
RCM-DX specification

Document Version v2.0.4-dev Commit ID
cc334a6dab9d79971b0f6b69f40e9f7db97b1771

Swiss federal railways SBB, Schweizerische
Bundesbahnen SBB, Chemins de fer fédéraux suisses
CFF, Ferrovie federali svizzere FFS



Contents

1	Specification of the RCM-DX Format	3
1.1	License	3
1.2	Contribution	3
1.3	Change history	3
1.4	Introduction	7
1.4.1	Motivation	7
1.4.2	Hints	7
1.5	Definitions	8
1.5.1	File names	8
1.5.2	Primitive and extended data types	9
1.5.3	(HDF5) Group	10
1.5.4	(HDF5) Attribute	11
1.5.5	(HDF5) Datasets	11
1.6	Data structures	13
1.6.1	Array	13
1.6.2	Coordinates	15
1.6.3	Pictures	18
1.6.4	Videos	20
1.7	Time-based data structures	21
1.7.1	Timestamp	21
1.7.2	Durations	21
1.8	RCM-DX file format	22
1.9	RCM-DX data hierarchy	22
1.9.1	Root	24
1.9.2	File	26
1.9.3	Data Processing	27
1.9.4	Processing log	28
1.9.5	Clearance	29
1.9.6	Platform	30
1.9.7	Session	32
1.9.8	Session configuration	34
1.9.9	Topology	35
1.9.10	Track	36
1.9.11	Line	39
1.9.12	Switch Track	40
1.9.13	Track Object	42

1.9.14	Track Point	43
1.9.15	Property	44
1.9.16	Setting	45
1.9.17	Sections	46
1.9.18	Measuring System	49
1.9.19	Datasource	50
1.9.20	Channel	51
1.9.21	Measurement mode	57
1.9.22	Logging	59
1.9.23	Position	64
1.9.24	EVENTS	67
1.9.25	Measurement system configuration	72
1.10	XML Schema Definitions	73
1.10.1	Events Comment	73
1.10.2	Events Defect	73
1.10.3	Events Generic	74
1.10.4	RCM-DX Data types	76

1 Specification of the RCM-DX Format

1.1 License

TODO: Lizenzhinweise notieren für RCM-DX TODO: Copyright hier vermerken TODO: Lizenzhinweise oder Copyright notieren für HDF5

1.2 Contribution

The RCM-DX specification is open source and freely accessible and usable by all with respect to the license. Any person who wants to improve this specification can do so. How exactly this works can be read on the Github website. The repository can be found at [RCM-DX repository](#) and the website at [RCM-DX specification website](#).

1.3 Change history

RCM-DX version	Document version	Date	Author	Change
1.0.0	0.1	06.03.2015	Martin Frey (SCS)	Initial version
1.0.0	0.2	20.03.2015	Martin Frey (SCS)	Extensions
1.0.0	0.3	15.04.2015	Patrik Wernli (SCS)	Review
1.0.0	0.4	20.04.2015	Martin Frey (SCS)	Extensions and revised
1.0.0	0.5	03.05.2015	Patrik Wernli (SCS)	Formal Adaptions
1.0.0	0.6	12.05.2015	Martin Frey (SCS)	PDR Feedback: Storing of Booleans, comments allowed on all levels, format independent of video codec, flags (including simulation) on session level. Schemas for exceedances, comments and drawings added
1.0.0	0.7	13.07.2015	Patrik Wernli (SCS)	Finalized for CDR

RCM-DX specification

RCM-DX version	Document version	Date	Author	Change
1.0.0	0.8	02.10.2015	Martin Frey (SCS)	Event model added, reference to specification event schema added.
1.0.0	0.9	30.11.2015	Patrik Wernli (SCS)	Adaptions for Infotrans position model. Version concept removed. Event model updated.
1.0.0	0.10	21.12.2015	Martin Frey (SCS)	Review
1.0.0	0.11	21.12.2015	Patrik Wernli (SCS)	Revised after review
1.0.0	0.12	16.02.2016	Pascal Brem (SCS)	Topology model in configuration.
1.0.0	0.13	17.02.2016	Martin Frey (SCS)	Review topology model
1.0.0	0.14	19.02.2016	Pascal Brem (SCS)	Sections added to file format.
1.0.0	0.15	23.02.2016	Martin Frey (SCS)	Global configuration and settings updated
1.0.0	0.16	11.07.2016	Pascal Brem (SCS)	New Hash code attribute for the topology.
1.0.0	0.17	15.07.2016	Pascal Brem (SCS)	New units and data types for positions
1.0.0	0.18	03.01.2018	Pascal Brem (SCS)	New GTG Track Id in the Topology.
1.0.0	0.19	03.01.2018	Pascal Brem (SCS)	Events are stored on session level.
1.1.0	0.20	09.01.2018	Pascal Brem (SCS)	Events and Sections in a group.
1.1.0	0.21	09.01.2018	Patrik Wernli (SCS)	Added chapter “HDF5 File Format Versions”
1.1.0	0.22	11.04.2018	Patrik Wernli (SCS)	Changed document template to official publishing template
1.1.0	0.23	16.08.2018	Pascal Brem (SCS)	Changes in the channel basis definition.

RCM-DX specification

RCM-DX version	Document version	Date	Author	Change
1.1.0	0.24	16.08.2018	Pascal Brem (SCS)	New attributes on the picture block channel.
1.2.0	0.25	04.09.2018	Pascal Brem (SCS)	New minor version.
1.3.0	0.26	28.11.2018	Pascal Brem (SCS)	New minor version for the topology attributes.
1.4.0	0.27	08.01.2019	Pascal Brem (SCS)	New availability group.
1.4.1	0.28	05.06.2019	Pascal Brem (SCS)	New switchtracks in the DfA
2.0.0	0.29	21.04.2020	Michael Ammann (SBB), Jakob Grilj (SBB)	Adaptation of the structure to new requirements. New major release with version number 2.0, due to major changes in the structure and goal for publication of the specification (open source).
2.0.0	0.30	27.10.2020	Aron Serafini (SCS)	Adjust consistency, measurementMode and processing chapters.
2.0.0	0.31	11.03.2021	Aron Serafini (SCS)	Adjust specification to implementation
2.0.0	0.32	31.08.2021	Aron Serafini (SCS)	Fix mistake: Topology is a child of CONFIGURATION, not Session.
2.0.1	1.0	06.04.2022	Aron Serafini (SCS)	Add attribute MeasurementDirectionDependent to Channel. Use a release version for this document.
2.0.1	1.01	05.06.2022	Mathias Vanden Auweele (Infrabel)	Removed 'group' in all titles and replaced with a reference to HDF5 group. Fix definition of UNRELEASED under the CLEARANCE group. Clarify the definition of "Platform". Changed the limitative platform list into an example platform list.

RCM-DX specification

RCM-DX version	Document version	Date	Author	Change
2.0.1	1.02	26.06.2023	Mathias Vanden Auweele (Infrabel)	Fixed formatting issues with document links. Improved definition of ChannelBasis. Fixed documentation bug with MeasurementDirectionDependent attribute specification
2.0.2	1.03	29.06.2023	Aron Serafini (SCS)	Rename MeasurementDirectionDependent to MoveDirAutoInvert (it was implemented under this name in the RCM-DX file). Add new optional session attributes: SessionAlias, SegmentationRuleset, PositionAlgoVersion.
2.0.2	1.04	23.11.2023	Stephanie Schalbetter (SBB)	Changes in the document according to the review with Infrabel, SBB and SCS.
2.0.3	1.05	03.10.2024	Stephanie Schalbetter (SBB)	Rename of UNRELEASED to REJECTED in the CLEARANCE group.
2.0.4	1.06	13.02.2025	Stephanie Schalbetter (SBB)	New enums in consistencytype.
2.0.4	1.07	20.03.2025	Stephanie Schalbetter (SBB)	Clarification of the description of position channel datasets.
2.0.4	1.08	31.10.2025	Stephanie Schalbetter (SBB)	Correction of attribute MoveDirAutoinvert to optional.
2.0.4	1.09	24.03.2026	Stephanie Schalbetter (SBB)	Correction Channel CommonTriggerDistance and CommonTriggerFrequency to 32 float.
2.0.5	-	TBD	TBD	Starting with release 2.0.5 the Document version will be discontinued and not updated anymore.

1.4 Introduction

1.4.1 Motivation

Railroad companies continuously gather data of their rail, overhead line, telecommunications, civil engineering structure and signaling systems by means of mobile and stationary measuring systems. Data flows from these systems through processing units – which enrich, evaluate and validate the data –, to systems that display the data to subject matter experts and also to systems that automatically analyse it.

This specification defines the rail condition monitoring data exchange format (RCM-DX format) which is a data format optimised for data in the railroad context, i.e. for data points localised within a railroad network. The RCM-DX format is a file format based on the HDF5 specification and defines a structure of HDF5 groups, datasets, and attributes. The document at hands also describes the content of the elements defined. Although the format is open and can in principle be implemented right away by any railroad company, this specification contains a few non-generic elements and naming conventions that are specific to SBB and Infrabel. The reason for this is that any file that adheres to this specification can be used with the RCM Viewer, an application available soon to the public.

The RCM-DX format is a file format detailing the HDF5 format version 2.0. HDF5 was chosen for several reasons, including that it is an open format. HDF5 is a hierarchical data storage where the data is arranged in a tree structure. The HDF5 format is described on the webpage of the [HDF5-group](#), in particular on the site [HDF5 file format specification](#). The HDF5 group offers tools and libraries for various programming languages and operating systems that allow to read and write HDF5 files.

1.4.2 Hints

1.4.2.1 RCM-DX structure RCM-DX defines a structure of HDF5 groups, datasets, and attributes that software solutions that use this format must adhere to.

The extension of the specification is permitted. However, it must be taken into account that such data may no longer be read or processed by existing systems.

1.4.2.2 Versioning The RCM-DX data format is subject to changes, these are indicated by the version number in the document, see chapter [Root](#). The version number consists of two numbers, separated by dots and is composed as follows: [Major].[Minor].[Feature]. Example: 1.0.0

Major

Defines the main version and indicates when major changes have been made. These are, for example, that changing the basic structure or renaming groups, datasets or attributes (major, minor attribute as an example), which are mandatory.

Minor

Indicates minor changes, such as changing the name of an attribute that is not mandatory or defining new groups, attributes, or datasets. These changes do not affect anything that cannot be read with an existing RCM-DX read-write library.

Feature

Indicates feature changes.

1.4.2.3 Diagrams The following figure shows the color coding used in diagrams:

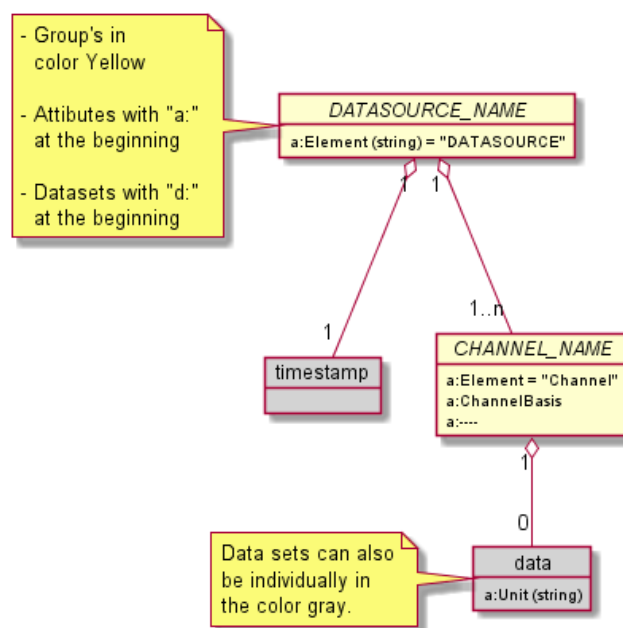


Figure 1: RCM-DX diagram overview

1.5 Definitions

This document defines technical restrictions as well as content descriptions. This chapter provides an overview of the data types used, types of names, and other important points.

1.5.1 File names

Files following the RCM-DX specification get their own defined file extension, which is `rcmdx`.

Example of a file name: `20201228_081522_TGMS.rcmdx`.

1.5.2 Primitive and extended data types

1.5.2.1 Primitive data types Below are the supported data types used in this specification including a short description. Not all data types possible with HDF5 are described. The read and write library specifies which data types can be used for data sets and attributes.

Data type	Description	Examples
boolean	Two values are possible, <code>true</code> or <code>false</code> , 0 or 1	<code>true</code>
byte	One byte or an unlimited number of bytes are possible	-
signed integer	Positive and negative values are allowed. Possible bit depths are: 8, 16, 32 or 64	1256442; -62334
unsigned integer	Only positive values are allowed. Possible bit depths are: 8, 16, 32 or 64	1256442; -62334
float	Positive and negative values are allowed. Possible bit depths are: 32 or 64	12.53; -3212.546
string	Multiple characters of undefined length	"RCM-DX is great!"

1.5.2.2 Extended data types Extended data types use the primitive data types, but have a more specialized format or meaning. The following is a list of the extended data types used.

Name	Description	Example
Timestamp	Unique and worldwide defined format of a time in nanoseconds since January 1, 1970 00:00 UTC without leap seconds, defined under wikipedia.org/wiki/Timestamp . Datatype is 64 bit unsigned integer	1553237099000000000

Name	Description	Example
Enum	Enumerations are predefined (and always capitalized) strings defined in this specification. Datatype is <code>string</code>	<code>MAX, MIN, RIGHT, LEFT</code>

1.5.3 (HDF5) Group

If we are talking about a group in this document, we mean the groups in HDF5 format (of the type `HDF5 Group`). These contain additional groups or datasets.

If a name of a group in this document is written in capital letters (for example `TOPOLOGY`), it is exactly the same as in the RCM-DX file. If the naming of a group is not fixed, the corresponding chapter describes in more detail how the name is composed.

Groups are described in this specification as follows:

Name	Parent object	Optional
<code>GROUP</code>	<code>RCMDX</code>	<code>no</code>

Name

The name of the group.

Group names whose ending is “_NAME” are wildcard names and are replaced as described in the corresponding paragraph. Example: `GROUP_NAME`

Parent object

A group can be a subgroup of a group, here the name of this group is mentioned. If the name is written in quotation marks, it can be freely chosen by the creator of the file. Without quotation marks, the name of the group is meant.

Optional

If a group is marked as optional (`optional = yes`), it can be omitted if it is not needed. If a group is marked as non-optional (`Optional = no`), then this group must exist if the parent group also exists, otherwise it does not.

1.5.4 (HDF5) Attribute

In the RCM-DX, attributes, groups and datasets can be assigned. The names of the attributes are written in the UpperCamelCase-Notation¹. Attributes are always of type `HDF5 attribute` unless otherwise specified.

Attributes are described in this specification as follows:

Name	Data type	Parent object	Optional	Description
StartTime	64 bit integer	SESSION_NAME	no	Start time in milliseconds, for example: 1553237099000000000

Name

The name of the attribute.

Data type

Primitive data type of the attribute, this describes the type of the content in the attribute itself.

Parent object

An attribute is always assigned to a group or a data set, here the name of this group or data set is mentioned.

Optional

If an attribute is marked as optional (`Optional = yes`), it can be omitted if it is not required. If an attribute is marked as non-optional (`Optional = no`), this attribute must be present if the group or dataset to which the attribute applies is also present, otherwise it is not.

Description

Description and or examples of the attribute.

¹Upper-Camel-Case-Notation: The Upper Camel Case Notation defines the way a composite name is written. Further information can be found at the following link: [Upper Camel Case](#)

1.5.5 (HDF5) Datasets

A data set is the structure in which data is stored. Examples of data are sensor, measurement, or position data. A data set is always of the HDF5 type `HDF5 Dataset`. In the HDF5 Specification a “Dataset” is a container for data objects. A channel as a group always has one data set. A channel

defines a type of sensor data that is stored in its data set. Several channels form a data source. Further information on the structure is described further down in this specification.

Below is a list of possible ways in which data can be stored in the RCM-DX:

Type of storage	Description
Array	Data array of arbitrary length
Image	An image taken at a defined time
Video	A video that has been streamed into several individual blocks of defined size, split and saved

The datasets are described in the lowerCamelCase-Notation². Datasets are described in this specification as follows:

Name	Data type	Parent object	Optional	Storage type
timestamp	Timestamp	DATASOURCE_NAME	no	Array

Name

The name of the data set.

Data type

Primitive data type of the content in the data set, thus the data type of the contained data.

Parent object

A data set is always assigned to a group, here the name of this group is mentioned.

Optional

If the dataset is marked as optional (**Optional** = **yes**), it can be omitted if it is not needed. If the dataset is marked as non-optional (**Optional** = **no**), this dataset must be present if the parent group also has these datasets, otherwise not.

Storage type

One of the storage types described in this chapter.

Descriptions of the data set are added outside the table.

²lower-Camel-Case-Notation: The lower camel case notation defines the way a composite name is written. Further information can be found under the following link: www.wikipedia.org/wiki/Camel_case

1.5.5.1 HDF5 Chunking The “HDF5 Chunking” is for data within a data set. This means that the data is divided into blocks, which in turn can be processed independently. This also allows faster access to parts of the data. Whether a splitting is allowed and recommended can be seen with each data set, for example: “HDF5 Chunking” is allowed and recommended. The HDF5 chunking is described in more detail on the website of the HDF5 group: www.support.hdfgroup.org/HDF5/doc/H5.user/Chunking.html

1.6 Data structures

Within a channel group, one of the following structures can be contained: Array, Limits, Coordinates, Pictures or Videos. These structures are described in more detail in this chapter.

1.6.1 Array

Channels which record individual measured values contain a data set with the name `data`, this data set is never optional. Single values are stored in this data set as a 1D array, the length of this array (or list) is not limited.

The possible data types are defined in chapter [Primitiv and extended data types](#)

Multidimensional measured values are given their own channel group per dimension and thus their own data set called `data`.

Name	Data type	Parent object	Optional	Storage type
<code>data</code>	A primitive or extended data type	CHANNEL_NAME	no	<code>Array</code>

The following attributes are assigned to this type of data set:

Name	Data type	Parent object	Optional	Description
<code>Unit</code>	string	data set <code>data</code>	no	A physical unit or empty if the data does not correspond to a physical unit

1.6.1.1 Limits Limits are stored in group **LIMIT**. Limits are threshold values for measurements. They are used for comparison and analysis of measurements.

A channel group can contain zero or one limit groups. Each limit sub-group **LIMIT_NAME** contains its own **timestamp** data set and contains a **duration** data set. The data set **limitvalue** contains the limit values. When which limit value is valid is defined in the dataset **timestamp** and how long it is valid is defined in the dataset **duration**. Values at the same index position of all three data resources belong together.

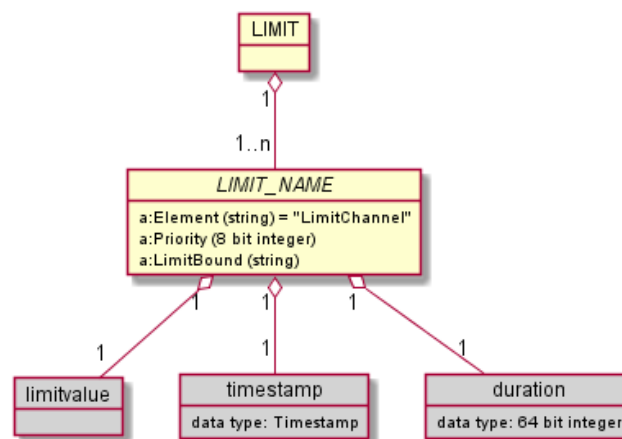


Figure 2: Limit group overview

The group of limit values is defined as follows:

Name	Parent object	Optional
LIMIT	CHANNEL_NAME	no

The group **LIMIT** now contains further groups, each with the name of the limit exceeding:

Name	Parent object	Optional
LIMIT_NAME	LIMIT	no

The following attributes are assigned to this group:

Name	Data type	Parent object	Optional	Description
Priority	8 bit integer	LIMIT_NAME	no	Priority of defined limit, lower values are higher priorities.
LimitBound	Enum	LIMIT_NAME	no	Defines the type of limit, possible values are MAX or MIN .

It contains the following datasets:

Name	Data type	Parent object	Optional	Storage type
limitvalue	A primitive or extended data type	LIMIT_NAME	no	Array
timestamp	Timestamp	LIMIT_NAME	no	Array
duration	Timestamp	LIMIT_NAME	no	Array

1.6.2 Coordinates

Measurement data that can be assigned to a coordinate system are given a defined name according to the following pattern: [coord.CN](#).

This type of data storage allows several entries to be recorded per measurement timestamp. Thus there are more entries in these datasets than in the data set [timestamp](#). How many entries per timestamp belong to each other (as a group) is stored in another data set with the name [sampleindex](#). The data set [sampleindex](#) is described in chapter [Sample index](#).

Element	Description
coord	Simple character string for identifying data of type Coordinates
.	Separators
C	Additional character for identifying data of type Coordinates

Element	Description
N	Index number beginning with “0”, increasing for each additional coordinate datasets

The data set is defined as follows:

Name	Data type	Parent object	Optional	Storage type
coord.CN	A primitive or extended data type	CHANNEL_NAME	no	Single

The following attributes are assigned to this type of data set `coord.CN`:

Name	Data type	Parent object	Optional	Description
<code>Unit</code>	string	Dataset <code>coord.CN</code>	no	One physical unit or empty if the data does not have any physical unit

1.6.2.1 Coordinate related measured values Further measured values can be recorded for each coordinate measuring point. These are stored in individual data sets. In the following the definition of these data sets:

Element	Description
value	Simple character string for identifying data of type Coordinates
.	Separators
V	Additional character for identifying data of type value
N	Index number beginning with “0”, increasing for each additional value set

The data set is defined as follows:

Name	Data type	Parent object	Optional	Storage type
value.VN	A primitive or extended data type	CHANNEL_NAME	yes	Single

The following attributes are assigned to this type of data set `value.VN`:

Name	Data type	Parent object	Optional	Description
Type	string	Dataset <code>value.VN</code>	no	Describes the content and type of the data it contains.

1.6.2.2 Sample Index If datasets are created for coordinates, a data set on the same level and with the name `sampleindex` must be available. The index number of an entry in `coord.CN`, is entered as the start of the next group. If the data set `sampleindex` has a value of 21 at index zero, the first 20 entries from the data set `coord.CN` belong together, the next data group starts at index number 21. The number of entries in `sampleindex` corresponds to the one in the data set `timestamp`.

The group sizes can vary among themselves, this can be seen from the index positions in `sampleIndex`.

Example

The rail cross profile serves as an example here. At one point on the railroad track, several data points of a rail profile are measured and stored. A channel with the name `coord.C0` for the X-axis and `coord.C1` is created for the Y-axis.

The data set `sampleindex` now contains the number of entries that belong together.

Below is a picture of a rail cross section measurement with about 2000 measuring points:

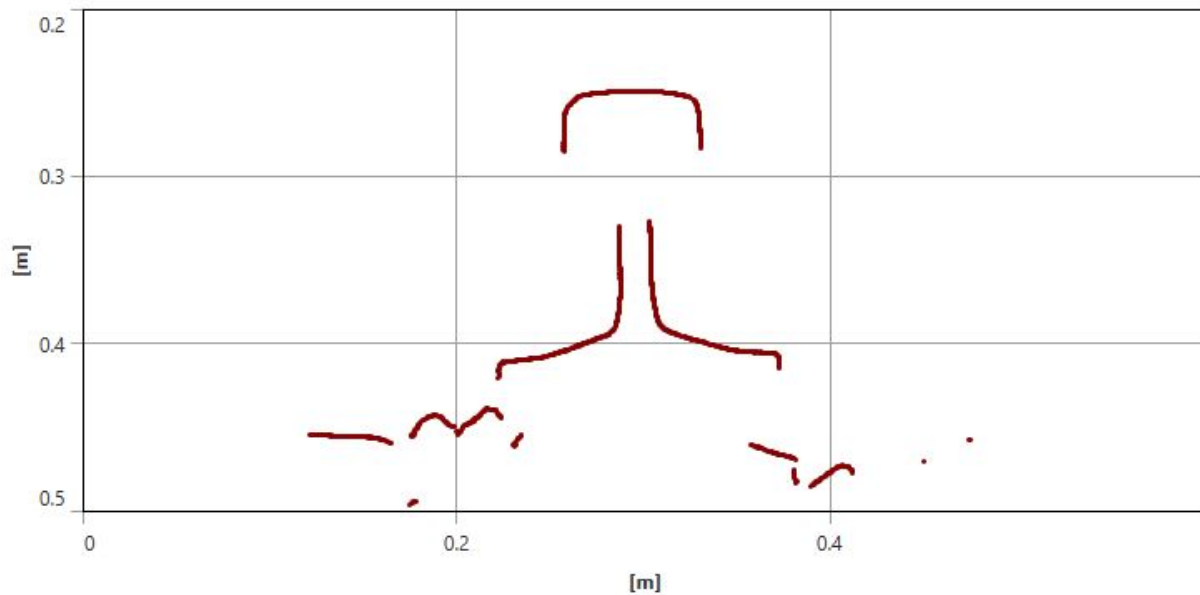


Figure 3: Image of a rail cross section measurement

1.6.3 Pictures

Images can be saved in compressed or uncompressed form. The format of the images is stored in an attribute so that the image can be read correctly.

Images are stored as binary data blocks, so an image results in a data set. All images are stored in a group called **IMG**. All images in this group have the same properties that are stored in the attributes.

Name	Parent object	Optional
IMG	DATASOURCE_NAME	no

The group **IMG** gets the following attributes for the more detailed description of the images contained therein:

Name	Data type	Parent object	Optional	Description
ImageContentType	string	IMG	yes	Image content type, for example JPEG
ResolutionType	Enum	IMG	no	Description in chapter Image resolution types

Name	Data type	Parent object	Optional	Description
ResolutionInfoX	32 bit float	IMG	no	Resolution in X direction
ResolutionInfoY	32 bit float	IMG	no	Resolution in Y-direction

Images can have different resolutions in X and Y direction, this must be considered for a correct representation and evaluation of the images.

1.6.3.1 Image resolution types The ‘ResolutionType’ attribute contains information about the values of the ‘ResolutionInfoX’ and ‘ResolutionInfoY’ attributes.

ResolutionType’ can contain the following values: ‘none’, ‘mmPerPixel’ or ‘dimension’

[none](#)

No information available.

[mmPerPixel](#)

Defines the pixel size in mm. The values in [ResolutionInfoX](#) and [ResolutionInfoY](#) give information about the real size of a pixel in millimeters.

[dimension](#)

Defines the resolution of the image. The values in [ResolutionInfoX](#) and [ResolutionInfoY](#) give information about the image size in pixels.

1.6.3.2 Naming the data set for an image

Name	Data type	Parent object	Optional	Storage type
img.NNNNNNNNN	integer, bit depth depending on color depth	IMG	no	image

The images are named according to the following pattern: [img](#) . NNNNNNNNN, hereinafter a description of the individual elements.

Element	Description
img	String for the name of an image

Element	Description
.	Separators.
NNNNNNNNNN	Picture number, beginning with 000000000 (nine characters)

1.6.4 Videos

Name	Data type	Parent object	Optional	Storage type
vid.NNNNNNNNNN	integer, bit depth depending on color depth	VID	no	image

As with the images, videos can be saved in compressed or uncompressed form. Videos are stored as streams in individual blocks. The blocks are single datasets with a given name.

Name	Parent object	Optional
VID	DATASOURCE_NAME	no

1.6.4.1 Name of the data set for a video A video data block is named according to the following pattern: `vid . NNNNNNNNNN`, hereinafter a description of the individual elements.

Element	Description
vid	String for the name of a video
.	Separator
NNNNNNNNNN	Video number, starting with 000000000 (nine characters), ascending +1

1.7 Time-based data structures

1.7.1 Timestamp

Each entry in a data set of a channel has a reference to an entry in a data set with the name `timestamp`, which lies within the data source group. In this `timestamp` data set, there are as many entries as there are entries in a data set of a channel.

The time stamps are always stored in ascending order.

Name	Data type	Parent object	Optional	Storage type
timestamp	Timestamp	DATASOURCE_NAME	no	Array

These time stamps are recorded either by a defined distance travelled or by a frequency, this is described in more detail in the chapter [Trigger mode](#).

1.7.2 Durations

If data is recorded that is valid for a certain period of time, the data set with the name `duration` is added to the data set `timestamp`. The timestamp recorded in the `timestamp` data set specifies the time at which the value was recorded and the `duration` data set specifies how long this value is valid in nanoseconds. The data set `duration` is `timestamp` within a data source group next to the data set.

The differentiation between discrete (data for discrete `timestamp`) and continuous (data for `timestamp` with `duration`) data is done on data source level. The existence or absence of a `duration` array defines if the data source is “Discrete” (no `duration` array) or “Continuous” (with `duration` array).

Example: Assuming there is a data source with a temperature value every second and a calculated average temperature for every minute. Such an average temperature would be stored in a continuous data source, within an array, with a duration of 60 sec.

Name	Data type	Parent object	Optional	Storage type
duration	64 bit integer	DATASOURCE_NAME	no for continuous values, otherwise yes	Array

1.8 RCM-DX file format

The RCM-DX consists of a file format of the HDF5-group www.hdfgroup.org/HDF5 in version 2.0. This allows to save the data in a tree structure. This structure, or rather the naming of the groups and datasets, is not specified by the HDF5 group, but by the RCM-DX specified here. The datasets can hold different data, what exactly is contained is specified as metadata.

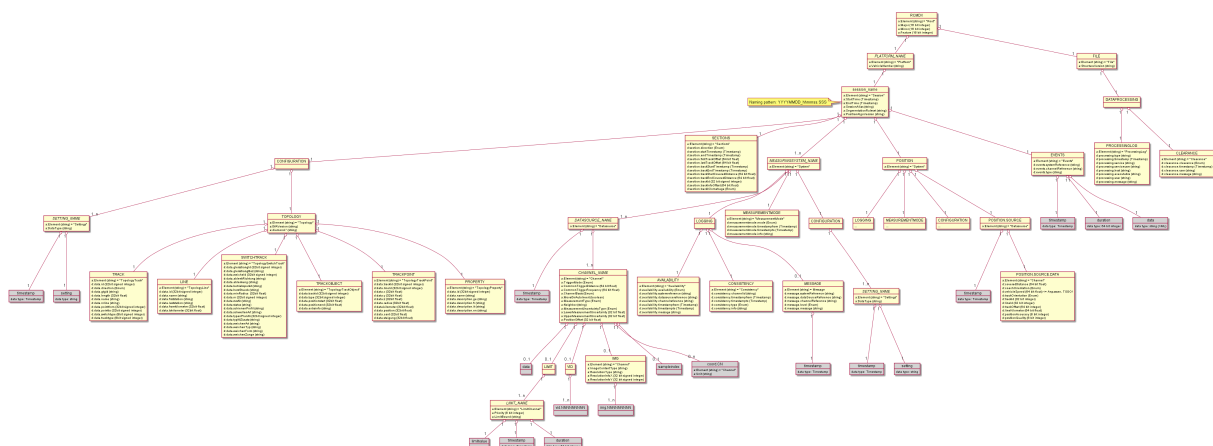
A change to the structure means a new version and thus a new release of this specification.

To read and write the HDF5 file format, the HDF5 group offers libraries for different languages. These can read and write the structure specified here without problems.

Further information about the structure of the HDF5 file format can be found under the following link: www.portal.hdfgroup.org/display/HDF5/Introduction+to+HDF5

1.9 RCM-DX data hierarchy

In the RCM-DX, the individual groups and datasets as well as their names are defined. Below is an overview of the structure specified in this document:



- ▼  20231114-123807_DFZ00_TGMS_v004_created-20231114-19
 - ▼  RCM DX
 - ▼  DFZ00
 - ▼  20231114_123807.000
 - ▼  CONFIGURATION
 - ▼  TOPOLOGY
 - >  LINE
 - >  PROPERTY
 - >  SWITCHTRACK
 - >  TRACK
 - >  TRACKOBJECT
 - >  TRACKPOINT
 - ▼  DFZ00.TGMS
 - >  CONFIGURATION
 - >  DFZ00.TGMS.TGPAR_112
 - >  LOGGING
 - >  MEASUREMENTMODE
 - >  EVENTS
 - ▼  POSITION
 -  CONFIGURATION
 - >  LOGGING
 - >  MEASUREMENTMODE
 - ▼  POSITION.SOURCE
 - ▼  POSITION.SOURCE.DATA
 -  data.covereddistance
 -  data.direction
 -  data.kilometrage
 -  data.line_id
 -  data.lineoffset
 -  data.positionaccuracy
 -  data.positionquality
 -  data.track_id
 -  data.trackoffset
 -  timestamp
 - >  SECTIONS
 - ▼  FILE
 - ▼  DATAPROCESSING
 - >  CLEARANCE
 - >  CONVERTERSOURCE
 - >  PROCESSINGLOG

Separate and more detailed specifications have been written for individual structure groups. Several

measuring instruments can be installed on one measuring platform. Each of these measuring devices generates new channels of data, which flow into the RCM-DX. Since these channels can be different for each measuring device, the specifications were separated. Another reason for this is the fact that other railway operators use different measuring and inspection equipment.

The individual groups are specified in more detail below in the subcategories.

1.9.1 Root

Root is of type 'Group' in HDF5.

The root group contains all other subgroups. This group defines the RCM-DX and bears its name and thus refers to this specification.

Name	Parent object	Optional
RCMDX	none (this is the root node)	no

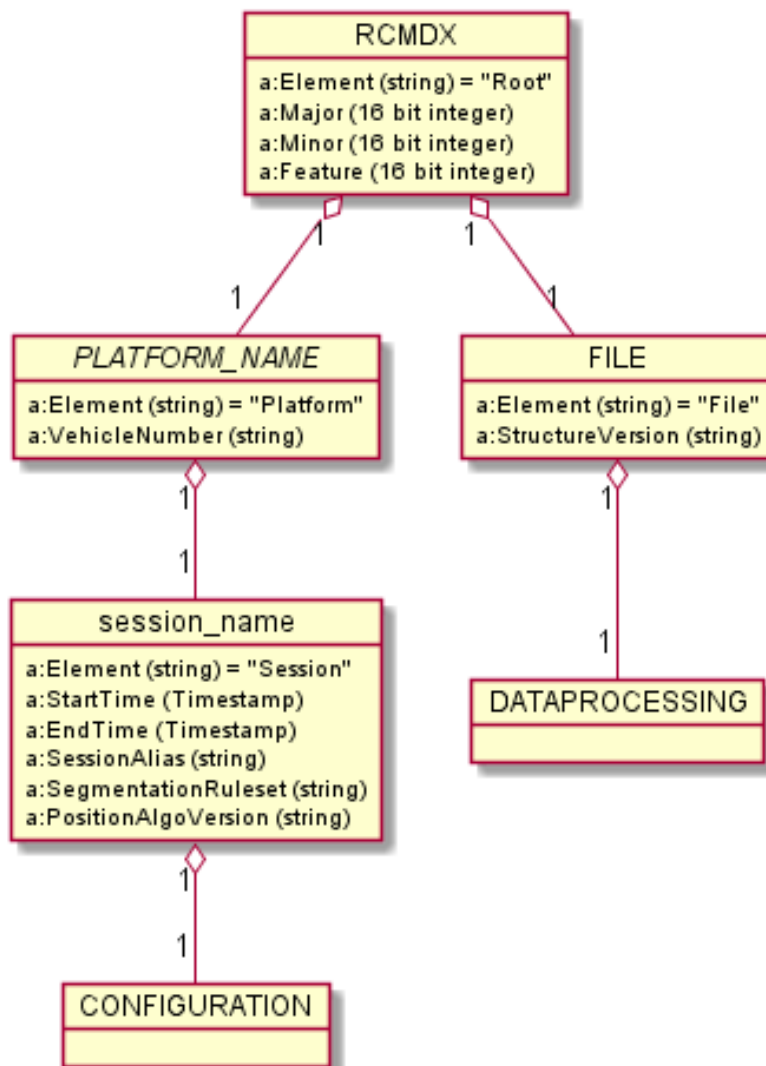


Figure 4: Root group overview

1.9.1.1 Attributes The following attributes are assigned to the group **RCM-DX**:

Name	Data type	Parent object	Optional	Description
Major	16 bit integer	RCM-DX	no	Major Version of the RCM-DX specification that corresponds to the structure of the created file

RCM-DX specification

Name	Data type	Parent object	Optional	Description
Minor	16 bit integer	RCMDX	no	Minor Version of the RCM-DX specification that corresponds to the structure of the created file
Feature	16 bit integer	RCMDX	no	Feature Version of the RCM-DX specification that corresponds to the structure of the created file

1.9.2 File

File is of type 'Group' in HDF5.

The file group contains file specific information, i.e. information specific for the given data set in its current processing state.

Name	Parent object	Optional
FILE	RCMDX	no

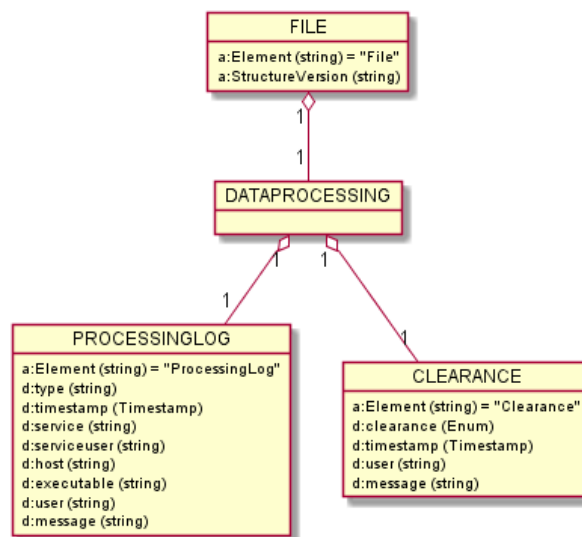


Figure 5: File group overview

1.9.2.1 Attributes The file group contains the following attribute:

Name	Data type	Optional	Description
Element	string	no	Equals to “File”. Identifies this node.
StructureVersion	string	no	Version identifier to describe the current measurement system setups on the platform. All underlying systems, datasources and channels can be identified based on this version. A change in StructureVersion happens for example when a measurement channel is added or removed from the platform.

1.9.3 Data Processing

Data Processing is of type ‘Group’ in HDF5.

The data source group **DATAPROCESSING** contains information about the data in this file, and the processing.

Name	Parent object	Optional
DATAPROCESSING	FILE	yes

1.9.4 Processing log

Processing log is of type ‘Group’ in HDF5.

The data source group [PROCESSINGLOG](#) contains information on data processing. This information is written by systems that make changes to the data. These changes, for example, can be a conversion from millimeters to meters.

Name	Parent object	Optional
PROCESSINGLOG	DATAPROCESSING	yes

1.9.4.1 Processing log datasets

The group [PROCESSINGLOG](#) contains one dataset:

Name	Data type	Parent object	Optional	Storage type
Element	string	no	Equals to “ProcessingLog”. Identifies this node.	
type	string	PROCESSINGLOG	no	Array
timestamp	64 bit integer	PROCESSINGLOG	no	Array
service	string	PROCESSINGLOG	no	Array
serviceUser	string	PROCESSINGLOG	no	Array
host	string	PROCESSINGLOG	no	Array
executable	string	PROCESSINGLOG	no	Array
user	string	PROCESSINGLOG	yes	Array
message	string	PROCESSINGLOG	yes	Array

All these datasets represent different columns in a table. Their sizes will therefore always be identical.

type

This record contains the type of the data processing step. The following are possible types: [CREATION](#), [CONVERSION](#), [MERGE](#), [CONSISTENCY_CHECK](#), [UNIT_CONVERSION](#), [COMMENT](#), ...

timestamp

Contains the time of the acquisition of the entry.

service

Name of the service that applied this processing step.

serviceUser

(Technical) User that runs the service.

host

Host that the service is run on.

executable

Executable that the service is run from.

user

Optional user-ID of the initiator of this processing step.

message

Optional message of the user.

1.9.5 Clearance

Clearance is of type 'Group' in HDF5.

This group is used by SBB to record information about the data release of all parties who have processed this data. The information is stored in the form of key-value pairs in a data set.

Name	Parent object	Optional
CLEARANCE	DATAPROCESSING	yes

The group [CLEARANCE](#) contains one dataset:

Name	Data type	Parent object	Optional	Storage type
Element	string	no	Equals to "Clearance". Identifies this node.	

Name	Data type	Parent object	Optional	Storage type
clearance	Enum	CLEARANCE	no	Array
timestamp	64 bit integer	CLEARANCE	no	Array
user	string	CLEARANCE	no	Array
message	string	CLEARANCE	yes	Array

All these datasets represent different columns in a table. Their sizes will therefore always be identical.

clearance

This record contains the enum value of the clearance. The following values are possible:

Name	Description
RELEASED	This file has been released
PENDING	The clearance has not been set yet
REJECTED	This file won't be released

timestamp

Contains the time of the acquisition of the entry.

user

User-ID of the initiator of this clearance.

message

Optional message of the user.

1.9.6 Platform

Platform is of type 'Group' in HDF5.

A platform group contains information about the measuring vehicle, mobile or handheld device that collects the data.

A platform is in this case a measurement platform and can be a vehicle / carrier, or a mobile device installed on a vehicle.

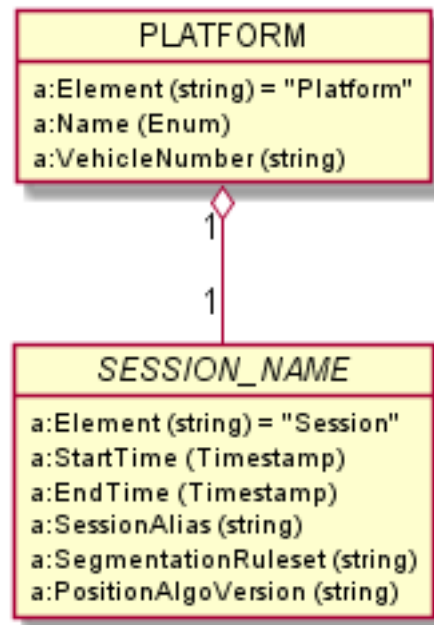


Figure 6: Platform group overview

The naming of the group is defined according to which platform produced the data. An overview of all names and the corresponding platform is specified in the chapter [Platforms example list](#).

Name	Parent object	Optional
Platform	RCMDX	no

1.9.6.1 Attributes The platform group contains the following attributes:

Name	Data type	Optional	Description
Element	string	no	Equals to "Platform". Identifies this node.
VehicleNumber	string	no	Unique number of the vehicle

1.9.6.2 Platforms example list Below is a list of the defined unique names of the platforms, their abbreviation and the associated vehicle number.

Platform Name	Abbreviation	Vehicle Number
DFZ00	DFZ	-
DFZ01	gDFZ	-
EM130	EM130	99 88 9162001-0
EM201	EM201	99 88 9163201-5

The Vehicle number is the European Vehicle Number (EVN).

1.9.7 Session

Session is of type 'Group' in HDF5.

The session group contains all data (measurement and other) that was collected during a specific period on a specific platform. A RCM-DX file contains exactly one session group.

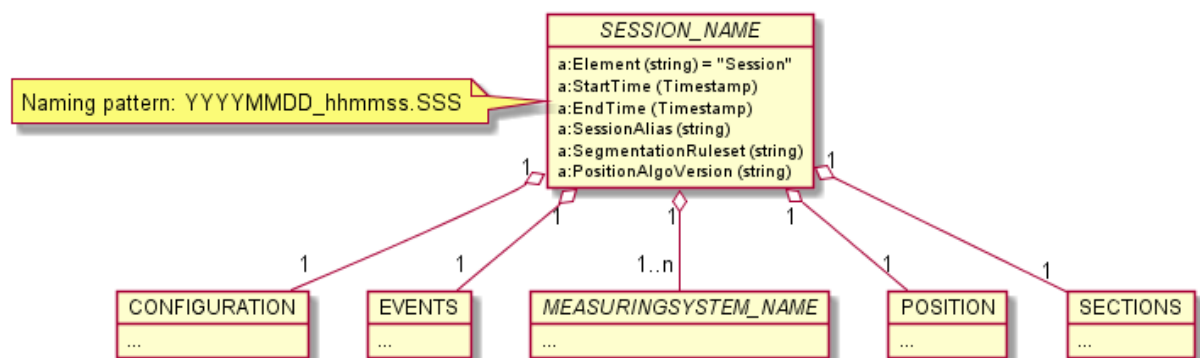


Figure 7: Session group overview

1.9.7.1 Naming The name of a session group is assigned according to the following pattern:

Name	Parent object	Optional
YYYYMMDD_hhmmss.SSS	PLATFORM	no

Example: 20190212_231255.592

The individual elements and their meaning are described below:

Pattern	Content
YYYY	The year in four digit representation
MM	The month in the year (01 for January)
DD	The day in the month
hh	The hour in the day (0-23)
mm	The minute in the hour
ss	The seconds in the minute
SSS	The milliseconds in the seconds
“_” or “.”	Characters as separator

For a certain period of time, only one session can exist in a file, this must be ensured by the creator of the file.

1.9.7.2 Attributes

Name	Data type	Parent object	Optional	Description
Element	string	SESSION_NAME	no	Names the type of the group, this is fix “Session”
StartTime	Timestamp	SESSION_NAME	no	Timestamp in nanoseconds as start time of the session
EndTime	Timestamp	SESSION_NAME	yes	Timestamp in nanoseconds as end time of the session. If the session has not yet been closed, this attribute is missing

Name	Data type	Parent object	Optional	Description
SessionAlias	string	SESSION_NAME	yes	Alias for this session, useful if the measurements of this session have an external identifier by which they can be recognised
PositionAlgoVersion	string	SESSION_NAME	yes	Version of the positioning algorithm that was used for this session
SegmentationRule	string	SESSION_NAME	yes	Identifier of the ruleset used to create a segmentation of tracks of this session

1.9.8 Session configuration

Session configuration is of type 'Group' in HDF5.

A session configuration is a set of parameters or settings about the session. These can include the rail network (topology), the IT network, or other configurations. Measurement system configurations are stored in a separate Group.

Configurations can be stored in the datasets of this group. The datasets are designed so that global and network specific configurations can be stored. The configuration can change and have not to be the same in each session.

Name	Parent object	Optional
CONFIGURATION	SESSION_NAME	no

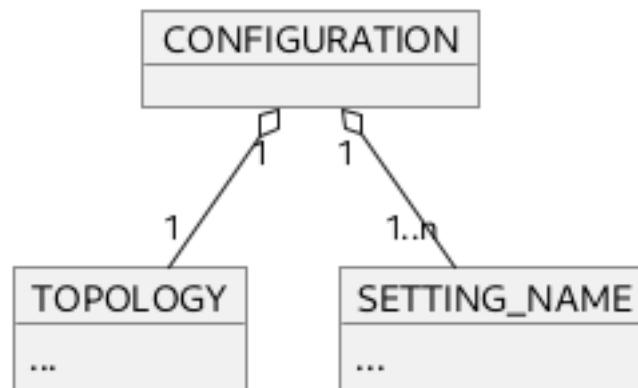


Figure 8: Session configuration group overview

1.9.9 Topology

Topology is of type 'Group' in HDF5.

A topology group contains all information on the route network of the respective railway company. This chapter proposes an optimised solution for SBB and may differ between railway companies. SBB's data processing chain provides for this structure, which is why it is described here. Other variants are allowed as long as dependencies on other Groups such as the Groups SECTION and POSITION are taken into account. The Group TOPOLOGY is not optional, if the RCM-DX format is intended to be self-containing. The Group TOPOLOGY is optional and can be omitted, if the topology is saved outside the RCM-DX format, for example in an external database.

Name	HDF5 Type	Parent object	Optional
TOPOLOGY	HDF5 Group	CONFIGURATION	no (self-containing format) / yes (not self-containing format)

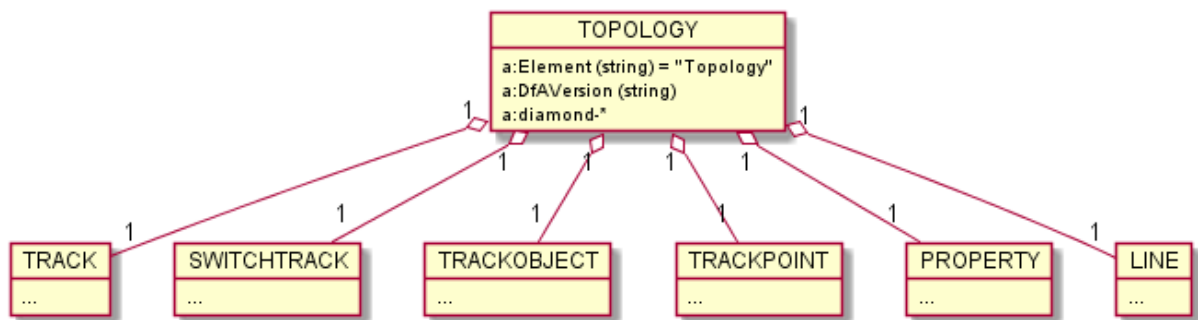


Figure 9: Topology group overview

1.9.9.1 Attributes The group **TOPOLOGY** contains the following attributes:

Name	Data type	Parent object	Optional	Storage type
Element	string	TOPOLOGY	no	Equals to “Topology”. Identifies this node.
DfAVersion	string	TOPOLOGY	no	Array
diamond-*	string	TOPOLOGY	no	Array

DfAVersion

Version number of topology, included to check validity.

The DfA (Database of fixed assets) is a SBB construct and reflects the SBB route network. The data comes from a database and is distributed as a file to the SBB measuring vehicles. They can read the information contained therein and also add it to the RCM-DX. This DfA is used for positioning and it is therefore possible to assign the measured data to an object from the route network.

Diamond-Metadata Attributes

List of attributes defined by SBB Diamond. All attributes start with “diamond-“ and have a String value.

1.9.10 Track

Track is of type ‘Group’ in HDF5.

This group contains information on the tracks of the railway network. The information is stored in separate datasets.

Name	Parent object	Optional
TRACK	TOPOLOGY	no

The following datasets are included in this group, some of which are described in more detail in the subchapters:

Name	Data type	Parent object	Optional	Storage type
Element	string	TRACK	no	Equals to “TopologyTrack”. Identifies this node.
direction	Enum	TRACK	no	Array
id	32 bit integer	TRACK	no	Array
gtgid	string	TRACK	no	Array
length	string	TRACK	no	Array
name	string	TRACK	no	Array
nrline	string	TRACK	no	Array
pointfrom	32 bit integer	TRACK	no	Array
pointto	32 bit integer	TRACK	no	Array
switchtype	8 bit signed integer	TRACK	no	Array
tracktype	8 bit signed integer	TRACK	no	Array

direction

The direction of a switch is specified in this data set.

If the track is of the type “switch”, a value other than ‘-’ must be selected here. Which number means what is shown in the following table:

Value	Description
-	No crossover

Value	Description
straight	Straight line switch track
left	Left-handed switch
right	switch running to the right

id

ID of the track.

gtgid

Unique GTG ID of a GTG string, this ID is stored as UUID.

length

The length of the track section.

name

Name of the track section.

pointfrom

ID of the starting point of the track section.

pointto

ID of the end point of the track section.

switchtype

If the track is of the type “Switch”, a value greater than zero must be selected here. Which number means what is shown in the following table:

Value	Description
0	Anything but a turnout
1	Simple switch
2	Double switch
3	Simple crossovers
4	Double track connection
5	Double crossover

trackType

The number in the tracktype data set defines the type of track that belongs to it. Which number means what is shown in the following table:

Value	Description
0	Station track
1	Track
2	Switch

1.9.11 Line

Line is of type 'Group' in HDF5.

This group contains information about a line in the route network. The information is stored in separate datasets.

Name	Parent object	Optional
LINE	TOPOLOGY	no

The following datasets are included in this group:

Name	Data type	Parent object	Optional	Storage type
Element	string	Line	no	Equals to "TopologyLine". Identifies this node.
id	32 bit signed integer	LINE	no	Array
name	string	LINE	no	Array
firststation	string	LINE	no	Array
laststation	string	LINE	no	Array
fromkilometer	string	LINE	no	Array
tokilometer	string	LINE	no	Array

id

Defines the ID of the line, this is unique.

name

The name of the line.

firststation

The name of the first station of this line.

laststation

The name of the last station of this line.

km

Start kilometre of the line, expressed in kilometres.

tokilometer

Final kilometer of the line, in kilometers.

1.9.12 Switch Track

Switch track is of type 'Group' in HDF5.

This group contains information about switches in the route network. The information is stored in separate datasets.

Name	Parent object	Optional
SWITCHTRACK	TOPOLOGY	no

The following datasets are included in this group:

Name	Data type	Parent object	Optional	Storage type
Element	string	SWITCHTRACK	no	Equals to "TopologySwitchTrack". Identifies this node.
gleisstrangId	32 bit signed integer	SWITCHTRACK	no	Array
gleisstrangBez	string	SWITCHTRACK	no	Array

Name	Data type	Parent object	Optional	Storage type
weicheld	32 bit signed integer	SWITCHTRACK	no	Array
ablenkRichtung	string	SWITCHTRACK	no	Array
ablenkung	string	SWITCHTRACK	no	Array
betriebspunkt	string	SWITCHTRACK	no	Array
herzStueck	string	SWITCHTRACK	no	Array
minRadius	32 bit signed integer	SWITCHTRACK	no	Array
nr	32 bit signed integer	SWITCHTRACK	no	Array
zusNr	string	SWITCHTRACK	no	Array
schienenProfil	string	SWITCHTRACK	no	Array
status	string	SWITCHTRACK	no	Array
schwellenArt	string	SWITCHTRACK	no	Array
typenPlanNr	32 bit signed integer	SWITCHTRACK	no	Array
typNrZusatz	string	SWITCHTRACK	no	Array
weichenArt	string	SWITCHTRACK	no	Array
weichenTyp	string	SWITCHTRACK	no	Array
weichenForm	string	SWITCHTRACK	no	Array
weichenZunge	string	SWITCHTRACK	no	Array

gleisstrangId

A reference to the GTG-ID.

gleisstrangBez

Contains a description of the track section.

weicheld

Contains the ID's of the switches as a reference.

1.9.13 Track Object

Track object is of type 'Group' in HDF5.

This group contains information about objects in the route network, for example a balise. The information is stored in separate datasets.

Name	Parent object	Optional
TRACKOBJECT	TOPOLOGY	no

The following datasets are included in this group:

Name	Data type	Parent object	Optional	Storage type
Element	string	TRACKOBJECT	no	Equals to "TopologyTrackObject". Identifies this node.
trackid	32 bit signed integer	TRACKOBJECT	no	Array
type	32 bit signed integer	TRACKOBJECT	no	Array
positionstart	32 bit signed integer	TRACKOBJECT	no	Array
positionend	32 bit signed integer	TRACKOBJECT	no	Array
extrainfo	string	TRACKOBJECT	no	Array

trackid

Contains the ID of the track to which the track is connected.

type

Type of the object.

positionstart

Start position of the object in meters.

positionend

End position of the object in meters.

Extrainfo

Additional information about the object, for example, the ID of a balise.

1.9.14 Track Point

Track point is of type 'Group' in HDF5.

This group contains information about defined points on the route network. The information is stored in separate datasets.

Name	Parent object	Optional
TRACKPOINT	TOPOLOGY	no

The following datasets are included in this group:

Name	Data type	Parent object	Optional	Storage type
Element	string	TRACKPOINT	no	Equals to “TopologyTrackPoint”. Identifies this node.
trackid	32 bit signed integer	TRACKPOINT	no	Array
lineid	32 bit signed integer	TRACKPOINT	no	Array
x	32 bit float	TRACKPOINT	no	Array
y	32 bit float	TRACKPOINT	no	Array
z	32 bit float	TRACKPOINT	no	Array
radius	32 bit float	TRACKPOINT	no	Array
kilometer	32 bit float	TRACKPOINT	no	Array
position	32 bit float	TRACKPOINT	no	Array
cant	32 bit float	TRACKPOINT	no	Array
steigung	32 bit float	TRACKPOINT	no	Array

trackid

Reference to the ID of the track section.

lineid

Reference to the ID of the line.

x

X-Coordinate of the point.

y

Y-Coordinate of the point.

z

Z-Coordinate of the point.

radius

The radius of a point, given in meters.

kilometer

Contains the line kilometre of the point in the route network, expressed in kilometres.

position

Position of the point, in meters.

cant

The inclination at this point, expressed in millimetres.

steigung

Gradient at this point, expressed in parts per thousand.

1.9.15 Property

Property is of type 'Group' in HDF5.

This group contains information about properties of the topology itself. The information is stored in separate datasets.

Name	Parent object	Optional
PROPERTY	TOPOLOGY	no

The following datasets are included in this group:

Name	Data type	Parent object	Optional	Description
Element	string	PROPERTY	no	Equals to “TopologyProperty”. Identifies this node.
id	32 bit signed integer	PROPERTY	no	Array
name	string	PROPERTY	no	Array
description.ge	string	PROPERTY	no	Array
description.fr	string	PROPERTY	no	Array
description.it	string	PROPERTY	no	Array
description.en	string	PROPERTY	yes	Array

id

Unique ID of the characteristic.

name

Name of the characteristic.

description.ge

Description of the characteristic in the language German.

description.fr

Description of the characteristic in French language.

description.it

Description of the characteristic in Italian language.

description.en

Description of the feature in English language.

1.9.16 Setting

Setting is of type ‘Group’ in HDF5.

Within this group there are further groups whose names identify the type of setting.

In the following SETTING_NAME is used as placeholder of the actual name of a setting.

Name	Parent object	Optional
SETTING_NAME	CONFIGURATION	yes

This group contains two datasets:

Name	Data type	Parent object	Optional	Storage type
Element	string	SETTING_NAME	no	Equals to “Settings”. Identifies this node.
setting	string	SETTING_NAME	no	Array
timestamp	64 bit integer	SETTING_NAME	no	Array

setting

Contains the actual configuration.

timestamp

Contains the time from when this configuration is valid and was used.

The following attributes are contained in this group:

Name	Data type	Parent object	Optional	Description
DataType	string	setting	no	Defines the datatype of the configuration within the data set setting . Data type specified as MIME ³ type, for example Content-Type: < text/strings>

1.9.17 Sections

Section is of type ‘Group’ in HDF5.

The group [SECTIONS](#), contains information about a session. This group has dependencies to the group [TOPOLOGY](#).

Name	Parent object	Optional
SECTIONS	SESSION_NAME	no

1.9.17.1 Data fields The following data fields are contained in the group “SECTIONS”:

Name	Data type	Parent object	Optional	Storage type
Element	string	SECTIONS	no	Equals to “Sections”. Identifies this node.
direction	Enum	SECTIONS	no	Array
startTimestamp	Timestamp	SECTIONS	no	Array
endTimestamp	Timestamp	SECTIONS	no	Array
firstTrackOffset	64 bit float	SECTIONS	no	Array
lastTrackOffset	64 bit float	SECTIONS	no	Array
trackInfoOffset	64 bit float	SECTIONS	no	Array
trackId	32 bit signed integer	SECTIONS	no	Array
trackStartTimestamp	Timestamp	SECTIONS	no	Array
trackEndTimestamp	Timestamp	SECTIONS	no	Array
trackKilometrage	Enum	SECTIONS	no	Array
trackStartCoveredDistance	64 bit float	SECTIONS	no	Array
trackEndCoveredDistance	64 bit float	SECTIONS	no	Array

direction

Indicates the orientation of travel of the measuring vehicle in the given section. This array contains only as many entries as there are sections.

Possible values of this enumeration are:

Value	Orientation of travel
FORWARD	forward
BACKWARD	backward

`startTimestamp`

Start time of the section as time stamp since 1.1.1970 at 00:00 UTC.

`endTimestamp`

End time of the section as time stamp since 1.1.1970 at 00:00 UTC. `endTimestamp` must be greater than `startTimestamp`.

`firstTrackOffset`

Indicates the distance in meters between the start of the track and the position at the beginning of the section. This array contains only as many entries as there are sections.

`lastTrackOffset`

Indicates the distance in meters between the position at the end of the measurement and the end of the rail. This array contains only as many entries as there are sections.

`trackInfoOffset`

This data set lists how many entries in the datasets of the “Track list group” belong to section. One entry is created per section in a session and the number of entries is defined. The number of tracks belonging to each section can be determined by calculating the specified offset value at the x position minus the offset value at the $x - 1$ position.

`trackId`

ID of the tracks that are part of the section.

`trackStartTimestamp`

Time since beginning of the session at which the measurement on a track started.

`trackEndTimestamp`

Time since the beginning of the session at which the measurement on a track ended. `endTimestamp` must be greater than `startTimestamp`.

`trackKilometrage`

Orientation of the track with respect to the section’s driving direction.

The following values are allowed:

Value	Meaning
ASCENDING	The track kilometrage increases with the driving direction
DESCENDING	The track kilometrage decreases with the driving direction

trackStartCoveredDistance

Start covered distance of the track in the section.

trackEndCoveredDistance

End covered distance of the track in the section.

1.9.18 Measuring System

Measurement system is of type 'Group' in HDF5.

Each measuring system has its own data sources, which have their own names, as well as their own channels, which in turn have their own names. Common features are described in this specification, everything else is defined in a separate specification. Since this part differs greatly among railway companies and with measuring equipment, a rigid specification has been dispensed with, but a certain framework is still given.

A group is created for each system that collects data. The name of the group is unique for each system. The composition of this name is not predefined.

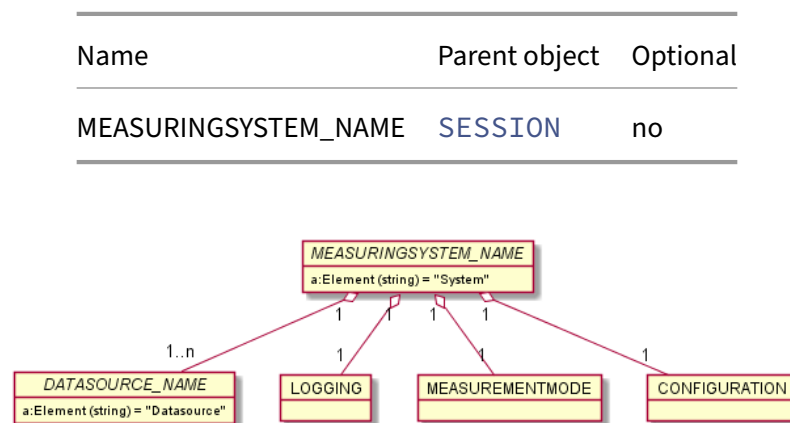


Figure 10: Measuring system overview

1.9.18.1 Attributes The following attributes are contained in the group of the measuring system:

Name	Data type	Parent object	Optional	Description
Element	string	MEASURINGSYSTEM_NAME	no	Indicates the type of the group, this is fixed System

1.9.19 Datasource

Datasource is of type 'Group' in HDF5.

A data source group can contain several channels and thus several data sources. This group combines these channels. The naming can be freely selected, but must be unique.

Name	Parent object	Optional
DATASOURCE_NAME	MEASURINGSYSTEM_NAME	no

A timestamp is available for each individual measuring point within a data source group. There are two types of data acquisition for a data source group. One is always after a defined distance (e.g. every 250 millimeters) and the other is the recording of measurement data at a certain frequency (e.g. 4000 Hz). The way the measurement data was recorded is shown in two attributes for each channel group. For a description see **Trigger mode**.

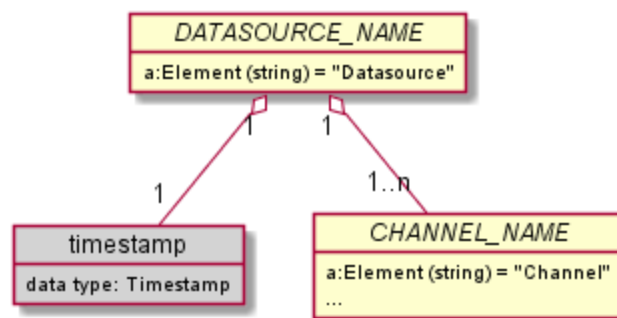


Figure 11: Datasource group overview

1.9.19.1 Attributes The following attribute is assigned to the group:

Name	Data type	Parent object	Optional	Description
Element	string	DATASOURCE_NAME	no	Contains the type of the group, this is fix Datasource

1.9.19.2 Example In our example the name of the data source group is assigned, which should contain our environmental measurement data, which we call [ENVIRONMENT](#).

1.9.19.3 Timestamp data set Each data source group contains a data set called [timestamp](#). It contains all timestamps at which a measurement was recorded. The size of this list of timestamps is the same as the size of the datasets per channel.

A more detailed description can be found in the chapter [Timestamp Array!](#)

1.9.20 Channel

Channel is of type 'Group' in HDF5.

A channel group contains metadata for the actual measurement data available in the channel. The naming can be freely selected, but must be unique within the data source group. Some attributes as TriggerMode are the same for each channel of the same [DATASOURCE](#). This is done intentionally to have all attributes in the group [CHANNEL](#) without the need of the attributes of the parent object.

Name	Parent object	Optional
CHANNEL_NAME	DATASOURCE_NAME	no

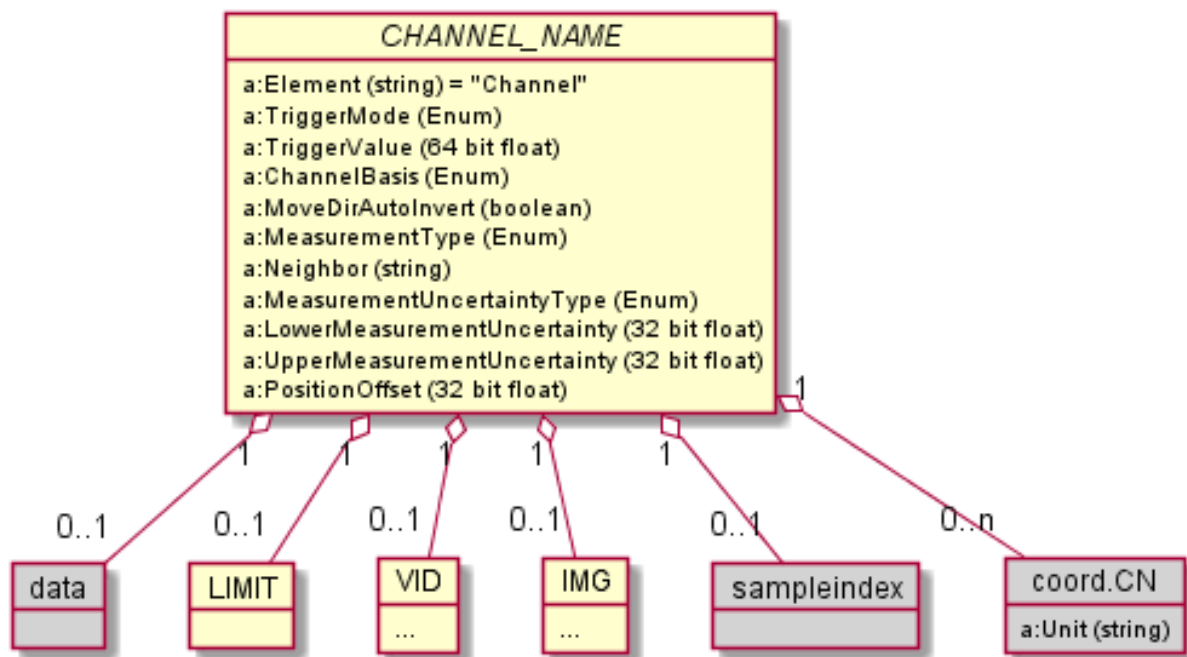


Figure 12: Channel group overview

The following attributes are contained in this group:

Name	Data type	Parent object	Optional	Description
TriggerMode	Enum	CHANNEL_NAME	no	See chapter Trigger Mode
CommonTriggerDistance	32 bit float	CHANNEL_NAME	no	See below
CommonTriggerFrequency	32 bit float	CHANNEL_NAME	no	See below
ChannelBasis	Enum	CHANNEL_NAME	no	See below
MoveDirAutoInvert	boolean	CHANNEL_NAME	yes	See below
MeasurementType	Enum	CHANNEL_NAME	no	See below
Neighbor	string	CHANNEL_NAME	no	See below
MeasurementUncertaintyType	Enum	CHANNEL_NAME	yes	See below
LowerMeasurementUncertainty	32 bit float	CHANNEL_NAME	yes	See below
UpperMeasurementUncertainty	32 bit float	CHANNEL_NAME	yes	See below

Name	Data type	Parent object	Optional	Description
PositionOffset	32 bit signed float	CHANNEL_NAME	no	See below
Element	string	CHANNEL_NAME	no	Contains the type of the group, this is fix Channel

CommonTriggerDistance

Data is collected every [value](#) meters.

In a channel group there is always only one common trigger value! The other trigger value is set to “0.0”

Both trigger values have the value “0.0” if [TriggerMode](#) contains the value [EVENTS](#).

CommonTriggerFrequency

Data is collected with a frequency of [value](#) Hertz.

In a channel group there is always only one common trigger value! The other trigger value is set to “0.0”

Both trigger values have the value “0.0” if [TriggerMode](#) contains the value [EVENTS](#).

ChannelBasis

Defines the applicable side of the data signal in relation to the vehicle or moving direction. A sensor can be installed on one side of the platform. Depending on the direction and the orientation of the vehicle, this can be ‘left’ or ‘right’ in relation to the movement but is always fixed in relation to the vehicle frame axes.

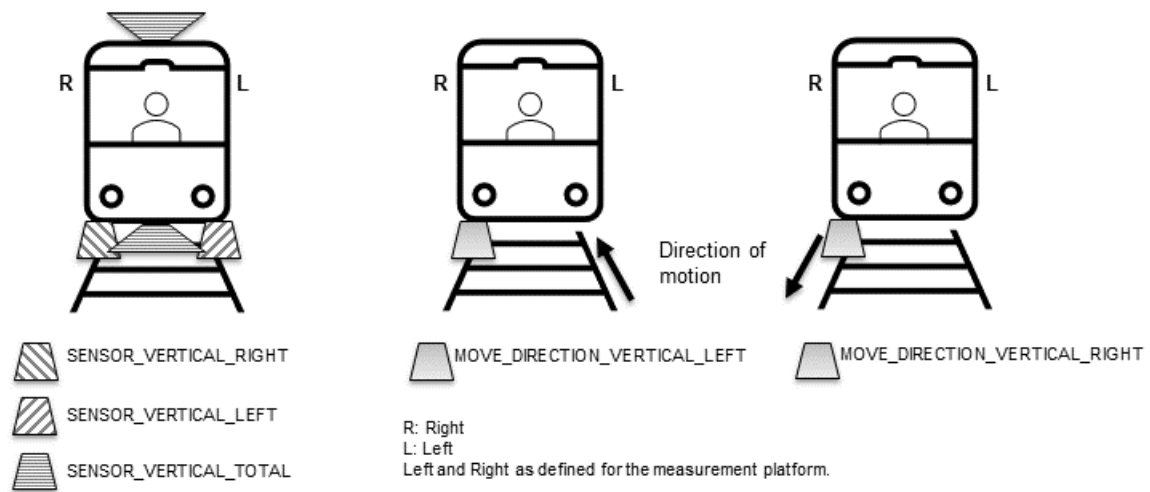


Figure 13: Explanation of Sensor location and detection

Possible values are:

Value	Description
TOTAL	The channel has no dependency on direction of motion or vehicle orientation
SENSOR_LEFT	Deprecated – same as SENSOR_VERTICAL_LEFT
SENSOR_RIGHT	Deprecated – same as SENSOR_VERTICAL_RIGHT
RAIL_LEFT	Deprecated – same as MOVE_DIRECTION_VERTICAL_LEFT
RAIL_RIGHT	Deprecated – same as MOVE_DIRECTION_VERTICAL_RIGHT
SENSOR_VERTICAL_LEFT	The sensor is located on the left-hand side of the measurement platform and detects signals in a vertical plane, providing a top-down or bottom-up view

Value	Description
SENSOR_VERTICAL_RIGHT	The sensor is located on the right-hand side of the measurement platform and detects signals in a vertical plane, providing a top-down or bottom-up view
MOVE_DIRECTION_VERTICAL_LEFT	The sensor detects signals on the left-hand side with respect to the direction of motion, irrespective of its orientation
MOVE_DIRECTION_VERTICAL_RIGHT	The sensor detects signals on the right-hand side with respect to the direction of motion, irrespective of its orientation
SENSOR_VERTICAL_TOTAL	The sensor is located at the center of the measurement platform and detects signals in a vertical plane, providing a top-down or bottom-up view
MOVE_DIRECTION_VERTICAL_TOTAL	The sensor detects signals in the center with respect to the direction of motion, irrespective of its orientation

MoveDirAutoInvert

Flag that implies, if set to true, that measurement data of this channel is dependent on direction and should be inverted if viewed with a section of different direction.

> If the flag is set to false, not set or only data of a single direction is viewed, then no inversion is needed when viewing data of this channel.

MeasurementType

Defines how a value was created. This can be measured, calculated or taken from a previously defined data source that was read from there and inserted into the file.

The following values are possible:

Value	Description
MEASURED_VALUE	Measured values
REFERENCE_VALUE	A setpoint of a third source

An example for reference values are defined target values which flow in from another source (file, database etc.) and are to be used for comparisons.

Neighbor

Refers to the name of an adjacent channel. This can be the right rail, for example, when measuring the track temperature of the left rail. Thus the attribute `neighbor` of the channel “TEMP_RAIL_L” would contain the name “TEMP_RAIL_R” and vice versa.

MeasurementUncertaintyType

Defines the type in which the measurement uncertainty is set, namely absolute (in which case it has the same unit as the channel itself), or relative (in which case it has no unit). The following values are thus allowed:

Value	Description
ABSOLUTE	Absolute values
RELATIVE	Relative values

This attribute is optional. If it is not set, no information is known about the measurement uncertainty.

LowerMeasurementUncertainty

This attribute contains the measurement accuracy of the channel according to the specifications of the measurement system. This is the lower bound for the measurement uncertainty range. This attribute is optional. If it is not set, no information is known about the measurement uncertainty.

UpperMeasurementUncertainty

This attribute contains the measurement accuracy of the channel according to the specifications of the measurement system. This is the upper bound for the measurement uncertainty range. This attribute is optional. If it is not set, no information is known about the measurement uncertainty.

PositionOffset

Describes the distance between a defined zero point (position) on the measuring vehicle and a the measuring sensor. This specification is used to convert the exact time at which the measurement was taken to a defined zero point. The value can be positive or negative and has the unit millimeter.

1.9.20.1 `TriggerMode` This attribute defines how the data was recorded.

Possible values are:

Value	Description	Unit
TIME	Time-based measurement data recording	Nanoseconds
DISTANCE	Distance-based measurement data acquisition	Millimeter
EVENT	EVENT based recording	none

1.9.20.2 Data object Each channel group receives a data set with the actual measurement data:

Name	HDF5 Type	Optional
data	HDF5 Dataset	no

There are as many measurement data entries as there are timestamps in the data set `timestamp` which is included in the channel group.

The data set needs more information, this is given as attributes:

Name	Data type	Parent object	Optional	Storage type
Unit	string	CHANNEL_NAME	no	<code>Array</code>

Unit:

The physical unit of the measurement data, such as “millimeter”. If no physical unit can be assigned to the data, this attribute remains empty.

The data set and the possible data that can be stored are described in more detail in the chapter [Dataset](#).

1.9.21 Measurement mode

Measurement mode is of type ‘Group’ in HDF5.

The following group contains important information about the measurement mode of the system. The `MEASUREMENTMODE` is depending on the operational state of the measurement system, for example “productive”.

Name	Parent object	Optional
MEASUREMENTMODE	MEASURINGSYSTEM_NAME	yes

The group **MEASUREMENTMODE** contains one attribute:

Name	Data type	Parent object	Optional	Storage type
Element	string	MEASUREMENTMODE	no	Equals to “Measurement-Mode”. Identifies this node.

The group **MEASUREMENTMODE** contains four datasets:

Name	Data type	Parent object	Optional	Storage type
mode	Enum	MEASUREMENTMODE	no	Array
timestampfrom	64 bit integer	MEASUREMENTMODE	no	Array
timestampto	64 bit integer	MEASUREMENTMODE	no	Array
info	string	MEASUREMENTMODE	yes	Array

All these datasets represent different columns in a table. Their sizes will therefore always be identical.

1.9.21.1 Mode (enum) There are three different measurement modes, which are explained individually below.

Name	Description
PRODUCTIVE	Productive data that will be further used.
TEST	Test data recorded during a diagnostic run with the aim of checking and testing the measuring equipment.

Name	Description
SIMULATION	Simulated values that the measuring systems produce themselves and are no longer used.

1.9.22 Logging

Logging is of type 'Group' in HDF5.

The logging group contains information about the status of the measuring systems. The data is divided into two subgroups, [AVAILABILITY](#) and [MESSAGES](#). These are described in separate chapters.

Name	Parent object	Optional
LOGGING	MEASURINGSYSTEM_NAME	yes

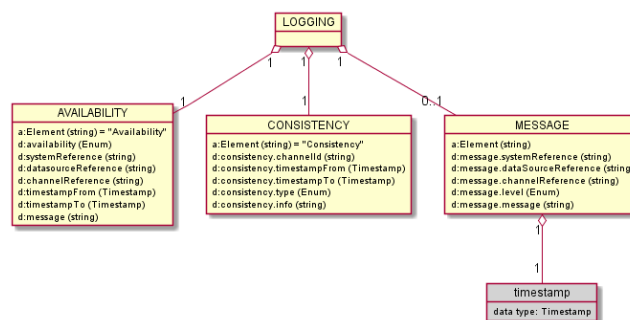


Figure 14: Logging group overview

1.9.22.1 Availability

Availability is of type 'Group' in HDF5.

In this group, failures and interruptions of measurement systems are recorded in a defined structure, each as its own data set.

Name	Parent object	Optional
AVAILABILITY	LOGGING	yes

The following datasets are included in this group:

Name	Data type	Parent object	Optional	Storage type
Element	string	AVAILABILITY	no	Equals to “Availability”. Identifies this node.
availability	Enum	AVAILABILITY	no	Array
systemreference	string	AVAILABILITY	no	Array
datasourcereference	string	AVAILABILITY	yes	Array
channelreference	string	AVAILABILITY	yes	Array
timestampfrom	Timestamp	AVAILABILITY	no	Array
timestampto	Timestamp	AVAILABILITY	no	Array
message	string	AVAILABILITY	no	Array

availability

See chapter [Availability](#)

systemreference

A reference to the measurement system.

datasourcereference

A reference to a data source.

channelreference

A reference to a channel.

message

This data set contains one message per entry about a failure of a measuring instrument.

timestampfrom

Start timestamp of the availability entry.

timestampto

End timestamp of the availability entry.

1.9.22.1.1 Availability Defines the severity of the failure or interruption of a measurement system. The following values are possible:

level	description
MONITORED	The measurement system or sensor was monitored during this period. All other
FAILURE	The measuring system or sensor has failed and has not recorded any measured values during the session.
WARNING	The measuring system or the sensor may have had a failure and the measured values may not be correct.

1.9.22.2 Consistency Consistency is of type 'Group' in HDF5.

The message about the consistency of the data is triggered by a system that checks all data according to certain criteria. For example, this could be a check for black images in a video. If all frames in the video are black, something is wrong and the video is unusable.

Name	Parent object	Optional
CONSISTENCY	LOGGING	yes

The group [CONSISTENCY](#) contains one dataset:

Name	Data type	Parent object	Optional	Storage type
Element	string	CONSISTENCY	no	Equals to "Consistency". Identifies this node.
channelid	string	CONSISTENCY	no	Array
timestampfrom	64 bit integer	CONSISTENCY	no	Array
timestampto	64 bit integer	CONSISTENCY	no	Array
consistencytype	Enum	CONSISTENCY	no	Array
consistencyinfo	string	CONSISTENCY	yes	Array

All these datasets represent different columns in a table. Their sizes will therefore always be identical.

channelid

Channel reference in which the finding was detected.

timestampfrom

Start timestamp of the consistency.

timestampto

End timestamp of the consistency. Note that consistencies can never overlap and will always be continuous (without holes) within sections.

consistencytype

The consistency type can have the following values:

Name	Description
CONSISTENT	The referenced data has been checked and is consistent (OK: green)
INCONSISTENT	The referenced data has been checked and is inconsistent (not OK: red)
WARNING	The referenced data has been checked and set to warning (undefined: orange)
EXCLUDED	The referenced data has been checked and set to excluded by definition
NO_DATA	Consistency has been checked, but no data was found

consistencyinfo

Contains additional information about this consistency (E.g. which rule decided the consistency type).

1.9.22.3 Message Message is of type 'Group' in HDF5.

This group contains messages, generated from a measurement system or a person, structured in data set.

Name	Parent object	Optional
MESSAGES	LOGGING	yes

The following datasets are included in this group:

Name	Data type	Parent object	Optional	Storage type
Element	string	MESSAGES	no	Equals to “Messages”. Identifies this node.
message	string	MESSAGES	no	Array
level	Enum	MESSAGES	no	Array
systemreference	string	MESSAGES	no	Array
datasourcereference	string	MESSAGES	no	Array
channelreference	string	MESSAGES	no	Array
timestamp	Timestamp	MESSAGES	no	Array

message

This data set contains one message per entry about a failure of a measuring instrument.

systemreference

A reference to the measurement system.

datasourcereference

A reference to a data source.

channelreference

A reference to a channel.

level

See chapter [Message level](#)

1.9.22.3.1 Message level Defines the importance of a message. Following values are possible:

level	description
INFO	The message is only informative.
WARNING	The message indicates a warning.
ERROR	The message indicates an error.

1.9.23 Position

Position is of type 'Group' in HDF5.

This group contains general information on the position. It is like a measuring system (see [Measuring System](#)) with following differences:

The channel with the position data has multiple (nine) datasets. This is because a position record always consists of these nine values and therefore this dataset is hardcoded, instead of configurable as for all other measurement systems.

The channel and datasets have no attributes (e.g., physical unit) because this information is part of the file format and not configurable.

This group has dependencies to the group [TOPOLOGY](#).

Name	Parent object	Optional
POSITION	SESSION	no

1.9.23.1 Position datasource See chapter [Datasource](#)

Name	Parent object	Optional
POSITION.SOURCE	POSITION	no

1.9.23.2 Position channel

Name	Parent object	Optional
POSITION.SOURCE.DATA	POSITION.SOURCE	no

1.9.23.2.1 Position channel datasets

Name	Data type	Parent object	Optional	Storage type
covereddistance	64 bit float	POSITION.SOURCE.DATA	no	Array
direction	Enum	POSITION.SOURCE.DATA	no	Array
kilometrage	Enum	POSITION.SOURCE.DATA	no	Array
track_id	32 bit integer	POSITION.SOURCE.DATA	no	Array
line_id	32 bit integer	POSITION.SOURCE.DATA	no	Array
trackoffset	64 bit float	POSITION.SOURCE.DATA	no	Array
lineoffset	64 bit float	POSITION.SOURCE.DATA	no	Array
positionaccuracy	32 bit float	POSITION.SOURCE.DATA	no	Array
positionquality	32 bit integer	POSITION.SOURCE.DATA	no	Array

covereddistance

The distance (in [m]) the vehicle is driven at the time of recording within the current session.

track_id

The ID of the track on which the vehicle is located at the time of recording. The value -1 indicates that no position could be determined.

line_id

The ID of the line on which the vehicle is located at the time of recording. The value -1 indicates that no position could be determined.

trackoffset

The offset (in [m]) on the track where the vehicle is located at the time of recording.

lineoffset

The line kilometer (in [km]) on the line where the vehicle is located at the time of recording.

positionquality

Marker for the quality of the position determination ranging from 0 meaning “very good” to 15 for “very poor”.

positionaccuracy

The position accuracy (in [m]) of the current position.

1.9.23.2.2 Coach Direction The data set [direction](#) contains the coach orientation of the vehicle. This information influences the position of the measuring systems.

This data set can contain the following values:

Value	Description
FORWARD	Vehicle moving forward
BACKWARD	Vehicle moving backward

1.9.23.2.3 Kilometrage The data set [kilometrage](#) contains the alignment of the track depending on the movement of the vehicle on the track (kilometrage decreasing/increasing). This information serves the correct evaluation of the kilometer data of the line, see data set [trackoffset](#).

This data set can contain the following values:

The following values are allowed:

Value	Meaning
INCREASING	Increasing track kilometrage on the track
DECREASING	Decreasing track kilometrage on the track

1.9.24 EVENTS

Events is of type 'Group' in HDF5.

The EVENTS group is used to store events that occurred during the recording of data. Events are bound to a channel, system or session and have a link to it. Systems can, for example, trigger an EVENT when a limit value is exceeded. Events are always time-bound which means an EVENT contains the exact time of occurrence and the duration of the EVENT. The duration can also be zero, so the EVENT occurred exactly at the specified time.

Name	HDF5 Type	Parent object	Optional
EVENTS	HDF5 Group	SESSION_NAME	yes

Within the group there are the following data fields:

Name	Data type	Parent object	Optional	Storage type
Element	string	EVENTS	no	Equals to "Events". Identifies this node.
systemref	string	EVENTS	no	Array
channelref	string	EVENTS	yes	Array
eventtype	string	EVENTS	no	Array
data	string	EVENTS	no	Array
duration	64 bit signed integer	EVENTS	no	Array
timestamp	Timestamp	EVENTS	no	Array

Each of these datasets contains a list with information about an entry, at a certain time. Each data set is described in more detail in the following subchapters.

systemref Contains a list of entries containing the name of the system that triggered the EVENT.

Name	Data type	Parent Object	Optional	Storage Type
systemref	string	EVENTS	no	Array

channelref Contains a list of entries that refers to a channel to which the EVENT applies.

Name	Data type	Parent Object	Optional	Storage Type
channelref	string	EVENTS	yes	Array

data This data set contains the actual information about an EVENT, this in the XML notation which is described in more detail in each chapter of the EVENT types (Defect EVENT type, Detected object EVENT type, Limit EVENT type).

A type can be stored for each EVENT. These are explained in more detail below.

Name	Data type	Parent object	Optional	Storage type
data	string	EVENTS	no	Array

duration Defines for each EVENT the duration of the EVENT itself. This value can also be zero.

Name	Data type	Parent object	Optional	Storage type
duration	64 bit integer	EVENTS	no	Array

type Contains the type of an EVENT.

Name	Data type	Parent object	Optional	Storage type
type	string	EVENTS	no	Array

In the list “type” the type of the recorded EVENT is shown. The different types contain different information which is shown in the following subchapters. There are corresponding XML schemas for all types that define the technical specifications.

1.9.24.1 Defect EVENT type This event contains information about elements found in the 2-dimensional data (for example a picture) that exceed defined thresholds. A defect can be, for example, an image of a rail showing a damage of the surface. This defect is recorded by a system. However, it may happen that this error is not one (incorrectly detected), this information can be specified afterwards (attribute “PossibleValidationResults”). Defects are always channel bound and

recorded or evaluated by a system. In the following, the elements and attributes that occur in a Defect as XML are described in more detail.

The XML Schema can be found in the chapter [EventsDefect](#).

1.9.24.1.1 XML elements Not all of these elements must be present, details can be taken from the XML Schema.

Name	Description	Parent object
Defect	XML Root Element	none
PossibleDefectNames	Name of a possible error	Defect
PossibleClassifications	Classification of a possible defect	Defect
PossibleValidationResults	Possible confirmations of the defect	Defect

1.9.24.1.2 XML attributes Below are the attributes of the root element “Defect”:

Name	Description	Parent object
Classification	Classification of the error	Defect
DefectName	Name of the error	Defect
Details	Further information or more detailed descriptions of the error	Defect
Parameter1Name	Name of the parameter 1	Defect
Parameter1Value	Value of parameter 1	Defect
Parameter2Name	Name of the parameter 2	Defect
Parameter2Value	Value of Parameter 2	Defect
Parameter3Name	Name of the parameter 3	Defect
Parameter3Value	Value of parameter 3	Defect
ID	Unique number for identification of the error	Defect

1.9.24.2 Detected object EVENT type These events contain information about detected infrastructure objects. These can be, for example, detected balises or tunnels. What exactly counts as a found object is not defined in this specification, only the information for a recorded EVENT.

The XML Schema can be found in the chapter [EventsGeneric](#).

1.9.24.2.1 XML elements Not all of these elements must be present, details can be taken from the XML schema.

Name	Description	Parent object
DetectedObject	Root Element	none
object	Element with information about the found object in the element itself or in the attributes	DetectedObject
Reference	Reference to a list of known and uniquely assignable objects of the railway company	DetectedObject
ObjectAttribute	Further information about the object, the information is contained in the attributes	DetectedObject

1.9.24.2.2 XML attributes

Name	Description	Parent object
Unique ID of the EVENT	DetectedObject	
Type	Type of object found	object
Description	further description or information about/from object	object
ObjectConsistency	Reference to the correctness of the specified data	object
ReferenceSystem	Reference to the name of the system from which the data originates	Reference
Key	Information about the data contained in the “ObjectAttribute” element	ObjectAttribute

1.9.24.3 Limit violation EVENT type A limit violation is an event where a 1-dimensional channel (for example a measurement signal) exceeds its threshold.

Limit value exceedances of measured values of a channel can also be recorded as events.

The XML schema can be found in chapter [Events Generic](#).

1.9.24.3.1 XML elements

Name	Description	Parent object
LimitViolation	Root Element	none

1.9.24.3.2 XML attributes

Name	Description	Parent object
TimestampMaxViolation	Time at which limit value was exceeded	LimitViolation
ViolatedLimit	Name of the defined limit	LimitViolation
ID	Unique ID of the EVENTS	LimitViolation

1.9.24.4 Comment Comments recorded during a diagnostic drive by the user. The content is not specified, only the XML structure. The XML schema can be found in chapter [Events Comment](#).

1.9.24.4.1 XML elements

Name	Description	Parent object
Comment	Root element and message, recorded by the user	none

1.9.24.4.2 XML attributes

Name	Description	Parent object
Username	Name of the user who recorded the message	Comment
ID	Unique ID of this message	Comment

1.9.24.5 Corrupt Messages of the type “damaged” or “unusable” do not receive a content specification, only the XML structure is predefined and described here. The XML schema can be found in chapter [Events Generic](#).

1.9.24.5.1 XML elements

Name	Description	Parent object
Corrupt	Root element and message, recorded by the user	none

1.9.24.5.2 XML attributes

Name	Description	Parent object
Username	Name of the user who recorded the message	Corrupt
ID	Unique ID of this message	Corrupt

1.9.25 Measurement system configuration

Measurement system configuration is of type 'Group' in HDF5.

The structure of this group is the same as for the configuration group below the group SESSION_NAME: **Session configuration**, but the parent group is *MEASURINGSYSTEM_NAME*: **Measuring system**. The measuring system configuration group also does not contain the Topology.

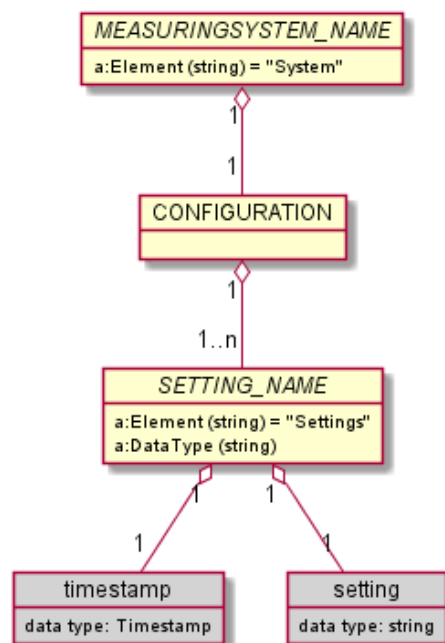


Figure 15: Measurement configuration group overview

1.10 XML Schema Definitions

1.10.1 Events Comment

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
3   xmlns:tns="http://www.sbb.ch/RCMDX/Events/Comment"
4   targetNamespace="http://www.sbb.ch/RCMDX/Events/Comment"
5   elementFormDefault="qualified">
6
7   <xs:include schemaLocation="../../../RcmDxDATATypes.xsd" />
8
9   <xs:element name="Comment">
10     <xs:complexType>
11       <xs:simpleContent>
12         <xs:extension base="xs:string">
13           <xs:attribute name="Username" type="xs:string" use="required"
14             />
15           <xs:attribute name="ID" type="tns:UUID" use="required" />
16         </xs:extension>
17       </xs:simpleContent>
18     </xs:complexType>
19   </xs:element>
20 </xs:schema>
```

1.10.2 Events Defect

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
3   xmlns:tns="http://www.sbb.ch/RCMDX/Events/Defect"
4   targetNamespace="http://www.sbb.ch/RCMDX/Events/Defect"
5   elementFormDefault="qualified">
6
7   <xs:include schemaLocation="../../../RcmDxDATATypes.xsd" />
8
9   <xs:element name="Defect">
10     <xs:complexType>
11       <xs:sequence>
12         <xs:element name="PossibleDefectNames" type="xs:string"
13           minOccurs="0" maxOccurs="unbounded" />
14         <xs:element name="PossibleClassifications" type="xs:string"
15           minOccurs="0" maxOccurs="unbounded" />
16         <xs:element name="PossibleValidationResults" type="xs:string"
17           minOccurs="0" maxOccurs="unbounded" />
18       </xs:sequence>
19       <xs:attribute name="Classification" type="xs:string" use="
20         required" />
21     </xs:complexType>
22   </xs:element>
23 </xs:schema>
```

```
17     <xs:attribute name="DefectName" type="xs:string" use="required"
18         />
19     <xs:attribute name="Details" type="xs:string" use="required" />
20     <xs:attribute name="Parameter1Name" type="xs:string" />
21     <xs:attribute name="Parameter1Value" type="xs:string" />
22     <xs:attribute name="Parameter2Name" type="xs:string" />
23     <xs:attribute name="Parameter2Value" type="xs:string" />
24     <xs:attribute name="Parameter3Name" type="xs:string" />
25     <xs:attribute name="Parameter3Value" type="xs:string" />
26     <xs:attribute name="ID" type="tns:UUID" use="required" />
27 </xs:complexType>
28 </xs:element>
29 </xs:schema>
```

1.10.3 Events Generic

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
3   xmlns:tns="http://www.sbb.ch/RCMDX/Events/Generic"
4   targetNamespace="http://www.sbb.ch/RCMDX/Events/Generic"
5   elementFormDefault="qualified">
6
7   <xs:include schemaLocation="../../../RcmDxDatatypes.xsd" />
8
9   <xs:element name="Corrupt">
10     <xs:complexType>
11       <xs:simpleContent>
12         <xs:extension base="xs:string">
13           <xs:attribute name="Username" type="xs:string" use="required"
14             />
15           <xs:attribute name="ID" type="tns:UUID" use="required" />
16         </xs:extension>
17       </xs:simpleContent>
18     </xs:complexType>
19   </xs:element>
20
21   <xs:simpleType name="ObjectConsistencyXml">
22     <xs:restriction base="xs:string">
23       <xs:enumeration value="Ok" />
24       <xs:enumeration value="OnlyInReal" />
25       <xs:enumeration value="OnlyInData" />
26       <xs:enumeration value="Measured" />
27     </xs:restriction>
28   </xs:simpleType>
29
30   <xs:element name="DetectedObject">
31     <xs:complexType>
32       <xs:sequence>
33         <xs:element name="object" minOccurs="1" maxOccurs="1">
```

```
33         <xs:complexType>
34             <xs:simpleContent>
35                 <xs:extension base="xs:string">
36                     <xs:attribute name="Type" type="xs:string" use="
37                         required" />
38                     <xs:attribute name="Description" type="xs:string" use="
39                         required" />
40                     <xs:attribute name="ObjectConsistency" type="tns:
41                         ObjectConsistencyXml" use="required" />
42                 </xs:extension>
43             </xs:simpleContent>
44         </xs:complexType>
45     </xs:element>
46     <xs:element name="Reference" minOccurs="0" maxOccurs="
47         unbounded">
48         <xs:complexType>
49             <xs:simpleContent>
50                 <xs:extension base="xs:string">
51                     <xs:attribute name="ReferenceSystem" type="xs:string"
52                         use="required" />
53                 </xs:extension>
54             </xs:simpleContent>
55         </xs:complexType>
56     </xs:element>
57     <xs:element name="ObjectAttribute" minOccurs="0" maxOccurs="
58         unbounded">
59         <xs:complexType>
60             <xs:simpleContent>
61                 <xs:extension base="xs:string">
62                     <xs:attribute name="Key" type="xs:string" use="
63                         required" />
64                 </xs:extension>
65             </xs:simpleContent>
66         </xs:complexType>
67     </xs:element>
68 </xs:sequence>
69     <xs:attribute name="ID" type="tns:UUID" use="required" />
70 </xs:complexType>
71 </xs:element>
72
73 <xs:element name="LimitViolation">
74     <xs:complexType>
75         <xs:attribute name="TimestampMaxViolation" type="xs:long" use="
76             required" />
77         <xs:attribute name="ViolatedLimit" type="xs:string" use="
78             required" />
79         <xs:attribute name="ID" type="tns:UUID" use="required" />
80     </xs:complexType>
81 </xs:element>
82
83 </xs:schema>
```

1.10.4 RCM-DX Data types

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
   elementFormDefault="qualified">
3   <xs:simpleType name="restrictedString">
4     <xs:restriction base="xs:string">
5       <xs:minLength value="1" />
6       <xs:maxLength value="512" />
7       <xs:pattern value="[a-zA-Z0-9_\-\.\.]" />
8     </xs:restriction>
9   </xs:simpleType>
10  <xs:simpleType name="restrictedStringWithColon">
11    <xs:restriction base="xs:string">
12      <xs:minLength value="1" />
13      <xs:maxLength value="512" />
14      <xs:pattern value="[a-zA-Z0-9_\-\.\.]" />
15    </xs:restriction>
16  </xs:simpleType>
17  <xs:simpleType name="restrictedID">
18    <xs:restriction base="xs:ID">
19      <xs:minLength value="1" />
20      <xs:maxLength value="512" />
21      <xs:pattern value="[a-zA-Z0-9_\-\.\.]" />
22    </xs:restriction>
23  </xs:simpleType>
24  <xs:simpleType name="restrictedIDREF">
25    <xs:restriction base="xs:IDREF">
26      <xs:minLength value="1" />
27      <xs:maxLength value="512" />
28      <xs:pattern value="[a-zA-Z0-9_\-\.\.]" />
29    </xs:restriction>
30  </xs:simpleType>
31  <xs:simpleType name="versionString">
32    <xs:restriction base="xs:string">
33      <xs:minLength value="1" />
34      <xs:maxLength value="32" />
35      <xs:pattern value="[a-zA-Z0-9_\-\.\.]" />
36    </xs:restriction>
37  </xs:simpleType>
38  <xs:simpleType name="portNumber">
39    <xs:restriction base="xs:int">
40      <xs:minExclusive value="0" />
41      <xs:maxInclusive value="65535" />
42    </xs:restriction>
43  </xs:simpleType>
44  <xs:simpleType name="ipAddress">
45    <xs:restriction base="xs:string">
46      <xs:pattern value="([0-9]{1,3}\.){3}[0-9]{1,3}" />
47    </xs:restriction>
48  </xs:simpleType>
```

```
49 <xs:simpleType name="network">
50   <xs:restriction base="xs:string">
51     <xs:pattern value="([0-9]{1,3}\.){3}[0-9]{1,3})/[0-9]{1,2}" />
52   </xs:restriction>
53 </xs:simpleType>
54 <xs:simpleType name="hostName">
55   <xs:restriction base="xs:string">
56     <xs:pattern value="([a-zA-Z0-9]|[a-zA-Z0-9][a-zA-Z0-9\-\]*[a-zA-Z0-9])\.)*([A-Za-z0-9]|[A-Za-z0-9][A-Za-z0-9\-\]*[A-Za-z0-9])" />
57   </xs:restriction>
58 </xs:simpleType>
59 <xs:simpleType name="ipAddressOrHostName">
60   <xs:union memberTypes="ipAddress hostName" />
61 </xs:simpleType>
62 <xs:simpleType name="nonNegativeInt">
63   <xs:restriction base="xs:int">
64     <xs:minInclusive value="0" />
65   </xs:restriction>
66 </xs:simpleType>
67 <xs:simpleType name="positiveInt">
68   <xs:restriction base="xs:int">
69     <xs:minInclusive value="1" />
70   </xs:restriction>
71 </xs:simpleType>
72 <xs:simpleType name="positiveFloat">
73   <xs:restriction base="xs:float">
74     <xs:minExclusive value="0" />
75   </xs:restriction>
76 </xs:simpleType>
77 <xs:simpleType name="positiveIntOrMinus1">
78   <xs:restriction base="xs:int">
79     <xs:minInclusive value="-1" />
80   </xs:restriction>
81 </xs:simpleType>
82 <xs:simpleType name="positiveLong">
83   <xs:restriction base="xs:long">
84     <xs:minExclusive value="0" />
85   </xs:restriction>
86 </xs:simpleType>
87 <xs:simpleType name="nonNegativeLong">
88   <xs:restriction base="xs:long">
89     <xs:minInclusive value="0" />
90   </xs:restriction>
91 </xs:simpleType>
92 <xs:simpleType name="compressionLevel">
93   <xs:restriction base="xs:integer">
94     <xs:minInclusive value="0" />
95     <xs:maxInclusive value="9" />
96   </xs:restriction>
97 </xs:simpleType>
```

```
98     <xs:simpleType name="mimeType">
99         <xs:restriction base="xs:string">
100             <xs:pattern value="![#$%'+\-\0-9A-Z\^_'a-z{|}~]+/![#$%'+\-\0-9A-
                Z\^_'a-z{|}~]+(; *[^;]+)*" />
101         </xs:restriction>
102     </xs:simpleType>
103     <xs:simpleType name="nonEmptyString">
104         <xs:restriction base="xs:string">
105             <xs:minLength value="1" />
106         </xs:restriction>
107     </xs:simpleType>
108     <xs:simpleType name="UUID">
109         <xs:restriction base="xs:string">
110             <xs:pattern
111                 value="(urn:uuid:)?[0-9a-fA-F]{8}-[0-9a-fA-F]{4}-[0-9a-fA-F]
                    ]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{12}|\{[0-9a-fA-F]{8}-[0-9a-fA-F]
                    ]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{12}\}" />
112             </xs:restriction>
113         </xs:simpleType>
114     <xs:simpleType name="vehicleNumber">
115         <xs:restriction base="xs:string">
116             <xs:pattern value="[0-9]{2} [0-9]{2} [0-9]{4} [0-9]{3}-[0-9]" />
117         </xs:restriction>
118     </xs:simpleType>
119     <xs:simpleType name="httpUrl">
120         <xs:restriction base="xs:anyURI">
121             <xs:pattern value="https?:/.+." />
122         </xs:restriction>
123     </xs:simpleType>
124 </xs:schema>
```