Specifications for ISO5436-2 XML File Format

Dr. Georg Wiora

NanoFocus AG

Revision 110

24. January 2008

Abstract

This document describes the structure of profile data encoded in an ISO5436-2 XML file. Further it defines the format of binary encoded data and best practice for the use of the data format.

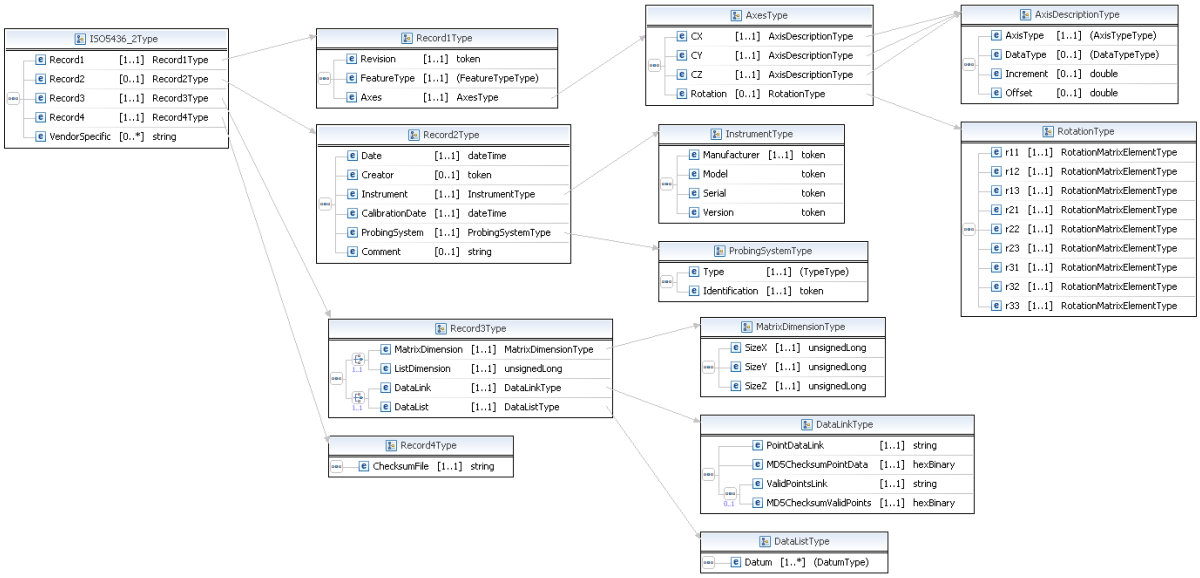

Figure 1: Structure of the ISO5436-2 XML description.

File Extension

The default file extension of ISO5436-2 data container files is `".X3P"` what can be remembered as "**X**ML **3**-D Surface **P**rofile".

Directory Structure of Container

The container's directory structure consists of a central document named `"main.xml"`. The main document is an XML-file as defined in `ISO5436-2.XSD`. For a simple document the container comprises only `"main.xml"`. More complex documents, with larger amounts of data, can store the actual measurement data in a binary file. A link in `"main.xml"` points to the binary file. Binary files are stored in a subdirectory named `"bindata"`. The name of the binary file can be chosen freely. A recommended name is `"data.bin"`. The relative URI pointing to the binary file would in the default case look like `"bindata/data.bin"`.

To mark valid and invalid points in a binary file an optional matrix with the recommended name `"bindata/valid.bin"` can be stored. This file contains a packed array of Boolean values (a bit field). The value `True` (bit value 1) is associated with a valid point.

If the validity file does not exist all points are valid that do not have the special floating point value `NaN` (Not a Number).

Format of Binary Files

The binary file is a plain matrix of data points, written in binary format. The matrix has three dimensions. Each data point contains one ore more coordinates in X,Y,Z sequence and types specified in `<DataType>` in `Record1` in `"main.xml"`.

Matrix Structure

The matrix always has three dimensions. By definition the matrix has to resemble the topological neighbourhood of the data points in the 3D-space. Clearly said two points that are neighbours in 3D have to be placed in neighbouring cells of the matrix grid.

Matrix indexing

The three dimensions of the matrix correspond to the three coordinate axes of the measured profile data. The first dimension corresponds to the x-axis, the second to the y-axis, and the third to the z-axis. The x-index is the fastest index y the next slower and z the slowest index. For a matrix of x-y-z-dimensions `<Matrix>3,3,2</Matrix>` the cell order of the points `P(x,y,z)` looks like the following example:

```
P(1,1,1), P(2,1,1), P(3,1,1),
P(1,2,1), P(2,2,1), P(3,2,1),
P(1,3,1), P(2,3,1), P(3,3,1),
P(1,1,2), P(2,1,2), P(3,1,2),
P(1,2,2), P(2,2,2), P(3,2,2),
P(1,3,2), P(2,3,2), P(3,3,2)
```

The indexing is used for binary files as well as for xml-data in the `<DataList>` tag.

Data Types

Possible data types are:

- `"I"`: 16 bit signed integer in Intel byte order that is low byte (LSB) first.
- `"L"`: 32 bit signed integer in Intel byte order that is LSB, 2nd SB, 3rd SB, MSB.
- `"F"`: 32 bit IEEE 754 floating point number
- `"D"`: 64 bit IEEE 754 floating point number

Data Order

Each cell can contain an X, Y and Z coordinate in this order. The X and Y coordinate are optional for points stored in a `<DataMatrix>`, depending on the types of axis defined. For an incremental X or Y axis, no X and Y-coordinates are written to the `DataMatrix` because the X- and Y- indices are defined by the matrix indices of a point. A few examples show how this can look like.

Example 1

A very typical simple case is, when we have only one set of z-coordinates: The XML file looks like this:

```
<Record1>
  <CX>
    <Axis><AxisType>I</Axistype><DataType>L</DataType> … </Axis>
  </CX>
  <CY>
    <Axis><AxisType>I</Axistype><DataType>L</DataType> … </Axis>
  </CY>
  <CZ>
    <Axis><AxisType>A</Axistype><DataType>F</DataType> … </Axis>
  </CZ>
</Record1>
```

That means that each matrix element contains only one z-coordinate. The x and y coordinates are defined incrementally by the matrix position of the point using the `<Axis>` definition for an incremental axis.

Example 2

A medium complex example uses three absolute axes:

```
<Record1>
  <CX>
    <Axis><AxisType>A</Axistype><DataType>F</DataType> … </Axis>
  </CX>
  <CY>
    <Axis><AxisType>A</Axistype><DataType>F</DataType> … </Axis>
  </CY>
  <CZ>
    <Axis><AxisType>A</Axistype><DataType>F</DataType> … </Axis>
  </CZ>
</Record1>
```

In this case each matrix element contains three coordinates in 32-Bit floating point format in the order x,y,z.

Checksum

A binary file has its own MD5 checksum, which is stored in the `<MD5Checksum>` tag inside the `<DataLink>` tag of the xml-document. The checksum of the xml file is placed in a separate file that is linked in `<Record4>` by the `<ChecksumFile>` relative URI. The recommended filename should be `“md5checksum.hex”`. The checksum is a 128-bit hexadecimal encoded number. The checksum file thus contains 32 hexadecimal digits.

The calculation of the checksum uses an *MD5 Digest* as implemented in the unix command `“md5sum”`.

Best practice

- Orientation for 2.5D data in the matrix: x, y is sensor grid, z is depth.
- Files with more than 10000 data points should use binary encoding of the `<DataList>` due to performance reasons. Always using binary encoding is also OK, but decreases the readability of the resulting file and increases the file size for very small profiles.