Description (#PCDATA)> <!-- The logical name of specified element. Sometimes referred to business name. If missing, URL will be used in place of Name. --> <!ELEMENT Name (#PCDATA)> <!-- Y2K Window Any year before Epoch is 2000+ Default value 30. --> </pre>

Version 1.0 of 10 May 2002 (6/6/2002)	Version 1.1 of 1 August 2002
<pre> <!ELEMENT Epoch (#PCDATA)> <!-- Specifies the symbol that separates CSV columns or ASCIIFixedLength records. --> <!ELEMENT Delimiter (#PCDATA)> <!-- The number of bytes skipped before reading of URL commences. --> <!ELEMENT SkipNumBytes (#PCDATA)> <!-- End Simple Types --> <!-- Start Complex Types --> <!-- Self-explanatory --> <!ELEMENT Range (From, (To Length)?)> <!-- The document element --> <!ELEMENT DataSet (Version, DataSupplier?, Command*, Media+, Command*)> <!-- Supported datatypes AlphaNumeric is the default datatype when a datatype is not defined. --> <!ELEMENT AlphaNumeric EMPTY> <!ELEMENT Date (Format?, Epoch?)> <!ELEMENT Numeric ((ImpliedAccuracy Accuracy)?)> </pre>	<pre> <!ELEMENT Epoch (#PCDATA)> <!-- Element(s) that separate columns or records. Semicolon is the default ColumnDelimiter. ';' CRLF or &#13;&#10; is the default RecordDelimiter. --> <!ELEMENT ColumnDelimiter (#PCDATA)> <!ELEMENT RecordDelimiter (#PCDATA)> <!-- The number of bytes skipped before reading of URL commences. Zero is the default when not specified. '0' --> <!ELEMENT SkipNumBytes (#PCDATA)> <!-- End Simple Types --> <!-- Start Complex Types --> <!-- Self-explanatory --> <!ELEMENT Range (From, (To Length)?)> <!ELEMENT FixedRange (From, (To Length))> <!-- The document element --> <!ELEMENT DataSet (Version, DataSupplier?, Command*, Media+, Command*)> <!-- Supported datatypes (mandatory) --> <!ELEMENT AlphaNumeric EMPTY> <!ELEMENT Date (Format?)> <!ELEMENT Numeric ((ImpliedAccuracy Accuracy)?)> <!-- Supported codepages: Be careful to explicitly define RecordDelimiter when using a non-default codepage. ANSI is the default codepage when not specified --> <!ELEMENT ANSI EMPTY> <!ELEMENT Macintosh EMPTY> <!ELEMENT OEM EMPTY> <!ELEMENT UTF16 EMPTY> <!ELEMENT UTF7 EMPTY> <!ELEMENT UTF8 EMPTY> </pre>

Version 1.0 of 10 May 2002 (6/6/2002)	Version 1.1 of 1 August 2002
<pre> <!-- Supported file formats --> <!ELEMENT ASCIIFixedLength ((Length Delimiter)?, ((FixedPrimaryKey+, FixedColumn*) (FixedPrimaryKey*, FixedColumn+)), ForeignKey*)> <!ELEMENT FixedColumn (Name, Description?, (Numeric AlphaNumeric Date)?, Map*, Range)> <!ELEMENT FixedPrimaryKey (Name, Description?, (Numeric AlphaNumeric Date)?, Map*, Range)> <!ELEMENT CSV (NamesFirstRow?, Delimiter?, TextEncapsulator?, ((PrimaryKey+, Column*) (PrimaryKey*, Column+)), ForeignKey*)> <!ELEMENT Column (Name, Description?, (Numeric AlphaNumeric Date)?, Map*)> <!ELEMENT PrimaryKey (Name, Description?, (Numeric AlphaNumeric Date)?, Map*)> <!-- Description of the entity supplying the data. (Optional) --> <!ELEMENT DataSupplier (Name, Location, Comment)> <!-- The first Media will contain index.xml. Importing will process each media listed --> <!ELEMENT Media (Name, Command*, Table+, Command*)> <!-- Elements common to ASCIIFIXEDLENGTH & CSV are propagated to Table. CSV is the default format if none is specified. --> <!ELEMENT Table (URL, Name?, Description?, Validity?, (DecimalSymbol, DigitGroupingSymbol)?, SkipNumBytes?, Range?, (CSV ASCIIFixedLength)?)> <!-- ForeignKeys denote joins and relationships between tables. Not specifying ForeignKeys prevent joins from occurring. --> </pre>	<pre> <!-- Supported file formats: FixedLength, VariableLength --> <!ELEMENT FixedLength ((Length RecordDelimiter)?, ((FixedPrimaryKey+, FixedColumn*) (FixedColumn+)), ForeignKey*)> <!ELEMENT FixedColumn (Name, Description?, (Numeric AlphaNumeric Date), Map*, FixedRange)> <!ELEMENT FixedPrimaryKey (Name, Description?, (Numeric AlphaNumeric Date), Map*, FixedRange)> <!ELEMENT VariableLength (ColumnDelimiter?, RecordDelimiter?, TextEncapsulator?, ((VariablePrimaryKey+, VariableColumn*) (VariableColumn+)), ForeignKey*)> <!ELEMENT VariableColumn (Name, Description?, (Numeric (AlphaNumeric, MaxLength?) Date), Map*)> <!ELEMENT VariablePrimaryKey (Name, Description?, (Numeric (AlphaNumeric, MaxLength?) Date), Map*)> <!-- Description of the entity supplying the data. (Optional) --> <!ELEMENT DataSupplier (Name, Location, Comment)> <!-- The first Media will contain index.xml. Importing will process each media listed --> <!ELEMENT Media (Name, Command*, Table+, Command*)> <!-- Elements common to FixedLength & VariableLength are propagated to Table. --> <!ELEMENT Table (URL, Name?, Description?, Validity?, (ANSI Macintosh OEM UTF16 UTF7 UTF8)?, (DecimalSymbol, DigitGroupingSymbol)?, SkipNumBytes?, Range?, Epoch?, (VariableLength FixedLength))> <!-- ForeignKeys denote joins or relationships between tables. To successfully join two tables make sure both the PrimaryKey and the referenced column (foreignkey) are of the same datatype. Results are undefined when joining two tables with different key datatypes. Most likely an error will occur. --> </pre>

Version 1.0 of 10 May 2002 (6/6/2002)	Version 1.1 of 1 August 2002
<pre> <!ELEMENT ForeignKey (Name+, References)> <!-- Maps data from 'From' to 'To' ie. From To ===== True 1 True -1 False 0 A map is basically an associative container. Take caution that 'From' is of the same type as the column datatype. --> <!ELEMENT Map (Description?, From, To)> <!-- Documentation for table validity. --> <!ELEMENT Validity (Range, Format?)> <!-- End Complex Types --> </pre>	<pre> <!ELEMENT ForeignKey (Name+, References)> <!-- Maps AlphaNumeric columns from 'From' to 'To' ie. From To ===== True 1 True -1 False 0 Basically, a map is an associative container. The standard implementation only supports AlphaNumeric datatypes. The following conversions are NOT supported. Numeric -> AlphaNumeric Date -> AlphaNumeric AlphaNumeric -> Date AlphaNumeric -> Numeric --> <!ELEMENT Map (Description?, From, To)> <!-- Documentation for table validity. --> <!ELEMENT Validity (Range, Format?)> <!-- End Complex Types --> </pre>

Table 48: Comparison of default data structure version 1.0 and 1.1

List of figures

Figure 1: Auditor.....	6
Figure 2: Data carrier session.....	7
Figure 3: Data organization	8
Figure 4: Import process.....	11
Figure 5: XML DTD	14
Figure 6: Default data structure – organization	15
Figure 7: Default data structure – DataSet.....	16
Figure 8: Default data structure – FixedLength.....	17
Figure 9: Default data structure – VariableLength.....	18
Figure 10: Example 1 – index.xml file	37
Figure 11: Example 2 – tables with transaction data and master data	38
Figure 12: Example 2 – master data.....	38
Figure 13: Example 2 – transaction data	39
Figure 14: Example 3 – index.xml file	40
Figure 15: Example 3 – uncompress.bat file.....	40
Figure 16: Range specification	43

List of tables

Table 1: Default data structure change history.....	3
Table 2: Document change history	3
Table 3: Overview of common operators	19
Table 4: List of names and numbers for the characters " & < >.....	19
Table 5: ELEMENT Version.....	20
Table 6: ELEMENT Location	20
Table 7: ELEMENT Comment	20
Table 8: ELEMENT Length.....	20
Table 9: ELEMENT References	21
Table 10: ELEMENT From	21
Table 11: ELEMENT To	21
Table 12: ELEMENT MaxLength	21
Table 13: ELEMENT TextEncapsulator	22
Table 14: ELEMENT Accuracy	22
Table 15: ELEMENT ImpliedAccuracy	22
Table 16: ELEMENT Format	23
Table 17: ELEMENT DecimalSymbol	23
Table 18: ELEMENT DigitGroupingSymbol	23
Table 19: ELEMENT Command	24
Table 20: ELEMENT URL	24
Table 21: ELEMENT Description	25
Table 22: ELEMENT Name	25
Table 23: ELEMENT Epoch.....	26
Table 24: ELEMENT ColumnDelimiter	26
Table 25: ELEMENT ColumnDelimiter – list of common separators	26
Table 26: ELEMENT RecordDelimiter	27
Table 27: ELEMENT RecordDelimiter – list of common separators	27
Table 28: ELEMENT SkipNumBytes	27
Table 29: ELEMENT Range	27
Table 30: ELEMENT FixedRange.....	28
Table 31: ELEMENT AlphaNumeric	28
Table 32: ELEMENT Date	28
Table 33: ELEMENT Numeric	29
Table 34: ELEMENT UTF8.....	30
Table 35: ELEMENT FixedLength	31
Table 36: ELEMENT FixedColumn.....	31
Table 37: ELEMENT FixedPrimaryKey.....	31
Table 38: ELEMENT VariableLength.....	31
Table 39: ELEMENT VariableColumn	32
Table 40: ELEMENT VariablePrimaryKey	32
Table 41: ELEMENT DataSupplier	32
Table 42: ELEMENT Media	32
Table 43: ELEMENT Table.....	33
Table 44: ELEMENT ForeignKey.....	33
Table 45: ELEMENT Map – redefinitions.....	34
Table 46: ELEMENT Map.....	34
Table 47: ELEMENT Validity	34
Table 48: Comparison of default data structure version 1.0 and 1.1	49