ZISRAW (CZI) File Format

Design specification

Release Version 1.02

for

ZEN 2011

Table of Contents

1 General

1.1 Definitions and wording

Within this document we focus on the description of the CZI format for image storage. The data storage schema itself is more fundamental and is defined by ZISRAW, which means “Zeiss Integrated Software RAW” data format.

This format is intended to be a general purpose data format to be used in other parts of the software to store streamed data in a binrary file with some XML and binary metadata attached.

Currently, CZI is the only ZISRAW based implementation in the ZEN software.

1.2 Requirements

The definition of this file format was inspired by requirements for format simplicity (flat file), maximum data safety and performance but should also provide flexibility for future extensions.

1.3 Acknowledgements

Applies to Metadata definitions

This format has been developed to be as close as possible to the OME specification, Copyright 2002-2011 OME (Open Microscopy Environment). The XSD has been defined to show a maximum compatibility with the OME tiff and XML data formats while maintaining the essential requirements to run Carl Zeiss ZEN software optimally. Carl Zeiss MicroImaging GmbH acknowledges the copyright of OME and those parts of the czi format which are similar to the OME xml scheme are marked appropriately.

2 Implementation overview

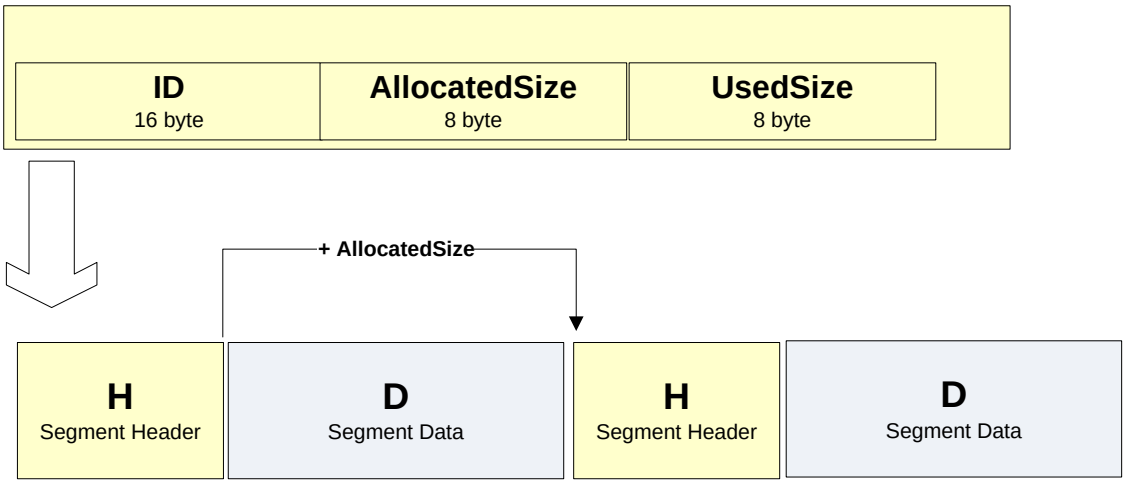
2.1 Segments and chaining

ZISRAW / CZI was designed to enable streaming of huge amount of data with a maximum of data safety.

The chosen architecture is a chain of “segments”. Each segment is identified by a header with defined Identifier (SID). Segments are aligned on 32 byte boundaries. This improves the speed of the recovery process when re-scanning the file in case of system crashes etc. A single search step in case of a missing segment header can move to the next multiple of 32 bytes instead of advancing byte by byte.

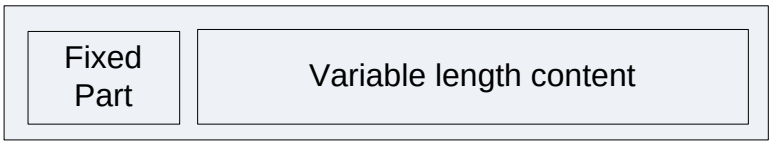
A SegmentHeader consists of

- A unique 16 byte ANSI – character **ID**, each prefixed with "ZISRAW"
- An allocated size (**AllocatedSize**) determining the space reserved for the segment data. Using the value, the reader can advance to the next segment. Allocated Size is always a multiple of 32 bytes.
- A usage size (**UsedSize**) determining the number of bytes already used in this segment. If zero, it is assumed that UsedSize is equal to AllocatedSize.



2.2 Segment Data

A segment's data consists of a defined fixed part and a variable length part.



The reader of a segment will first read the fixed part to determine the content and the data size of the variable part.

The data structure of the fixed part is given by the segment ID (SID) of the segment header.

2.3 Main content data types

2.3.1 Pixels (ImageSubBlock)

An **ImageSubBlock** contains the image pixels and closely related metadata. Raw, uncompressed pixel blobs may be multi-dimensional, e.g. stacks of 2D images.

The dimensionality is given by a definition structure array using an entry for each of the contained dimension. This entry defines the logical and physical (stored) size in means of pixels. The sequence of the entries defines the storage sequence within the pixel Blob.

2.3.2 Common Metadata

Each file has one Metadata segment containing

- an XML string matching a defined schema
- a binary section with additional data in ZIP format.

2.3.3 Attachments (Embeddings)

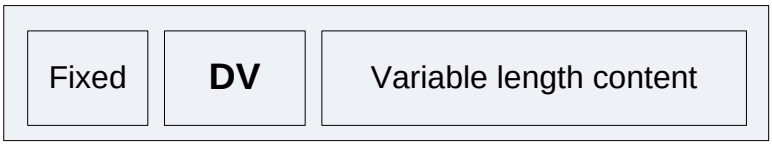
Any kind of content, e.g. TimeStamp Blobs can be embedded in the storage using the **Attachment** segments. Attachments are identified using a **name** instead of the dimension information used by the **SubBlock** segment.

Reserved names are e.g.: "**Preview**" and "**Thumbnail**" which are JPG files to be used in shell extensions and image browsers.

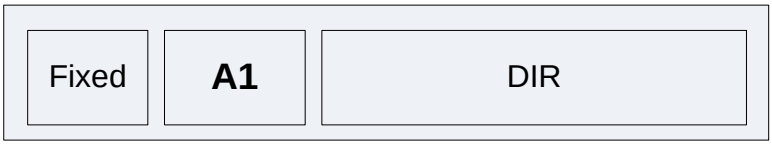
2.4 SubBlock / Attachment - Directories

Readers can access the whole file using the segment's ***AllocatedSize*** to advance from segment to segment. This may be time consuming with many segments, so the most commonly used segment types (***SubBlock*** and ***Attachment***) have **Directory** information parts that are cached in a summary segments.

A **SubBlock** Segment contains a directory entry (currently: DV-variable length schema) with index /and size information as each *SubBlock* is identfied by its bounds in a x-dimensional hyperspace and resolution (sub/supersampling)



An **Attachment** segment contains a directory entry (currently: Schema A1) with a name and a globally unique identifier (GUID).



Each DV/E or A1 entry contains the file part and the binary segment offset within the file.

The corresponding summary segments are **SubBlockDirectory** and **AttachmentDirectory**. They both contain a fixed part, followed by a list / array of DV or A1 entries. In most situations, this provides enough information about the related item, so full loading of the corresponding segment can be deferred until really needed.

3 Container

3.1 Byte order

Data is stored in "little-endian" (x64 compatible) format.

3.2 General

Data storage is based on "**Segments**" with defined schemas. Each of those segments contains data in Binary (Pixels, attachments) or XML format.

In any scenario, there is at least one master ZISRAW file containing the common directories and metadata.

The following storage scenarios are possible:

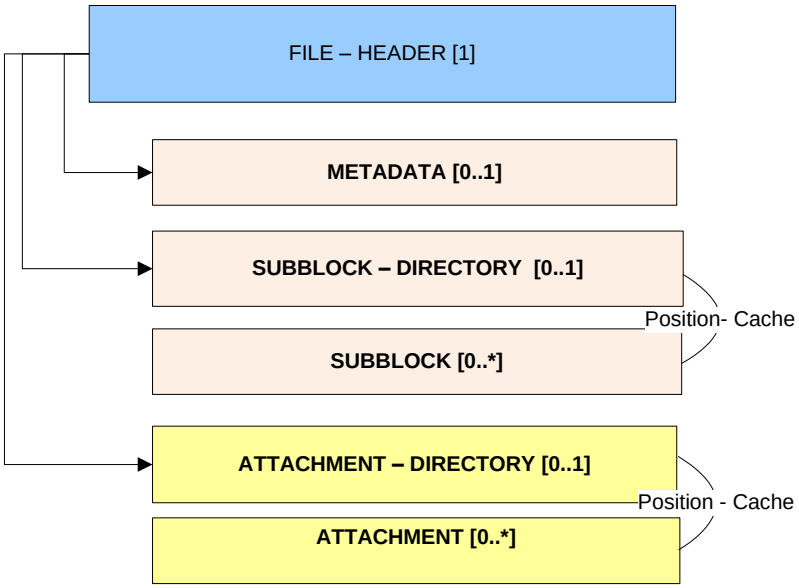
- **A single file.**
- **Multi – File** (master file and file parts)
- **[planned] CZI** container file and pixel data in external files. Pixels may be saved as JPG / TIFF etc.

3.3 File structure

Each ZISRAW file consists of one or more of the following segment types:

- **FileHeader** – one segment per file.
- **Metadata** – zero or one segment per file.
- **SubBlock** – one segment per SubBlock – optionally containing (XML) metadata and attached binary data like a thumbnail representation.
- **SubBlockDirectory** – directory of subblocks – zero or one segment per file. If a SubBlock segment exists, the ***SubBlockDirectory*** is required.
- **Attachment** – one segment per attachment (embedding) – identified via name. Contains any kind of data.
- **AttachmentDirectory** – directory of attachments - zero or one segment per file (optional). The directory contains the names and file offsets of the attachment segments. If an Attachment segment exists, the ***AttachmentDirectory*** is required.

New segment types may be introduced as required without breaking compatibility.



3.4 Single File Container

All data is contained in the same physical file. All file references are offsets into this file.

3.5 Multi – File container

The multi – file scenario is used in the following contexts:

- Avoid exceeding a certain size limit for a single file (e.g. a DVD chunk size)
- Split a multi-dimensional data storage based on the contained dimension to get independent files (e.g. one file per Z-Stack) to be used either stand-alone or in the context with others.

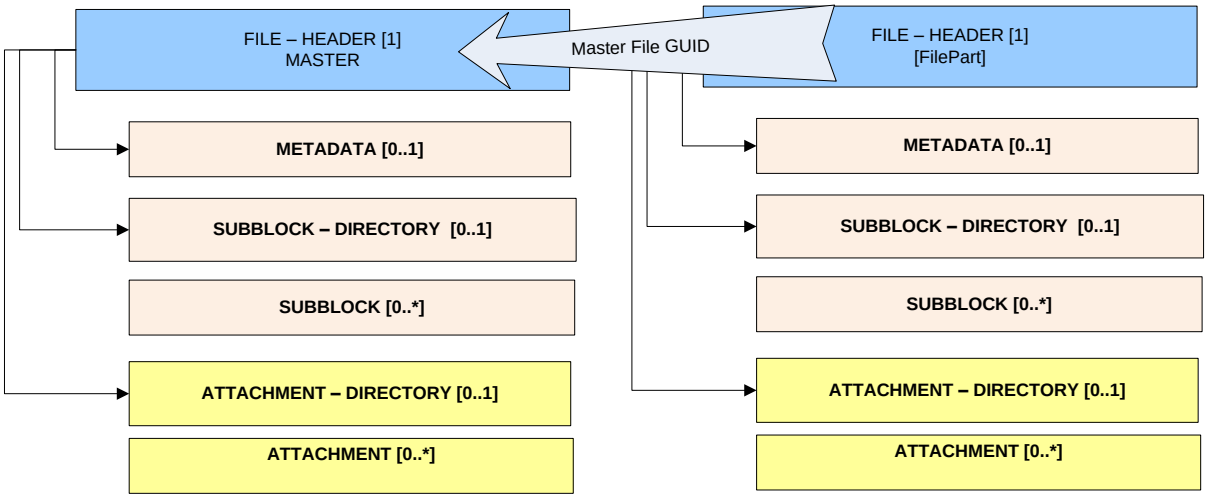
Each of the files of the multi-file storage is a complete valid image – maybe with its own thumbnail and preview.

If the user opens the **MasterFile** storage, the file parts are opened in addition and the contents of the various Directories are merged. The original file part number(s) is/are remembered.

When saving a multi-file based data storage, each file part directory is written separately using the **FilePart** information in the directory entry.

The following information is contained in the MasterFile only:

- The **Experiment** attachment segment
- The **Metadata** segment
- The **Preview** and **Thumbnail** attachments.



3.5.1 Multi-File naming schema

Using a strict naming schema we can avoid the need for embedding file names and enable renaming on file basis.

Master File:	Name.czi
File Part 1	Name (1).czi
File Part 2	Name (2).czi
File Part 3	Name (3).czi

The names "File (x). czi " never appear in the stored data, only the part numbers (1 .. 3).

The naming pattern was chosen to be compatible with the *Windows Explorer* multi-file renaming implementation. The user may multi-select all files in the multiple-file set – renaming the master file will also rename the file parts. The space between the name and the file part (in brackets) is optional, but it is added when renaming the files in *Windows Explorer*.

3.6 Recommendations for multi-file usage

When using the multi-file schema, it is recommended to follow those conventions:

- Use a separate folder for each file set (even if it is possible to store multiple file sets in a single folder).
- Name the folder according to its master file, e.g. folder “Name” contains Name.czi, Name (1).czi, Name (2).czi etc.

4 Data Storage - Segment schemas

4.1 Segment header

SchemaID = *SegmentHeader*

Item	Type	Offset	Size	Comments
Id	Byte[]	0	16	A sequence of up to 15 Ansi – characters 'A'…'Z' , e.g. "ZISSUBBLOCK" The special name "DELETED" marks a segment as deleted – readers should ignore or skip this segment.
AllocatedSize	Int64	16	8	The total numer of bytes allocated for this segment.
UsedSize	Int64	24	8	The currently used number of bytes.

Segment headers are aligned on 32 byte boundaries. A reader knowing nothing about the content of a segmented file can read the 32 byte header, take the first 15 characters as display text and then use **AllocatedSize** to advance the file pointer to the next segment.

4.1.1 Currently defined segment IDs (SID)

SID	Comments
ZISRAWFILE	File Header segment, occurs only once per file. The segment is always located at position 0.
ZISRAWDIRECTORY	Directory segment containing a sequence of "DirectoryEntry" items.
ZISRAWSUBBLOCK	Contains an **ImageSubBlock** containing an XML part, optional pixel data and binary attachments described by the ***AttachmentSchema*** within the XML part .
ZISRAWMETADATA	Contains **Metadata** consisting of an XML part an and binary attachments described by the **AttachmentSchema** within the XML part .
ZISRAWATTACH	Any kind of namend Attachment, some names are reserved for internal use.
ZISRAWATTDIR	Attachments directory.
DELETED	Indicates that the segment has been deleted (dropped) and should be skipped or ignored by readers.

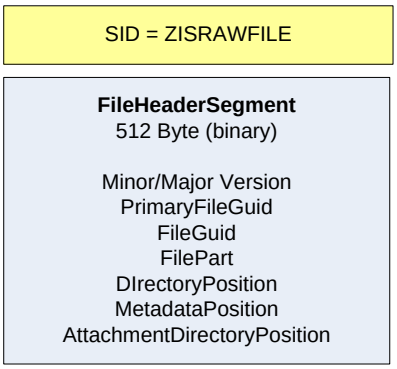
If a segment update would result in an overflow (**UsedSize** would exceed **AllocatedSize**), the segment is marked **DELETED** and a new segment is added to the end of the file.

Additional schemas may be defined if required.

4.2 File (Header) Segment

One segment per file.

4.2.1 Overview



4.2.2 Segment content schema

SchemaID = ***FileHeader***

Item	Type	Offset	Size	Comments
Major	Int	0	4	"1"
Minor	Int	4	4	"0"
Reserved1	Int	8	4	-
Reserved2	Int	12	4	-
PrimaryFileGuid	GUID	16	16	Unique Guid of Master file (FilePart 0)
FileGuid	GUID	32	16	Unique Per file

FilePart	Int32	48	4	Part number in multi-file scenarios
DirectoryPosition	Int64	52	8	File position of the SubBlockDirectory Segment
MetadataPosition	Int64	60	8	File position of the Metadata Segment.
UpdatePending	Bool	68	4	0xffff, 0 This flag indicates a currently inconsistent situation (e.g. updating Index, Directory or Metadata segment). Readers should either wait until this flag is reset (in case that a writer is still accessing the file), or try a recovery procedure by scanning all segments.
AttachmentDirectory Position	Int64	72	8	File position of the AtatchmentDirectory Segment.

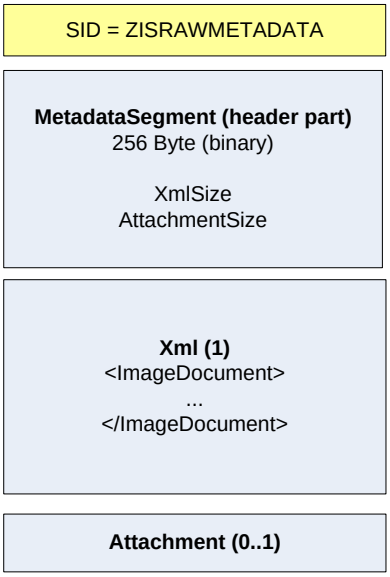
Single file: **PrimaryFileGuid** and the **FileGuid** are identical. The FilePart is 0.

Multi file: In the master file, the **PrimaryFileGuid** and the **FileGuid** are identical. In file Parts, the ***PrimaryFileGuid*** is the Guid of the master file and ***FileParts*** are > 0.

4.3 Metadata segment

One segment per file. This segment is used for global (once per image) metadata. The current Segment Position (seek offset) within the file is available from the file header (**MetadataPosition**)

4.3.1 Overview



4.3.2 Segment content schema

SchemaId = ***MetadataSegment***

Item	Type	Offset	Size	Comments
XmlSize	Int32	0	4	Size of the XML data.
AttachmentSize	Int32	4	4	Size of the the (binary) attachments. NOT USED CURRENTLY.
<spare>	<spare>	8	256-8	

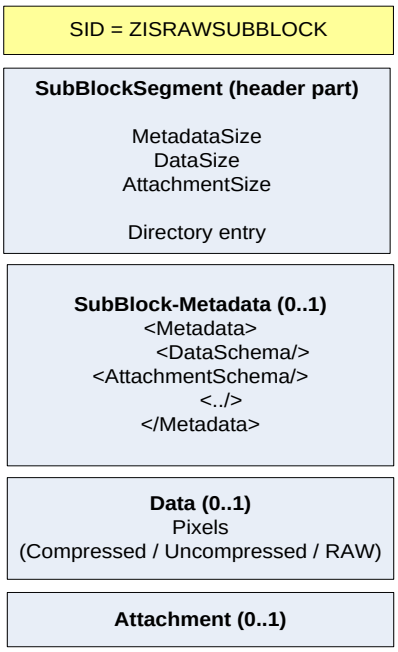
4.3.3 XML

XML data is stored as a string using UTF8 encoding.

4.4 SubBlock Segment

A SubBlock contains pixel data for a homogenous set / subset of the image. In light microscopy applications, most of the SubBlocks are two–dimensionsional where X and Y represent the camera frame. Confocal acquisition modes require other or more dimensions.

4.4.1 Overview



4.4.2 Segment content schema

SchemaId = ***SubBlockSegment***

Item	Type	Offset	Size	Comments
MetadataSize	Int32	0	4	Size of the metadata section.
AttachmentSize	Int32	4	4	Size of the optional attachment. section.
DataSize	Int64	8	8	Size of the data section..
DirectoryEntry	***DirectoryEntryDV***	16	variable	Subset indices and size information, an 1:1 copy will be stored as part of the File's **SubBlockDirectory** Segment. The length of this information depends on the directory schema.
Fill		n = variable + 16	Max(256 – n, 0)	Fill segment up to Minimum of 256 bytes
[*Metadata*]	String	off = Max(256, n)	<*MetadataSize*>	Metadata
[*Data*]	<any>	off + <*MetadataSize*>	<*DataSize*> >	Data (Pixels)
[*Attachments*]	<any>	off + <*MetadataSize*> + <*DataSize*> >	<*AttachmentSize*>	Attachments (binary, text, etc, may be complete files, e.g. a ZIP file)

Thus, the offset of the metadata section is 256 or the end of the directory entry if the size of the directory entry is greater than 240 (256 – 16)..

4.4.3 Metadata (XML)

XML data is stored as a string using UTF8 encoding.

4.4.4 Attachment

Optional attachments. Schema available in the **AttachmentSchema** node of the Metadata

4.4.5 Data

Main data (Pixels) either compressed or uncompressed as indicated by the "Compression" mode in the Directory part.

4.4.6 Directory Entry – Schema DV [Directory Variable length]

This schema uses a variable length entry to represent an variable number of dimensions. Each entry is represented by a **DimensionEntry** structure of one of ***DimensionEntryDV*** .

SchemaID = ***DimensionEntryDV1(20 bytes)***

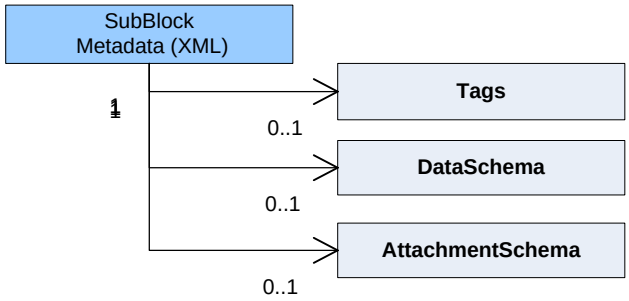
Item	Type	Offset	Size	Comments
Dimension	Byte[4]	0	4	Typically 1 Byte ANSI e.g. 'X', see Dimensions / dimensions indices
Start	Int32	4	4	Start position / index. May be < 0.
Size	Int32	8	4	Size in units of pixels (logical size). Must be > 0.
StartCoordinate	Float	12	4	Physical start coordinate (units e.g. micrometers or seconds)
StoredSize	Int32	16	4	Stored size (if sub / supersampling, else 0)

SchemaID = ***DirectoryEntryDV (32 bytes + EntryCount * 20)***

Item	Type	Offset	Size	Comments
SchemaType	Byte[2]	0	2	"DV"
PixelType	Int32	2	4	The type of the image pixels, see PixelTypes.
FilePosition	Int64	6	8	Seek offset of the referenced SubBlockSegment relative to the first byte of the file
FilePart	Int32	14	4	Reserved.
Compression	Int32	18	4	See Compression constants
PyramidType	Byte	22	1	[INTERNAL] Contains information for automatic image pyramids using SubBlocks of different resolution, current values are: None=0, SingleBlock=1, MultiBlock=2.
spare	Byte	23	1	Reserved
spare	Byte[4]	24	4	Reserved
DimensionCount	Int32	28	4	Number of entries. Minimum is 1.
DimensionEntries	***DimensionEntryDV***	32	DimensionCount * 20	Variable length array of dimensions. Each dimension can occur only once.

4.5 SubBlock specific Metadata

The small Metadata section for each **ImageSubBlock** defined to have the following content:




```
<METADATA>
   <DataSchema/>
   <Tags/>
   <AttachmentSchema/>
</METADATA>
```

4.5.1 Tags

Tags is a list of Name/Value pairs to provide any kind of additional information.

Sample:

```
<Tags>
    <ExposureTime>10.34</ExposureTime>
    <StageXPosition>103.4</StageXPosition>
    <StageYPosition>203.4</StageYPosition>
    <AcquisitionTime>2010-05-30T09:30:10.5</AcquisitionTime>
</Tags>
```

Physical positions

Tag	Type	Info
FocusPosition	double	Focus position in micrometers.
AcquisitionTime	DateTime	Acquisition Time in Xml “RoundTrip” format, e.g. `2010-05-30T09:30:10.5`
StageXPosition	double	Stage axis X position in micrometers.
StageYPosition	double	Stage axis Y position in micrometers.

Acquisition specific

Tag	Type	Info
ExposureTime	double	Exposure Time in milliseconds

4.5.2 DataSchema

The optional **DataSchema** node carries information about the structure of the data contained in the *Data* section. This can be the pixel format identifier, the pixel component details (e.g. BGR Triples, XYZ tokens) etc. and also data to decode the pixels if they are in RAW format.

```
<DataSchema>
       <DataFormat>Pixels</DataFormat>
       <ValidBitsPerPixel>12</ValidBitsPerPixel>
</DataSchema>
```

Most elements within this schema are optional. for pixels encoded using a "Raw" – encoded format (AxioCam etc), RawDeoderParameters and RawDecoderContextId are mandatory.

Image files using Raw formats can be decoded into standard formats using specific conversion tools. The "Raw" encoding schemas are not part of this documentation.

Name	Type	Info
DataFormat	String	Information about the data format. Valid values are: • **Pixels** (default) = n-dimensional pixel array– each pixel is defined by the **PixelType** identifier in the directory header.
Name	String	A user defined name.
MinValue	Double	Minimum value of a pixel. For Composite / RGB types, the minimum component value.
MaxValue	Double	Maximum value of a pixel. For Composite / RGB types, the maximum component value.
LowValue	Double	Recommended normalized Low value for display mapping with respect to the range of the date value. A value of 0 means: start mapping with data value 0.
HighValue	Double	Recommended normalized High value for display mapping with respect to the range of the date value. A value of 1.0 means: the maximum value for the given data type.
ValidBitsPerPixel	Int	The number of bits with meaningful value. Will be set to 12 for 12 Bit and to 14 for 14 Bit sensors. For multi-component types, the value is the sum of all components, e.g. 12 Bit RGB camera has a ValidBitsPerPixel of 36.
RawDecoderParameters	XML Node	Decoder parameters as Xml fragment. The decoder will de-serialize this information if required.
RawDecoderContextId	String	The Id (name) of an attachment containing additional binary data for the image decoder.

4.5.3 AttachmentSchema

This optional **AttachmentSchema** node contains storage details of the data in the "Attachments" section.

```
<AttachmentSchema>
       <DataFormat>MeasurementRegion</DataFormat>
</AttachmentSchema>
```

DataFormat for attachment schema

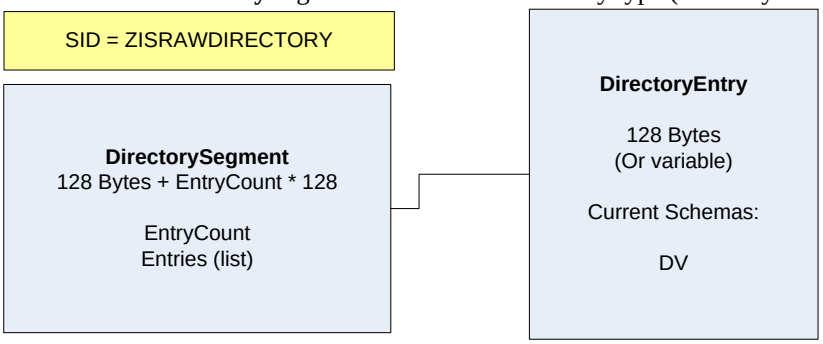
Name	Type	Info
DataFormat	String	Information about the attachment data format.

4.6 SubBlock - Directory

One segment per file.

4.6.1 Overview

The **SubBlockDirectory** segment contains entries of any type (currently the only supported schema is DV).



4.6.2 Segment content schema

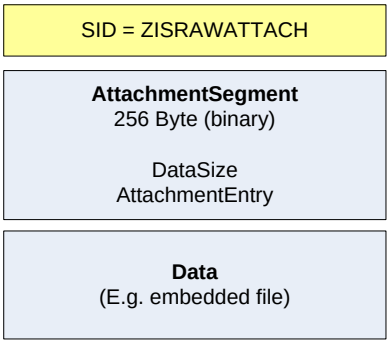
SchemaID = ***SubBlockDirectorySegment***

Item	Type	Offset	Size	Comments
EntryCount	Int	0	4	The number of entries
Reserved	byte	4	124	
Entry[]	***DirectoryEntryDV***	128	variable	List of **EntryCount** items. Each item is a copy of the ***DirectoryEntry*** in the referenced SubBlock segment.

4.7 Attachment Segment

Multiple segments per file.

4.7.1 Overview



4.7.2 Segment content schema [256 byte]

SchemaId = ***AttachmentSegment***

Item	Type	Offset	Size	Comments
DataSize	Int32	0	4	Size of the data section.
<spare>	Byte	4	12	reserved
AttachmentEntry	***AttachmentEntryA1***	16	128	Core information, an 1:1 copy will be stored as part of the File's **AttachmentDirectory** Segment.
<spare>	Byte	144	112	reserved
[*Data*]	<any>	256	<*DataSize*>	Embedded file.

4.7.3 Data

Binary or textual data as specified in the **AttachmentEntry** information.

4.7.4 AttachmentEntry - Schema A1 [128 bytes]

SchemaID = ***AttachmentEntryA1*** *(fixed length = 128 bytes)*

Item	Type	Offset	Size	Comments
SchemaType	Byte[2]	0	2	"A1"
Reserved	Byte[10]	2	10	Reserved
FilePosition	Int64	12	8	Seek offset relative to the first byte of the file
FilePart	Int	20	4	Reserved
ContentGuid	GUID	24	16	Unique Id to be used in strong, fully qualified references
ContentFileType	Byte[8]	40	8	Unique file type Identifier (see table below)
Name	Byte[80]	48	80	Null terminated (80-1) character UTF8 encoded string defining a name for this item. May be used in references instead of GUID.

ContentFileType identfier strings (samples)

File types correspond to the file extension when the attachment is saved as a separate file.

Type	Comments
ZIP	ZIP compressed stream
ZISRAW	Embedded ZISRAW file..
CZTIMS	Time stamp list.
CZEVL	Event List..
CZLUT	Lookuptable.
CZPML	Pal molecule List.
JPG, DOC, XLS, ..	Any registered MIME file type

4.7.5 Reserved Attachment names

The following values for Name in AttachmentEntryA1 schema are reserved and have determined contents.

Item	DataFormat	Content FileType	Comments
Thumbnail	JPG file	JPG	Contains a thumbnail representation of the image. Typically, a thumbnail is downscaled to match a raster of 256 x 256 pixels, but other sizes are also possible. Thumbnails do not necessarily match the individual pixel SubBlocks, instead. They can represent be a symbolic representation to quickly identify the content.
Preview	Media file	JPG (or other media type)	The optional preview image / media file is used to implement a more detailed view of the image using some representative data (e.g. one of the center slices of a Z-Stack). Typical sizes will fit in a 1024 x 1024 raster.
Experiment	XML	CZEXP	If the image is the result of a (multidimensional) experiment, this attachment contains the original experiment definition. The Schema is described in the Metadata section as Experiment.
HardwareSetting	XML	CZHWS	Contains the initial hardware setting / configuration used to record this image.
TimeStamps	Binary	CZTIMS	Time stamps in seconds relative to the start time of the acquisition engine..
EventList	Binary	CZEVL	EventList stores the events reported during a timeseries.
Lookuptables	Binary	CZLUT	Contains the properties of the lookup tables.
PalMoleculeList	Binary	CZPML	The PalMoleculeList provides access to the list of molecules and fiducials which have been detected by the PAL-M method for resolution enhancement. A PalMolecule contains the results for the PAL-M single molecule detection method for resolution enhancements of a single molecule. The list entries are generated form a series of TIRF images by sub-pixel localization of molecules with the highest intensity in the single images by fitting to the PSF of the microscope.
FocusPositions	Binary	CZFOC	Focus positions relative to the start position of the acquisition engine..

Please note that the combination of 'Item' and 'Content FileType' is generally not arbitrary. Therefore e.g. the item 'Thumbnail' must be of content filetype 'JPG'. The only exception is the item Preview, where JPG is expected but other media types are possible. The entry names are case sensitive.

4.7.6 TimeStamps content schema

SchemaId = ***TimeStampSegment***

Item	Type	Offset	Size	Comments
Size	Int32	0	4	Size of the whole block used for time stamps.
NumberTimeStamps	Int32	4	4	Number of time stamps in the list.
TimeStamps	double[]	8	NumberTimeStamps * 8	Time stamps in **seconds** relative to

				the start time of the acquisition engine.

4.7.7 FocusPositions content schema

SchemaId = ***FocusPositions***

Item	Type	Offset	Size	Comments
Size	Int32	0	4	Size of the whole block used for focus positions.
NumberPositions	Int32	4	4	Number of positions in the list.
Positions	double[]	8	NumberPositons * 8	Focus positions in **micrometers** relative to the Z start position of the acquisition engine.

4.7.8 EventList content schema

SchemaId = ***EventListSegment***

Item	Type	Offset	Size	Comments
Size	Int32	0	4	Size of the whole block in bytes.
NumberEvents	Int32	4	4	Number of recorded events in the list.
Events	EventListEntry[]	8	variable	

4.7.9 EventListEntry content schema

SchemaId = ***EventListEntry*** *(fixed length = 16 bytes + DescriptionSize)*

Item	Type	Offset	Size	Comments
Size	Int32	0	4	Size of the entry in bytes.
Time	double	4	8	Time of the event in seconds relative to the start time of the LSM electronic module controller program.
EventType	Int32	12	4	Can be one of the following values: EV_TYPE_MARKER (= 0) - Experimental annotation EV_TYPE_TIMER_CHANGE (= 1) - The time interval has changed EV_TYPE_BLEACH_START (= 2) - Start of a bleach operation EV_TYPE_BLEACH_STOP (= 3) - End of a bleach operation EV_TYPE_TRIGGER (= 4) - A trigger signal was detected on the

				user port of the electronic module.
DescriptionSize	Int32	16	4	Size of the description character array.
Description	Byte[]	20	variable	Null terminated (80-1) character UTF8 encoded string defining a description for this event.

4.7.10 LookupTables content schema

SchemaId = ***LookupTablesSegment***

Item	Type	Offset	Size	Comments
Size	Int32	0	4	Size of the whole description block including this header in bytes.
NumberLookupTables	Int32	4	4	Number of lookup tables handled in the description block.
LookupTables	LookupTableEntry[]	8	variable	12 bit to 12 bit LUT for the corresponding channels 1..N.

4.7.11 LookupTableEntry content schema

SchemaId = ***LookupTableEntry***

Item	Type	Offset	Size	Comments
Size	Int32	0	4	Size of the whole description block including this header in bytes.
Identifier	Byte[80]	4	80	Null terminated (80-1) character UTF8 encoded string defining a name for this lookup table.
NumberComponents	Int32	84	4	Number of components handled in the description block.
Components	ComponentEntry[]	88	variable	Component part of the LookupTable.

4.7.12 ComponentEntry content schema

SchemaId = ***ComponentEntry***

Item	Type	Offset	Size	Comments
Size	Int32	0	4	Size of the whole description block including this header in bytes.
ComponentType	Int32	4	4	Component type: CO_TYPE_RGB (= -1) - All (Rgb) CO_TYPE_RED (= 0) - Red CO_TYPE_GREEN (= 1) - Green CO_TYPE_BLUE (= 2)

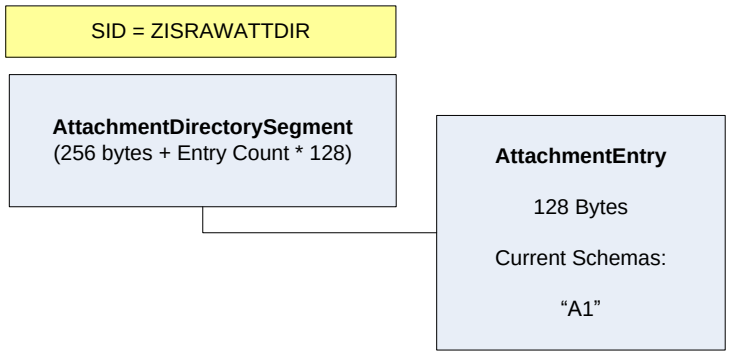
				- Blue
NumberIntensities	Int32	8	4	Number of intensities.
Intensity	Int16[]	12	NumberIntensities*2	Intensities for the component of the lookup table.

4.8 AttachmentDirectory Segment

Zero or one segment per file.

The **AttachmentDirectory** segment is used to manage attached / embedded files which may, in turn, be valid ZISRAW files. Typical usage this segment is to hold commonly used parts like a **Thumbnail** or a **Preview**.

4.8.1 Overview



4.8.2 Segment content schema

SchemaID = ***AttachmentDirectorySegment***

Item	Type	Offset	Size	Comments
EntryCount	Int	0	4	The number of entries
Reserved	Byte	4	252	
Entry[]	***AttachmentEntryA1***	256	EntryCount * 128	List of **EntryCount** items..

5 Pixel storage

5.1 PixelTypes

Id	Value	Bytes/Pixel	Info
Gray8	0	1	8 bit unsigned
Gray16	1	2	16 bit unsigned.
Gray32Float	2	4	32 bit IEEE float
Bgr24	3	3	8 bit triples, representing the color channels Blue, Green and Red
Bgr48	4	6	16 bit triples, representing the color channels Blue, Green and Red
Bgr96Float	8	12	Triple of 4 byte IEEE float, representing the color channels Blue, Green and Red

Bgra32	9	4	8 bit triples followed by an alpha (transparency) channel
Gray64ComplexFloat	10	8	2 x 4 byte IEEE float, representing real and imaginary part of a complex number
Bgr192ComplexFloat	11	24	A triple of 2 x 4 byte IEEE float, representing real and imaginary part of a complex number, for the color channels Blue, Green and Red
Gray32	12	4	32 Bit integer [planned]
Gray64	13	8	Double precision floating point [planned]

5.2 Dimensions / dimensions indices

The ***Bounds*** definition of a **SubBlock** defines the index within a multi-dimensional hyperspace. All indices are integers but,, Using the **StartCoordinate** member of the dircectory, it is possible to associate a floating point value, e.g. to implement sub-pixel accurancy for image tiles (X/Y offset).

Dimensions in both metadata and binary information blocks are identified by a unique ANSI/ASCII character 'A', 'Z'. The following dimensions are currently defined, more will be added -

Id	Info
X	Pixel index / offset in the X direction. Used for tiled images.
Y	Pixel index / offset in the Y direction. Used for tiled images.
C	Channel in a Multi-Channel data set
Z	Slice index (Z – direction).
T	Time point in a sequentially acquired series of data.
R	Rotation – used in acquisition modes where the data is recorded from various angles.
S	Scene – for clustering items in X/Y direction (data belonging to contiguous regions of interests in a mosaic image).
I	Illumination - illumination direction index (e.g. from left=0, from right=1).
B	(Acquisition) Block index in segmented experiments.
M	Mosaic tile index – to reconstruct the image acquisition order and to be used in conjunction with a global position list and the scaling information to define the pixel offset (alternative to using StartX, StartY to allow multi-M SubBlocks in case of homogenous mosaics, i.e. all tile share the same position list)
H	Phase index – for specific acquisition methods.

5.3 Compression

Pixel data may be either compressed or uncompressed. Compression is determined by a ***Compression*** constant other than ***Uncompressed***.

Compresion	Value	Info
Uncompressed	0	Data contains uncompressed pixels.. Data is a stream of items with the specified PixelType and Size information. Each item is adressed by evaluating the "StoredSize" information in dimension order.

		The total data size can be calculated from BytePerPixel(PixeType) * multiplication of all stored sizes in all contained dimensions.
JPEG	1	Data contains a single RGB or monochrome JPEG file. This compression mode can only be used with 2D SubBlocks, i.e. SizeC = SizeZ = SizeT = 1 The summary information about the stored size represents the bitmap size in pixels. Anyway, the reader should check the size obtained from the JPEG decoder.
LZW	2	Data contains pixels compressed with the Lemple-Ziff-Welch algorithm.
Camera	100-999	Camera specific RAW data..
System	1000-	System specific RAW data..

6 Metadata (XML)

6.1 General

Metadata is always in XML format and is stored as UTF-8.

This chapter is intended to provide a course overview. Detailed schema documentation is available as a separate HTML or CHM based documenntation.

6.2 Acknowledgements

This part of the specification is close to the OME specification, Copyright 2002-2011 OME (Open Microscopy Environment), see our acknowledgment in the header."

6.3 Versioning

Versioning in ZISRAW has two aspects which will be named Reorganization-Versioning and Sub-Versioning. Sub-Versioning means a ‘friendly extension’ of a given version while Reorganization-Versioning aims at a redesign of one or many given structures.

6.3.1 Sub-Versioning

To understand the idea of Sub-Versioning we have to make certain assumptions in advance. They refer to the way we treat XML and XSD. Sub-Versioning is characterized by the digits after the dot e.g. 1.00, 1.01, 1.02. This is stressed by the filename e.g. czi_v1.xsd so the Sub-Versioning is ‘internal’.

The following assumptions are fundamental

- XML data is always saved back in full range. This is especially true for XML data that cannot be interpreted completely so that parts that could not be understood are persisted.
- XSD is only enlarged as far as data and structure is concerned. This implies that changes to the XSD just mean new features have been added, like new elements, new attributes or even new structures. Therefore reorganization of already given structures, renaming of structures, elements or attributes or changing of semantics of elemens is not allowed in this context.
- Must-Entries are not used
- For the validation of the Sub-Version we result in different scenarios.

  Scenario 1 – XML and XSD contain same data

  - Validation is successful.
    The program can continue without errors
  - Validation failed.
    There exists a more or less severe error. The user has to be informed. The object model catches the error and gives detailed information. If possible, the program proceeds with the given limitations. In severe situations the opening of the image is stopped and the user has the possibility to reorganize the file to use it later on with a minimum of functionality.

  Scenario 2 – XML is an older version than XSD

  - Validation is successful.
    As the XSD has more capabilities than the XML needs, the XML can always be validated. This is even true when the XML and the Version of ZEN are older than the XSD itself. In this case it depends on the program version and its object model how much of the data can be used.
  - Validation failed.
    See Scenario 1.

  Scenario 3 – XSD is an older version than XML

  - The Versions are recognized and the problem is known to the program

- The user is asked to download the actual XSD directly via internet. If this is not possible, e.g. a connection to the internet is not found, the url and the target directory are shown to the user and he is asked to manually supply the most recent XSD. After this procedure Scenario 1 or Scenario 2 is applicable.
  In any case the user is able to go on working because we make the assumption that the data is valid and not corrupted. The software is simply supplied with more data than it can work with.
- In case the validation is turned on, the software must guarantee that the ZEN version can at least access its corresponding XSD. This is achieved by compiling the XSD directly into the versions of ZEN. The software looks the latest version of the XSD, especially within the Sub-Version of a required Version.

6.3.2 Reorganization-Versioning

Reorganization-Versioning means rearrangement of already given structures, renaming of structures, elements or attributes, changes in definitions or any other severe, far reaching changes.

Reorganization-Versioning is characterized by the digits in front of the dot e.g. 2.00, 3.01, 4.02. Therefore each version has its own file and the xml tells the software which XSD to load.

As the XSD consists of several XSD-files like 'Experiment.xsd', 'HardwareSettings.xsd', 'Information.xsd', 'DisplaySettings.xsd' etc. it is possible to reorganize any specific part of the schema. This means that for instance a reorganized 'Experiment_2.00.xsd' may coexist with previous versions like 'HardwareSettings_1.09.xsd', 'Information_1.09.xsd'

Although an earlier version of the software will be able to validate the xml, see Scenario 3, it will automatically detect areas that cannot be handled. The degree of incompatibility depends on the degree of rearrangement or changes in the XSD compared to the object model holding the data. The software with the new XSD is automatically able to verify an old czi file, because the xml knows the version and the older versions are also supplied in an xsd-folder of the ZEN-software or Scenario 3 is applicable. For that reason internally the old way to access the data must be maintained to show and handle the information as expected.

Optional: When saving an image the user is asked, whether old or the new format is to be applied.

6.4 General XML information

In ZISRAW files, XML metadata is entirely optional as all information about dimensions, pixel types etc. is contained in the binary SubBlock and / or SubBlockDirectory segments .

Anyway, to provide useful information and enable advanced processing, some elements should be provided. In the following chapters, the column "req" (required) indicates one of

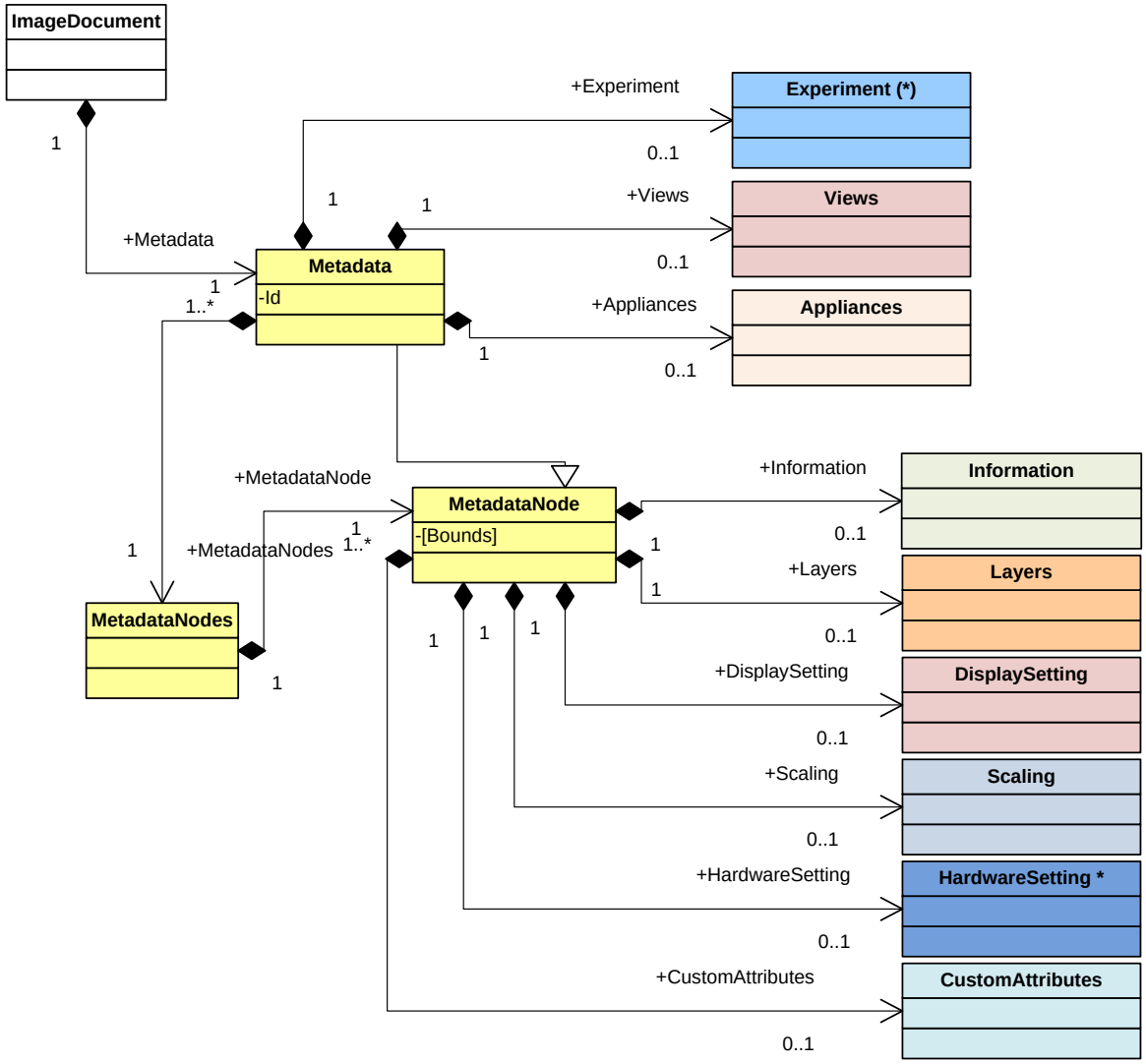
- Y – mandatory, marked as required in the schema defintion
- B – basic information, optional in schema definition but required to show GUI elements like rulers and display setting
- A – requirement is application dependent, e.g. required for 3D deconvolution

Unless otherwise noted, the default value for all values is as follows

- **xs:double** : NaN
- **xs:string**: "" (an empty string) or the string constant shown in bold face if string uses an enumerated restriction.
- **xs:boolean** : false

6.5 Storage structure

XML Storage is based on the ***MetadataNode*** schema containing the predefined elements **Information**, **Layers**, **DisplaySetting**, **Scaling** and **CustomAttributes**.



NOTE: Items marked with (*) may be stored in separate attachment segments, they are:

- **Experiment** – the full experiment description at the time when the acquisition process was started. Normally this XML node is created once. The Experiment schema is currently undocumented.
- **HardwareSetting** – the hardware setting / configuration at the time the experiment was started. This XML node represents the full hardware state including all the metadata used to reconstruct the GUI elements if required. State changes during the flow of the experiment are expressed via the nodes ***HardwareSetting*** node of the subset specific Metadata nodes. HardwareSettings are an advanced feature and are currently undocumented.

Element	req	Type (XSD)	Sample	Description
`Version`		xs:string	1.0	Version information for this XML content, expressed as <Major>.<Minor>. Default Value is 1.0.
`Experiment`		<unspecific>	n/a in this context	Serialized Experiment. Separate schema is available as element **Experiment** in Experiment.xsd.
`Views`		Views	<complex content/>	Reserved. Stores multiple predefined views for the data set.
`Applicances`		Appliances	<complex content/>	Reserved. Stores data specific for named

				appliances like measurement or advanced data analysis.
Information	B	Information	<complex content/>, see Information.	Typical document properties like comments, keywords, author, etc.
Layers		Layers	<complex content/>, see Layers.	A collection of graphical (overlay) layers, either global or subset specific.
DisplaySetting	B	DisplaySetting	<complex content/>,see DisplaySetting.	Default or subset specfic multi-channel display setting (channels selection, colors, display mapping curves, gamam etc.)
Scaling	B	Scaling	<complex content/>,see Scaling	Scaling values for the various dimensions. Normally, only X,Y and Z have useful entries.
HardwareSetting		<unspecific>	<complex content/>	Detailed information about the recording hardware in original manufacturer format. Separate schema documentation is available as HardwareSetting.xsd.
CustomAttributes		CustomAttributes	<complex content/>, see Custom attributes	Location for unlimited, but unchecked storage of named custom values (simple or complex types).

6.6 Custom attributes

CustomAttributes is an opaque list of elements that should be persisted when loading and saving files.
The reader of this element must store the XML of all child elements, e.g. in a dictionary, to be restored when saving.

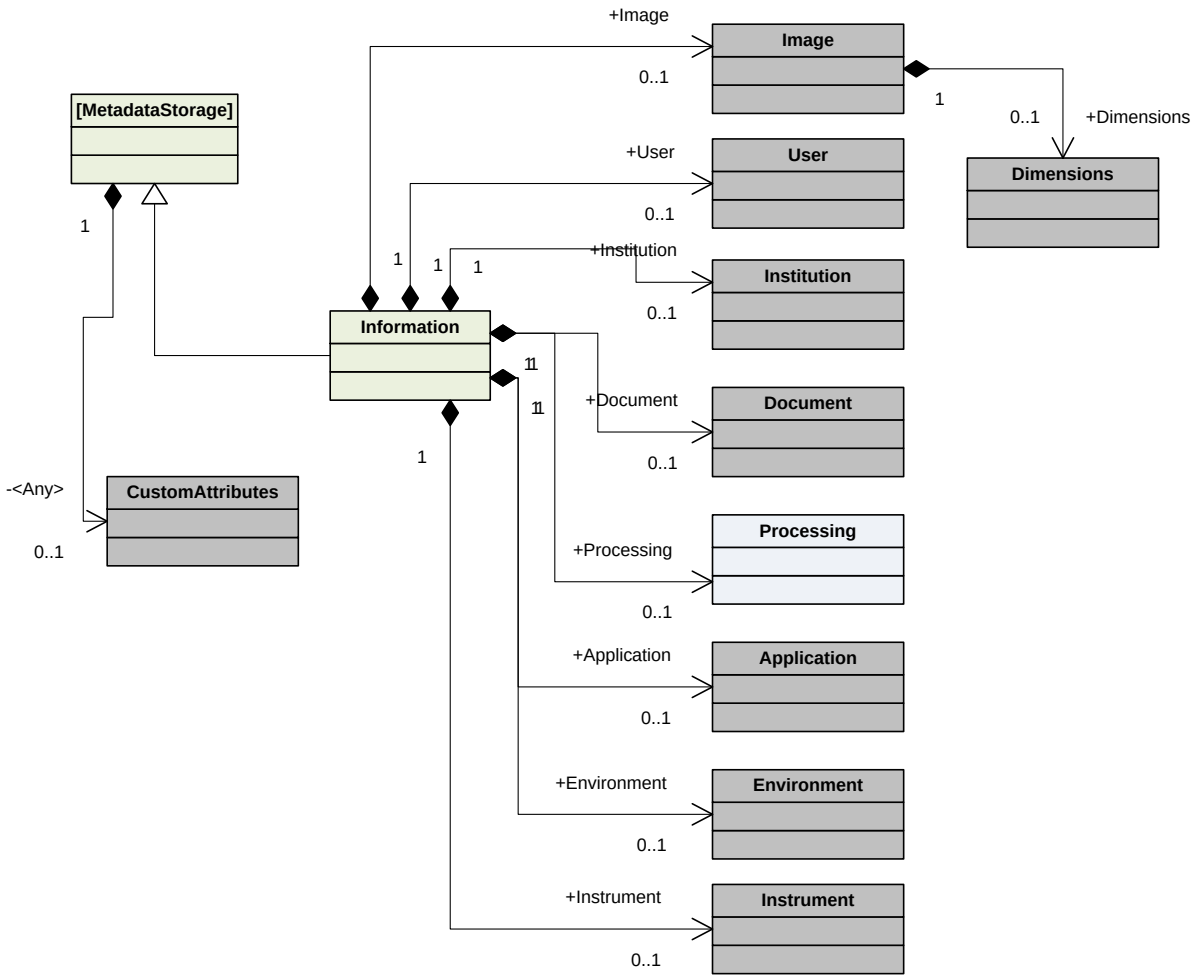
6.6.1.1 Sample: CustomAttributes

```
<CustomAttributes>
  <Test>The content<Test>
  <Test2/>
  <Test2/>
</CustomAttributes>
```

6.7 Information

The **Information** element carries all the data used to implement the informational views with the application and the various GUI elements to work with a multi-dimensional image.

All elements within the information storage are based on the ***MetadataStorage*** schema and provide a **CustomAtttributes** node for user-defined extensions without breaking the schema's compatibility.

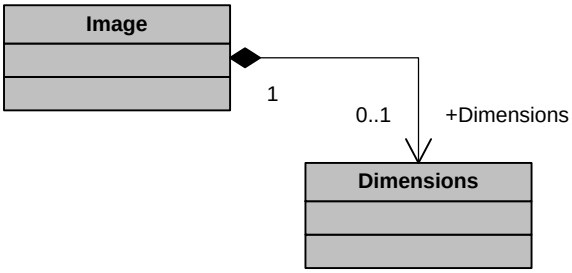


Element	req	Type (XSD)	Sample	Description
`Image`	B	Image	<complex content/>, see Information/Image	Describes the image's main attributes like multi-dimensional bounds and dimension specific data.
`User`		User	<complex content/>, see Information/User	User related information added provided by he application (Name etc.)
`Institution`		Institution	<complex content/>,see Information.Institution.	Information about the institution where the image was recorded (name, address, email etc).
`Document`		Document	<complex content/>, see Information/Document.	Typical document properties like comments, keywords, author, etc.
`Processing`	A	Processing	complex content, see Information/Processing.	Specific data for use within processing algorithms, e.g. the PSF information for deconvolution.
`Application`		Application	<complex content/>,see Information.Application.	Information about the creating application (program)..
`Environment`		Environment	<complex content/>	Information for experiment / recording environment (temperature, humidity, etc).
`Instrument`	BA	Instrument	<complex content/>,,see	Detailed information about the

			Information.Instrument	recording hardware.

6.7.1 Information / Image

Contains basic image information like dimensions and their sizes and optionally details about the contained Dimensions.



The **Sizes** and pixel data related metadata information is provided in addition only to provide a convenient method to display summary information by reading the XML segment only.

Sizes define the overall dimensions (bounds) of the multi-dimensional data set. All values must be > 0, the default value for each dimension is 1.

When opening an image, its "real" bounds should be calculated from the contained **SubBlock** segments (as specified in the directory information).

Special for "req" column: If enclosed in brackets, e.g. (B) means that this element should be available if the image contains this dimension.

Element	req	Type (XSD)	Sample	Description
SizeX	B	xs:integer	200	Number of pixels in x sdirection (width).
SizeY	B	xs:integer	200	Number of pixels in y direction (height).
SizeC	(B)	xs:integer	4	Number of channels.
SizeZ	(B)	xs:integer	20	Number of Z slices.
SizeT	(B)	xs:integer	100	Number of timepoints.
SizeH	(B)	xs:integer	1	Number of phases.
SizeR	(B)	xs:integer	1	Number of rotation angles (indices).
SizeS	(B)	xs:integer	1	Number of scenes.
SizeI	(B)	xs:integer	1	Number of illumination direction indices.
SizeM	(B)	xs:integer	1	Number of mosaic tiles (regular mosaics only).
SizeB	(B)	xs:integer	1	Number of acquisition / recording / blocks Each block may have specific dimensions expressed via Information.Image in a subset specfic Metadata node.
PixelType	B	PixelType (xs:string restriction)	Gray8	preferred pixel type (individual types are defined iin binaryimage data and / or channel XML information)

				one of … *Gray8, Gray16, Bgr24, Bgra32, Gray32Float, Bgr48, Bgr96Float, Gray64ComplexFloat, Gray32Float, Bgr192ComplexFloat.*
ComponentBitCount	B	xs:integer	12	ComponentBitCount for the entire image is for informative purpose only. In ***Dimensions/Channels/Channel*** each Channel has its own *ComponentBitCount* element. If not specified or 0, the component bit count is derived from the pixel type. Normally, this value is used with 16 bit images only to indicate the maximum possible valid bits of the sensor, e.g. 12 defines a 12 bit camera.
OriginalScanData		xs:boolean	true	Set to true if the image is the output of a scanning process and has not been modified up to now.
Dimensions	B	Dimensions	<complex content/>, see Information.Image.Dimensions	More detailed information about each contained dimensions.

6.7.1.1 Sample: Image

```
<Image>
 <SizeX>200</SizeX>
 <SizeY>200</SizeY>
 <SizeZ>20</SizeZ>
 <SizeT>100</SizeT>
 <OriginalScanData>true</OriginalScanData>
 <AcquisitionDateAndTime>2001-01-28T00:00:00</AcquisitionDateAndTime>
 <PixelType>Gray8</PixelType>
 <ComponentBitCount>8</ComponentBitCount>
 <ComponentHighValue>255</ComponentHighValue>
 <OriginalScanData>true</OriginalScanData>
 <Dimensions>
       <Channels>
            <Channel Id="Channel:1">
..

</Image>
```

6.7.2 Information / Image / Dimensions

Contains information about contained dimensions.

Element	req	Type (XSD)	Sample	Description
Channels		DimensionChannel	complex content, see Information / Image / Dimensions / Channels / Channel	A collection of Channel information
Tracks		<anonymous complex type>	complex content. See Information / Image / Dimensions / Tracks	Contains information about tracks.

LambdaStacks		<anonymous>		
T		<anonymous>	complex content, see	
Z				

6.7.3 Information / Image / Dimensions / Channels / Channel

XSD complex type **DimensionChannel**.

There must be at least one element per channel in the Image, even for a single-plane image.

And information about how each of them was acquired is stored in the various optional *Ref elements.

The IlluminationType element is a string enumeration which may be set to 'Transmitted', 'Epifluorescence', 'Oblique', or 'NonLinear'.

Contains user and site information.

Attribute	req	Type (XSD)	Sample	Description
Id	Y	xs:string	Channel:0	Unique ID within a list of siblings. Recommended format is Channel:<nr>.
Name		xs:string	DAPI	Default, user supplied, name of the channel.

Element	req	Type (XSD)	Sample	Description
PixelType	B	restriction of xs:string	Gray8	Information about the pixel type of this channel (should match the binary data in the SubBlock segment) One of.. *Gray8, Gray16, Bgr24, Bgra32, Gray32Float, Bgr48, Bgr96Float, Gray64ComplexFloat, Gray32Float, Bgr192ComplexFloat.*
ComponentBitCount	B	xs:string	12	ComponentBitCount for the channel If not specified or 0, the component bit count is derived from the pixel type. Normally, this value is used with 16 bit images only to indicate the maximum possible valid bits of the sensor, e.g. 12 defines a 12 bit camera.
AcquisitionMode		restriction of xs:string	Brightfield	AcquisitionMode describes the type of microscopy performed for each channel. One of.. *Brightfield, LaserScanningConfocalMicroscopy, SpinningDiskConfocal, SlitScanConfocal, MultiPhotonMicroscopy, StructuredIllumination, SingleMoleculeImaging, TotalInternalReflection, FluorescenceLifetime, SpectralImaging, FluorescenceCorrelationSpectroscopy, NearFieldScanningOpticalMicroscopy, SecondHarmonicGenerationImaging, PALM, STORM, STED, TIRF, FSM, LCM,* ***Other***
IlluminationType	A	restriction of	Transmitte	The method of illumination used to capture

		xs:string	d	the channel. One of *Transmitted, Epifluorescence(*1), Obliquem NonLinear, Other*
ContrastMethod	A	restriction of xs:string	Brightfield	***ContrastMethod*** describes the technique used to achieve contrast for each channel. One of *Brightfield, Phase, DIC, HoffmannModulation, ObliqueIllumination, PolarizedLight, Darkfield, Fluorescence,* ***Other***
IlluminationWavelength	A	SpectrumCharacteristic	340	Illumination Wavelength as either a single Peak or a list of Ranges. This characterizes the light used for illuminating the specimen. More specific, it is about that the light that actually hits the specimen, not the light as it leaves the light source.
DetectionWavelength	A	SpectrumCharacteristic	320	Detection Wavelength as either a single Peak or a list of Ranges.This characterizes the part of the spectrum for which the detector used for this channel is actually sensitive for. It gives in any case the "net result"-it does not matter what technique is used to limit the detection range.
ExcitationWavelength	A	xs:double	400	Wavelength of excitation for this channel in nanometers. This characterizes the fluorochome - the fluochrome one was interested in this channel. It has just an informative meaning - it is not meant to characterize the fluochrome in depth, that is what the DyeId is meant for. [units:nanometers].
EmissionWavelength	A	xs:double	400	Wavelength of emission for this particular channel, in nanometres[nm].
DyeId	A	xs:double	400	The dye id which is unique within the original dye database.
DyeDatabaseId		xs:string		The id (Guid) of the original database this dye is taken from.
PinholeSize	A	xs:double	1.2	The optional PinholeSize element allows specifying adjustable pin hole diameters for confocal microscopes. The Pinhole is track specific. I.e. channels of the same track need to have the same value here. [units:micrometers].
PinholeSizeAiry	A	xs:double		The size of the pinhole in units of the airy disc. Since this value cannot (easily) be derived from the above ***PinholeSize*** (in micrometers), the rule is that we store this value separately. The Pinhole is track specific. I.e. channels of

				the same track need to have the same value here. [units:micrometers].
PinholeGeometry	A	restriction of xs:string	Circular	The geometry of the pinhole, either circular or rectangular. One of *Circular*, *Rectangular*, ***Other***
Fluor		xs:string		The Fluor element is used for fluorescence images. This is the name of the fluorophore used to produce this channel [plain text string]. This element is just for informative purposes. The fluorochrome is far better identified by the DyeId. However - if you do not have a DyeId at hand, you may use this field in order to give at least an informative string.
NDFilter		xs:float		The ***NDfilter*** element is used to specify the combined effect of any neutral density filters used. [units:optical density expressed as a PercentFraction]
PockelCellSetting		xs:integer		The ***PockelCellSetting*** used for this channel. is the amount the polarization of the beam is rotated by. [units:none]
Color	B	restriction of xs:string	#5566ff	The original color of the channel (the color that was defined by the dye or the user before the acquisition).
ExposureTime	B	restriction of xs:string pattern value="\s*((\d+)|(\d+-\d+))\s*"	4000	Exposure Time used to acquire this channel for informative purposes. The value may be given as a single number or a range. Examples: "4000" "89944" "887-1100" "100-1000" This is element gives just an informative value for the exposure-time used to acquire this channel. It must not be understood to have the meaning of "exposuretime is constant for all pictures in this channel". If the exposure time is not constant, then a range may be given here. If it does not apply at all (e. g. because no CCD-camera or similar was used as detector), leave it out. [units:nanoseconds].
SectionThickness		xs:double	12.3	For SIM this gives the thickness of the section. [unit: micrometers].
DetectorSetting		ChannelDectectorSettings	<complex content/>, see ChannelDectectorSettings	Supplements or overrides detector parameters for acquisition of this particular channel.
LightSourceSettings		ChannelLightSourcesSettings	<complex content/>, see ChannelLightSources	Supplements or overrides light sources parameters for acquisition of this particular channel.

			Setting	
LightPath		ChannelLight Path	<complex content/>	For the moment, the LightPath is track specific. I.e. channels of the same track need to have the same value here.
FilterSet		FilterSetRef	<complex content/>	Refers to an instance of InstrumentFilterSet in Information/Instrument/FilterSets.
LaserScanInfo		ChannelLaser ScanInfo	<complex content/>	Here we find information how the laser operated when scanning the field.
Reflector		xs:string		
CondenserContrast		xs:string		
NACondenser		xs:double	0.45	
Ratio		Ratio		The ratio between two active channels.

(1) *"Epifluorescence" in this context just describes the fact that the objective is used to bringzhe fluorescence inducing light ("the illumination") to the specimen (in contrast to transmitted illumination e.g.). This does not neccessarily require that we do a fluorescence acquisition, maybe "lightthroughobjective" would be less confusing...

6.7.3.1 ChannelDectectorSettings complex type

Element	req	Type (XSD)	Sample	Description
Detector		DetectorRef	<complex content/>	Refers to an instance of ***InstrumentDetector*** in Information/Instrument/Detectors.
Binning		restriction of xs:string	1x1	Represents the number of pixels that are combined to form larger pixels. One of *1x1, 2x2, 4x4, 8x8,* ***Other***
Gain		xs:double	1.45	The Gain of the detector. [units:none]
DigitalGain		xs:double	0.78	The digital Gain of the detector. [units:none]
Offset		xs:double	0.6	The Gain Offset of the detector. [units:none]
EMGain		xs:double	0.2	The EM Gain
Voltage		xs:double	6.78	The Voltage of the detector. volts[V]
ReadOutRate		xs:double	10.5	The speed at which the detector can count pixels. Units of ***ReadOutRate*** is MHz. This is the bytes per second that can be read from the detector (like a baud rate). megahertz[MHz]
UseBrightnessContrastCorrection		xs:boolean	true	The brightness and contrast correction for stacks and z-scans was active during acquisition and is defined by sets of variable values for AOTF power, PMT gain and detector amplifier gain and offset for the positions of the focus drive. Default: ***false***.

6.7.3.2 ChannelLightSourcesSettings complex type

Element	req	Type (XSD)	Sample	Description
LightSourceSettings		ChannelLightSourceSettings	<complex content/>, see ChannelLightSourceSettings	A sequence of ***LightSourceSetting*** elements.

6.7.3.3 ChannelLightSourceSettings complex type

Channel acquisition specific overrides for the Light Source.

Element	req	Type (XSD)	Sample	Description
`LightSource`		LightSourceRef	<complex content/>	Refers to an instance of ***InstrumentLightSuzrce*** in Information/Instrument/LightSources.
`Wavelength`		xs:double	420	The Wavelength of the light source. nanometres[nm].
`Attenuation`		xs:double	0.45	The Attenuation of the light source [units:none] A fraction, as a value from 0.0 to 1.0. A value of 0.0 means "no attenuation", and 1.0 means "all light was blocked". So, this is 1 - light_intensity_after_attenuation / light_intensity_before_attentuation.
`Intensity`		xs:double	0.78	The intensity of the light source. The intensity might be set in percent or in Volt; the unit is part of the string value.

6.7.4 Information / Image / Dimensions / Tracks

Element	req	Type (XSD)	Sample	Description
`MultiplexType`		restriction of xs:string	Frame	Specifies when a switch to the next track is done. Possible values are: • ***Frame*** - after a frame, • *Line* - after a line. The value is the same for all tracks in a recording.
`MultiplexOrder`		<ananymous complex type>	<complex content/>	The switch order of tracks. This type consists of a sequence of Track references (via Id)
`Track`		DimensionTrack	<complex content/>, see Information / Image / Dimensions / Tracks / Track	Individual track information

6.7.5 Information / Image / Dimensions / Tracks / Track

XSD complex type **DimensionTrack**.

Attribute	req	Type (XSD)	Sample	Description
Id	Y	xs:string	Track:1	Unique ID within a list of siblings. Recommended format is Track:<nr>.
Name		xs:string	Bleaching, Phase 1	The name of the track as specified by the user.

Element	req	Type (XSD)	Sample	Description
`ChannelRefs`	Y	ChannelRef	<complex content/>	Sequence of references to the contained channels (Information / Image / Dimensions/Channels) via ID.

6.7.6 Information / User

Contains user and site information.

Attribute	req	Type (XSD)	Sample	Description
Id	Y	xs:string	User:1	Unique ID within a list of siblings. Recommended format is User:<nr>.

Element	req	Type (XSD)	Sample	Description
DisplayName		xs:string	John Doe	Name to be displayed in user interface elements.
FirstName		xs:string	John	First name, sometimes called christian name or given name or forename.
MiddleName		xs:string	Mc.	Any other names.
LastName		xs:string	Doe	A person's last name sometimes called surname or family name.
Email		xs:string	john@doe.com	A person's email address
Institution		xs:string	Carl Zeiss Munich	A person's institution.
UserName		xs:string		This is the username of the experimenter (in a 'logon' or 'database' sense).
Phone		xs:string	+49 989 7898	Phone number in stanard notation.
Fax		xs:string	+1 8989 8998	Fax number in standard notation
Address		xs:string	Main Street 2	Street and number.
City		xs:string	Munich	User's city.
Country		xs:string	Germany	User's country.
State		xs:string	Bavaria	State (required for US address specification)

6.7.6.1 Sample: User

```
      <User Id="User1">
        <DisplayName>John Mc. Doe</DisplayName>
        <FirstName>John</FirstName>
        <LastName>Doe</LastName>
        <Email>john@doe.com</Email>
        <Institution>Carl Zeiss Munich</Institution>
        <MiddleName>Mc.</MiddleName>
        <Phone>+4989-234578</Phone>
..
      </User>
```

6.7.7 Information / Document

Contains general "document" information.

Element	req	Type (XSD)	Sample	Description
Name		xs:string	FluoCells	Specific name given to this image.
Author		xs:string	John Doe	The document's author (typically the experimenter or image generator).
UserName		xs:string	jdoe	User name in unique format, e.g. system account name.

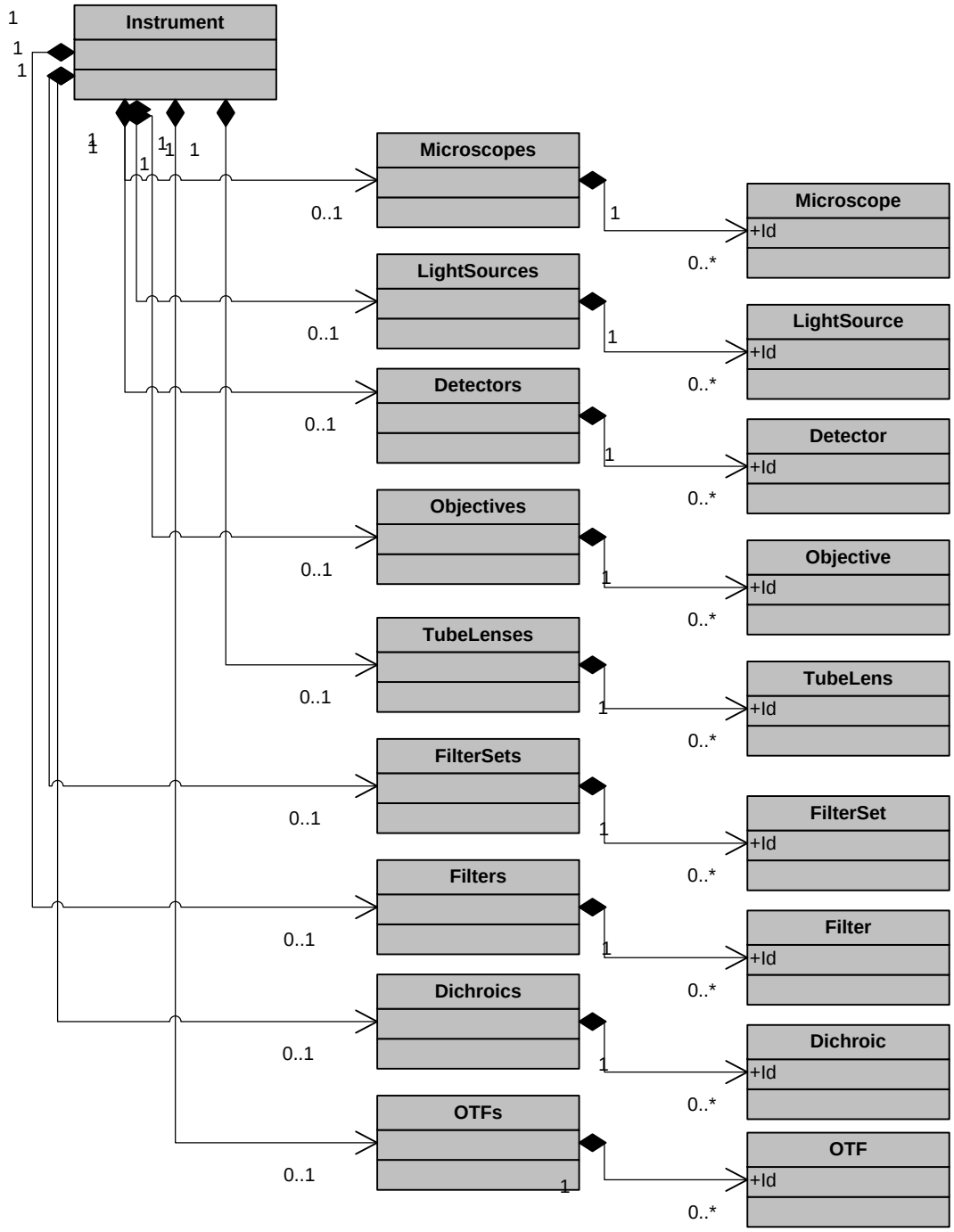
SubType		xs:string	Image	Document sub-type to further specifiy the individual generation process and usage of this image. Samples are "PSF", "SIM". The default is an empty string or "Image".
Title		xs:string	My first experiment	A reasonable title given to the image.
CreationDate		xs:dateTime	2001-01-28T00:00:05	Date and time when the document was created.
Description		xs:string	Cells in a …	Advanced (may be lengthy) description of what was acquired and in which context.
Thumbnail		xs:string	cells.jpg	A thumbnail to be displayed as the image's icon / symbol (application / organization specific location).
Comment		xs:string	Annotated by XY	Additional comments.
Rating		xs:integer	1	A rating value from 0..3.
Keywords		xs:string	cells john	A list of keywords for full text searches etc..

6.7.7.1 Sample: Document

```
<Document>
  <CreationDate>2001-01-28T00:00:00</CreationDate>
  <Description>This is a sample image by WBa</Description>
  <Thumbnail href="images/thumb1"/>
  <Comment>This is a comment</Comment>
  <Keywords>one two three</Keywords>
</Document>
```

6.7.8 Information / Instrument

Decribes the Hardware.



Element	req	Type (XSD)	Sample	Description
`Microscopes`		Microscopes, InstrumentMicroscope	see Information/Instrument/Microscopes/Microscope	Defines the microscopes used. Typically there is only a single element of this type.
`LightSources`		LightSources InstrumentLightSource	see Information/Instrument/LightSources/Light	A collection of available light sources.

			Source	
Detectors		Detectors InstrumentDetector	see Information/Instrument/Detectors/Detector	A collection of available light detectors.
Objectives		Objectives InstrumentObjective	see Information/Instrument/Objectives/Objective	A collection of available objectives.
TubeLenses		TubeLense InstrumentTubeLens	see Information / Instrument / TubeLenses / TubeLens	A collection of available tube lenses.
FilterSets				A collection of available filter sets.
Filters				A collection of available filters.
Dichroics				A collection of available dichroics.
OTFs				A collection of available OTFs.

6.7.8.1 Information / Instrument / Microscopes / Microscope

XSD Complex type: **InstrumentMicroscope**.

Describes the microscope (stand).

Attribute	req	Type (XSD)	Sample	Description
Id	Y	xs:string	Microscope:1	Unique ID within a list of siblings. Recommended format is Microscope:<nr>.

Element	req	Type (XSD)	Sample	Description
Manufacturer		Manufacturer	<complex content/>	Information about the manufacturer of this hardware.
System		xs:string	LSM700	A list of components where each part is divided by a colon. The purpose is to name the whole system.For a confocal system the schema is "[LsmName][, RtScannerName][, CameraName][, StandName].
Type		xs:string	Upright	The type of the microscope. One of *Upright*, *Inverted*, *Dissection*, *Electrophysiology* or ***Other***.

6.7.8.2 Information / Instrument / LightSources / LightSource

XSD Complex type: **InstrumentLightSource**.

The lightsource for the instrument. An instrument may have several light sources.

The type of lightsource is specified by one of the child-elements which are 'Laser', 'Filament', 'Arc' or 'LightEmittingDiode'.

Each of the light source types has its own ***Type*** attribute to further differentiate the light source (eg, Nd-YAG for Laser or Hg for Arc).

Attribute	req	Type (XSD)	Sample	Description
Id	Y	xs:string	LightSource:1	Unique ID within a list of siblings. Recommended format is LightSource:<nr>.

				A LightSource ID must be specified for each light source, and the individual light sources can be referred to by their LightSource IDs (e.g. from Channel)
Name		xs:string	AttoArc	The name of this hardware component.

Element	req	Type (XSD)	Sample	Description
Manufacturer		Manufacturer	<complex content/>	Information about the manufacturer of this hardware.
Power		xs:float	23.5	The light-source power, units milliwatts[mW]
LightSourceType		choice of complex types	<Laser>..</Laser>	The type of the light source. One of complex types *Laser, Filament, Arc, LightEmittingDiode*

6.7.8.3 Information / Instrument / Detectors / Detector

XSD Complex type: **InstrumentDetector**.

The detector used to capture the image. The Detector ID can be used as a reference within the ***Channel*** element in the ***Image*** element.

Attribute	req	Type (XSD)	Sample	Description
Id	Y	xs:string	Detector:1	Unique ID within a list of siblings. Recommended format is Detector:<nr>.
Name		xs:string		The name of this hardware component.

Element	req	Type (XSD)	Sample	Description
Manufacturer		Manufacturer	<complex content/>	Information about the manufacturer of this hardware.
Gain		xs:float	1.2	The Detector Gain for this detector, as a float. [units:none]
Voltage		xs:float	5.6	The Voltage of the detector (e.g. PMT voltage) as a float. volts[V]
Offset		xs:float	0.1	The Detector Offset. [units:none]
Zoom		xs:float	1.78	The Zoom or "Confocal Zoom" or "Scan Zoom" for a detector. [units:none]
AmplificationGain		xs:float	0.5	Gain applied to the detector signal. This is the electronic gain (as apposed to the inherent gain) that is set for the detector. [units:none]
AmplificationOffset		xs:float	0.1	Offset applied to the detector signal. [units:none]
Type		restriction of xs:string		Type of the detector, one of CCD, IntensifiedCCD, AnalogVideo, PMT, Photodiode, Spectroscopy, LifetimeImaging. CorrelationSpectroscopy, FTIR, EMCCD, APD, CMOS, EBCCD, Other
Adapter		DetectorAdapter	<complex content/>	If the detector type is supposed to have an adapter, this is the respective adapter data.

				E.g. a CCD can have a camera adapter with a magnification, a PMT or a Photodiode doesn't have an adapter. DetectorAdapter contains the elements Manufacturer, Magnification and CustomAttributes.
GammaDefault		xs:float	1.1	The default gamma value. This value will be set by the Reset DisplaySetting function. [units:none] default is 1.0.

6.7.8.4 Information / Instrument / Objectives / Objective

XSD Complex type: **InstrumentObjective**

A description of the microscope's objective lens.
Required elements include the lens numerical aperture, and the magnification, both of which a floating point (real) numbers.

The values are those that are fixed for a particular objective: either because it has been manufactured to this specification or the value has been measured on this particular objective.

Attribute	req	Type (XSD)	Sample	Description
Id	Y	xs:string	Objective:1	Unique ID within a list of siblings. Recommended format is Objective:<nr>.
Name		xs:string		The name of this hardware component.

Element	req	Type (XSD)	Sample	Description
Manufacturer		Manufacturer	<complex content/>	Information about the manufacturer of this hardware.
Correction		restriction of xs:string	PlanApo	This is the type of correction coating applied to this lens. One of… *UV, PlanApo, PlanFluor, SuperFluor, VioletCorrected, Achro, Achromat, Fluor, Fl, Fluar, Neofluar, Fluotar, Apo, PlanNeofluar,* ***Other***
Immersion		restriction of xs:string	Oil	This is the type of immersion medium the lens is designed to work with. It is not the same as 'Medium' in Information/Image/ObjectiveSettings (a single type) as here Immersion can have compound values like 'Multi'. One of … *Oil, Water, WaterDipping, Air, Multi, Glycerol,* ***Other***
ImmersionRefractiveIndex		xs:float	1.234	The refractive index of the immersion. [units:none] If this field is empty, the refractive index is assumed to be the default refractive index of the specified immersion.

LensNA	AB	xs:float	0.7	The numerical aperture of the lens expressed as a floating point (real) number. Expected range 0.02 - 1.5 [units:none] The depth of focus can be retrieved via the formula: depthOfField = 0.55 / (LensNA * LensNA).
NominalMagnification	AB	xs:float	40	The magnification of the lens as specified by the manufacturer - i.e. '40' is a 40x lens. [units:none].
CalibratedMagnification		xs:float	39.978	The magnification of the lens as measured by a calibration process - i.e. '39.987' for a 40x lens. [units:none].
WorkingDistance	AB	xs:float	105.534	The working distance of the lens expressed as a floating point (real) number. Units are microns[um].
Iris		xs:boolean	false	Records whether or not the objective was fitted with an Iris. [flag]
PupilGeometry	AB	restriction of xs:string	Circular	Records what type of phase-rings ("Phasenringe") the objective has. One of… *Circular*, *Annular*, *PhaseRing1*, *PhaseRing2*, *PhaseRing3*, ***Other***

6.7.8.5 Sample: InstrumentObjective

```
<Objective Id="Objective:0">
        <Manufacturer>
        <Model>Plan Neofluar 40/1.30 Oil Ph3 (DIC III)</Model>
        </Manufacturer>
        <LensNA>0.7</LensNA>
        <Immersion>Oil</Immersion>
        <ImmersionRefractiveIndex>1.234</ImmersionRefractiveIndex>
        <NominalMagnification>40</NominalMagnification>
        <WorkingDistance>105.534</WorkingDistance>
        <PupilGeometry>Other</PupilGeometry>
</Objective>
```

6.7.8.6 Information / Instrument / TubeLenses / TubeLens

XSD Complex type: **InstrumentTubeLense**

A description of the tube lens.

Required elements include the magnification.
A tube lens might be a tube lens, an optovar lens or a Bertrand lens.

Attribute	req	Type (XSD)	Sample	Description
Id	Y	xs:string	TubeLens:1	Unique ID within a list of siblings. Recommended format is TubeLens:<nr>.
Name		xs:string		The name of this hardware component.

Element	req	Type (XSD)	Sample	Description
Manufacturer		Manufacturer	<complex content/>	Information about the manufacturer of this hardware.
Magnification	Y	xs:float	1.4	The magnification of the lens as specified by

				the manufacturer [units:none].
Type		(restriction of) xs:string	Optovar	The type of the lens in the revolver element (might be a tube lens, an optovar lens or a Bertrand lens). One of.. (no restriction up to now!) Optovar, TubeLens, BetrandLens

6.7.8.7 Sample: InstrumentTubeLens

```
<TubeLens Id="TubeLens:1">
        <Magnification>10.9</Magnification>
</TubeLens>
```

6.7.9 Information / Application

Main application specific information. Expandable via CustomAttributes.

Element	req	Type (XSD)	Sample	Description
Name	Y	xs:string	AimApplication	Name of the application (executable) which created the image application.
Version		xs:float	1.1	Version of the application which created the image

6.7.10 Information / Processing

This node contains specific data for special processing data requirements, e.g.

- DFT
- DCV
- Colocalization
- Stitching

The persisted information may be obtained by the processing function upon first usage. It uses other metadata like Information/Instrument to get the values, provides modification features in the GUI and stores the results for later retrieval.

6.7.10.1 Sample: Processing

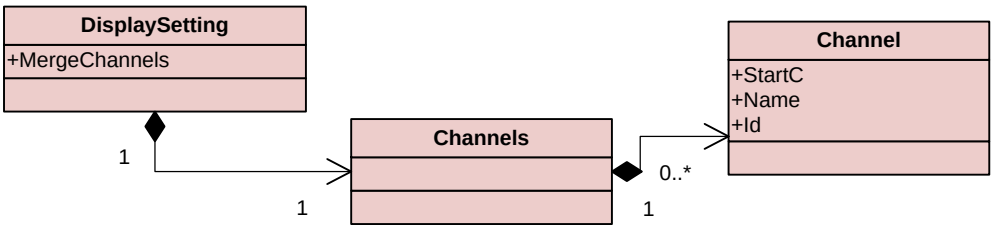
```
<Processing>
    <DFT>
      <BlockShift>true</BlockShift>
      <ImageType>Basic</ImageType>
      <ProcessingDimension>2</ProcessingDimension>
      <RefPixelType>3</RefPixelType>
      <ThirdDimension>4</ThirdDimension>
      <WindowMode>4</WindowMode>
    </DFT>

    <PSF>
      <AnticipatedPolarHeight>10</AnticipatedPolarHeight>
      <AxialResolution>3.5</AxialResolution>
      <Defocus>10</Defocus>
      <DesignCoverglassThickness>1e-3</DesignCoverglassThickness>
      <AnticipatedPolarWidth>3</AnticipatedPolarWidth>
      <Dimension>3</Dimension>
      <UsedImmersionIndex>3</UsedImmersionIndex>
      <Type>3</Type>
      <Source>3</Source>
      <WorkingDistance>3.4</WorkingDistance>
      <ZStackDirection>3</ZStackDirection>
```

```
</PSF>
```

6.8 DisplaySetting

Contains generic multi-channel display adjustments (Low/High/Gamma) and advanced information like channel color mapping via palettes (lookup – tables).



The main use of DisplaySetting is to parametrize the GUI elements used in image visualization.

Element	req	Type (XSD)	Sample	Description
MergeChannels		xs:boolean	true	Indicates whether display should be in multichannel (merged) mode.
ShowRangeIndicator		xs:boolean	true	If true, shows display the image in "range idicator" mode to visualize under- and overexposure
Channels	Y	Channels Channel	<complex content/>	One entry for each contained image channel

6.8.1 DisplaySetting / Channels / Channel

Defines the standard / current display settings for a single channel.

Attribute	req	Type (XSD)	Sample	Description
Id		xs:string	Channel:0	Unique channel ID must match the identifiers in Information / Dimension / Channels to uniquely address a channel. Id not specified, the correspondance is via collection index.
Name		xs:string		The (user-defined) name of this channel.

Element	req	Type (XSD)	Sample	Description
Low		xs:double	0.1	The normalized low (=black) value of the mapping range. Default value is 0.0.
High		xs:double	0.5	The normalized high (=white) value of the mapping range. Default value is 1.0.
Gamma		xs:double	1.0	The gamma value to be applied to the mapping range. Default value is 1.0.
IsAutoApplyEnabled		xs:boolean	false	Indicates whether the command specified in ***AutoApplyMode*** is applied each time a new image subset is selected (e.g. when the player is running). Default valus is false.
AutoApplyMode		restriction of xs:string	MinMax	Mode to be applied when the ***AutoApplyMode*** is set to true. One of ***MinMax***, *BestFit*.
LowerBestFitThreshold		xs:double	0.1	The lower threshold for the ***BestFit*** operation.

				The value is useful e.g. to skip large nearly black areas with some noise. Default is 0.1.
UpperBestFitThreshold		xs:double	0.1	The upper threshold for the ***BestFit*** operation. The value is useful e.g. to skip large white or nearly white areas. Default is 0.1.
Mode		restriction of xs:string	Spline	The mode: can be *Spline, Ramp or **None***. Default is *None*.
Points		xs:string	0.1, 0.7 0.3, 0.6	If ***Mode*** is Spline, these are the Spline Points. If mode is *Ramp*, these are the Points for the Ramp mode. If mode is *None*, there is supposed to be no Points node, i.e. Points are ignored.
Description		xs:string		User defined description of the channel.
DyeName		xs:string		The original Dye name taken from the acquisition information when the image was acquired.
ShortName		xs:string		The information displayed in small GUI elements like channel buttons. If not set, ***ShortName*** is derived from the ***Name*** or the ***DyeName***.
Color		(restriction of) xs:string	#FF0077	The color in which to display the channel if ***ColorMode*** = Color. The string must be in #RRGGBB or #AARRGGBB format where R,G or B are the Red, Green and Blue components in hexadecimal notation.
ColorMode		restriction of xs:string	Color	The color mode in which to display the channel. One of Indeterminate, None, Color, Palette, Dye, Custom. If set to *Color*, uses the ***Color*** value, if set to *Palette* uses the ***PaletteName*** name to select a predefined system palette. *None* means: use original channel color. Currently, the other values are not used.
OriginalColor		(restriction of) xs:string	#FF0077	The original color of the channel, i.e. the color the channel had on the most recent save event.
IsSelected		xs:boolean	true	If the DisplaySetting is in ***MergeChannels*** mode, the channel is only displayed if ***IsSelected*** is true.
PaletteName		xs:string	dawn	If ***ColorMode*** = *Palette*, the channel is displayed with the lookup-table (LUT) defined by the ***PaletteName***.
ChannelWeight		xs:float	1.0	The channel weight (ratio among all selected channels).
ChannelUnit		<anonymous>	<complex content/>	The ***ChannelUnit*** contains scaling factor, offset and unit for each image channel. The data are currently used by images with ion

				concentration data. The display value is calculated by DisplayValue = Factor * PixelIntensity / MaxPixelIntensity + Offset. where "MaxPixelIntensity" is the maximum possible pixel intensity for the data type (4095 or 255).

6.8.1.1 Sample: DisplaySetting

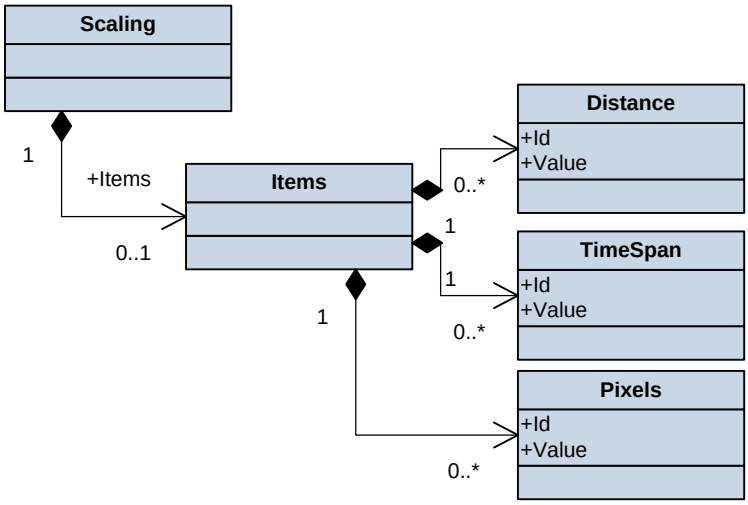
```
<DisplaySetting>
 <Channels>
    <Channel StartC="0" Name="FITC" Id="Channel:0">
      <Color>#00FF00</Color>
      <Gamma>0.3</Gamma>
      <IsAutoApplyEnabled>true</IsAutoApplyEnabled>
      <Low>0.0</Low>
      <ColorMode>Color</ColorMode>
      <LowerBestFitThreshold>0.1</LowerBestFitThreshold>
      <PaletteName>dawn</PaletteName>
      <Weight>1.0</Weight>
    </Channel>

    <Channel StartC="1" Name="DAPI" Id="Channel:1">
      <Color>#FF00FF</Color>
      <Gamma>0.3</Gamma>
      <DyeName>Dye1</DyeName>
      <IsAutoApplyEnabled>true</IsAutoApplyEnabled>
      <ColorMode>Color</ColorMode>
      <IsSelected>true</IsSelected>
      <Weight>1.0</Weight>
    </Channel>

    <Channel StartC="2" Name="Rhodamin" Id="Channel:2">
      <Color>#FF0000</Color>
      <Gamma>0.3</Gamma>
      <DyeName>Dye1</DyeName>
      <IsAutoApplyEnabled>true</IsAutoApplyEnabled>
      <Low>0.0</Low>
      <IsSelected>true</IsSelected>
      <High>0.7</High>
      <PaletteName>dawn</PaletteName>
      <Mode>Spline</Mode>
      <Points>0.1,0.09 0.24,0.54 0.63,0.40 0.79,0.90</Points>
      <Weight>1.0</Weight>
    </Channel>
  </Channels>
 </DisplaySetting>
```

6.9 Scaling

Scaling is a collection of **UnitItems** – each representing an image dimension via Key (string).



In addition, scaling carries some information about some required parameters to implement the “automatic scaling” feature (select a scaling froma predefined based on the current hardware setting).

The optional **AutoScaling** node enables usage of this scaling for automatic scaling by providing elements that can be compared to the actual hardware state.

The value of the scaling items specify the units/index, or units/pixel in case of X and Y index, e.g.

```
<Distance Id="X">
  <Value>1.21e-6</Value>
</Distance>
```

means that the image has a scaling of 1.21 micrometers (10^{-6} m) per pixel.

Element	req	Type (XSD)	Sample	Description
`AutoScaling`		AutoScalingSettings	<complex content/>	Additional data to use this scaling for an Automatic scaling feature. Reserved for internal use.
`Items`	Y	choice of complex types *DistanceUnitItem*, *TimeSpanUnitItem* *PixelUnitItem*	<complex content/>	One item for each dimension, Accepted element names for the relted complex types are: *Distance, TimeSpan or Pixels*

6.9.1.1 DistanceUnitItem complex type

Specifies a distance in meters.

Attribute	req	Type (XSD)	Sample	Description
`Id`	Y	xs:string	X	Identifier of the related dimension. One of ***X,Y,Z,T*** but may also specifiy other dimensions if units can be expressed in one of the supported UnitItems.

Element	req	Type (XSD)	Sample	Description
`Value`	Y	xs:double	1e-4	Value in meters [m].

DefaultUnitFormat		xs:string	um	Optional preselection of a display unit in GUI elements One of… ***m***, *cm, mm, u, µ, um, µm, nm, pm, i, inch, mil*.
IsReciprocal		xs:boolean	false	If true, shows values as reciprocal units, e.g. 1 m shows as 1 m-1.
Origin		xs:double	0.0	Specifies an alternate origin (default is 0.0), used e.g. for rulers.
Direction		xs:integer	0	Optionally defines the scaling direction as follows: 1 positive, -1 negative, 0 (default) undefined.

6.9.1.2 TimeSpanUnitItem complex type

Specifies a time span in seconds.

Also supports: **Id**, **IsReciprocal**, **Origin** and **Direction** – see **DistanceUnitItem**.

Element	req	Type (XSD)	Sample	Description
Value		xs:double	1e-4	Value in seconds [s].
DefaultUnitFormat		xs:string	um	Optional preselection of a display unit in GUI elements One of… ***s***, *ms, us, µs, ns, ps*.

6.9.1.3 PixelUnitItem complex type

Specifies pixel (or "no") scaling. For use in a Scaling context, the ***Value*** is normally 1.0.

Also supports: Id, IsReciprocal, Origin and Direction – see DistanceUnitItem

Element	req	Type (XSD)	Sample	Description
Value		xs:double	1e-4	Value in pixels [px].
DefaultUnitFormat		xs:string	um	Optional preselection of a display unit in GUI elements One of… ***px***, *mpx*.

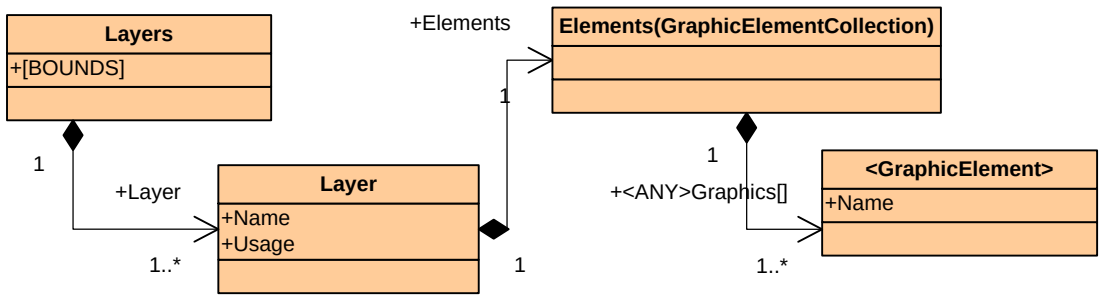
6.9.1.4 Sample:Scaling

```
<Scaling>
  <AutoScaling>
    <CameraFramePixelDistance>1</CameraFramePixelDistance>
    <CameraFrameBinning>1</CameraFrameBinning>
    <CameraAdapterMagnification>1</CameraAdapterMagnification>
    <Optovar>1</Optovar>
    <ReflectorMagnification>1</ReflectorMagnification>
    <Objective>Plan-Neofluar 1.25x/0.04</Objective>
  </AutoScaling>
  <Items>
    <Distance Id="X">
      <DefaultUnitFormat>um</DefaultUnitFormat>
      <Origin>10.1</Origin>
      <Value>1.21e-6</Value>
    </Distance>
    <Distance Id="Y">
```

```
      <DefaultUnitFormat>um</DefaultUnitFormat>
      <Direction>-1</Direction>
      <Value>1.22e-6</Value>
    </Distance>
    <TimeSpan Id="Z">
      <DefaultUnitFormat>ms</DefaultUnitFormat>
      <Value>1.22e-3</Value>
    </TimeSpan>
  </Items>
</Scaling>
```

6.10 Layers

A collection of (graphical) layers for overlays.

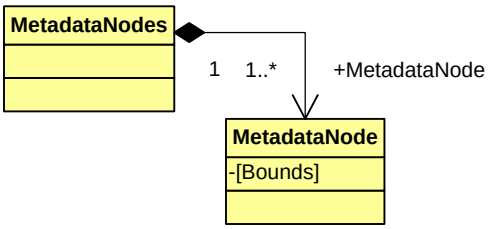


6.10.1.1 Sample: Layers

```
<Layers>
  <Layer Name="Layer1">
    <Usage>Annotation</Usage>
    <IsProtected>false</IsProtected>
    <LayerFlags>1</LayerFlags>
    <Elements>
      <Line Id="7">
        <Geometry>
          <X1>32.067510548523217</X1>
          <Y1>167.93248945147678</Y1>
          <X2>167.93248945147678</X2>
          <Y2>232.06751054852322</Y2>
        </Geometry>
      </Line>
      <Rectangle Id="10">
        <Geometry>
          <Left>215</Left>
          <Top>149</Top>
          <Width>170</Width>
          <Height>100</Height>
        </Geometry>
      </Rectangle>
      <Bezier Id="56">
        <Geometry>
          <Points>572.417721518987,302.95358649789 691.405063291139,304.64135021097
696.46835443038,432.911392405063 555.540084388186,483.544303797468</Points>
        </Geometry>
      </Bezier>
    </Elements>
  </Layer>
```

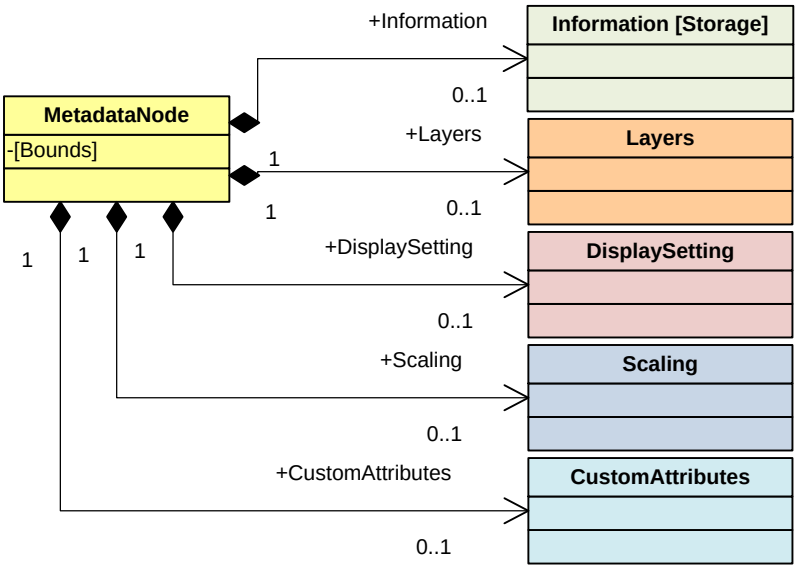
6.11 MetadataNodes

A collection of nodes, each containing metadata for an image subset.



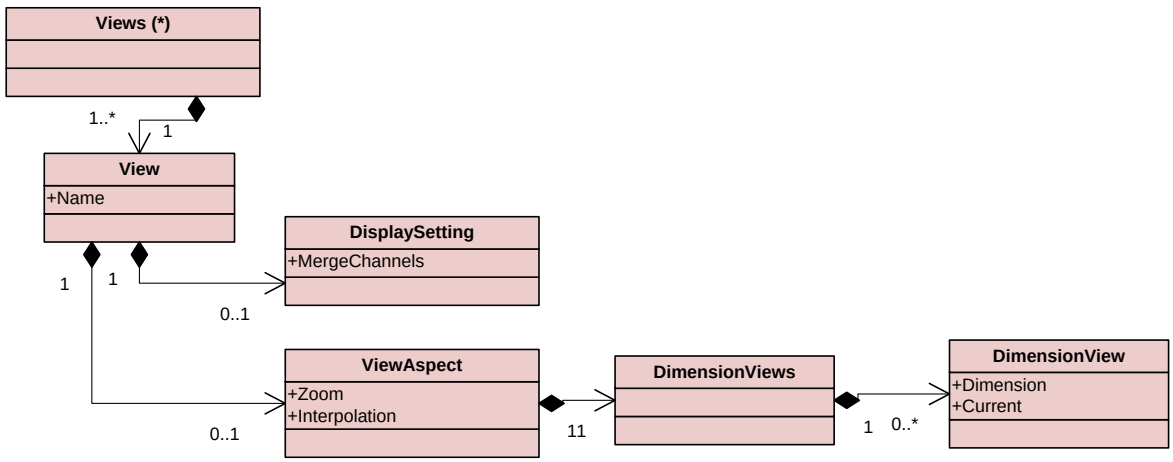
6.11.1 MetadataNode

Contains Metadata for a specific Image subset. The elements in this node are the same as the global information nodes in the **Metadata** element. The main usage is to define additional data for a given subset or information data overriding the default values.



6.12 Views

The **Views** tree contains a collection of various named views defining a specific aspect of the image (e.g. T2, Z5) to set the displayed position within a multi-dimensional image and optional display settings.



6.12.1.1 Sample: View

```
<View Id="Default" Name="Default">
   <ViewAspect>
     <DimensionViews>
       <DimensionView Id="Z">
         <Current>10</Current>
       </DimensionView>

       <DimensionView Id="T">
         <Current>5</Current>
       </DimensionView>
     </DimensionViews>

   </ViewAspect>

   <DisplaySetting>
     <Channels>
       <Channel Id="Channel:0">
         <Color>#220077</Color>
         <High>0.7</High>
       </Channel>

       <Channel Id="Channel:1">
         <Color>#ff0077</Color>
         <PaletteName>dawn</PaletteName>
         <Weight>1.0</Weight>
       </Channel>

       <Channel Id="Channel:2">
         <Gamma>0.3</Gamma>
       </Channel>

       <Channel Name="TexasRed" Id="Channel:4">
         <Color>#CC00FF</Color>
       </Channel>

     </Channels>
   </DisplaySetting>
 </View>
```