i3Dr Data Storage considerations

Configuration

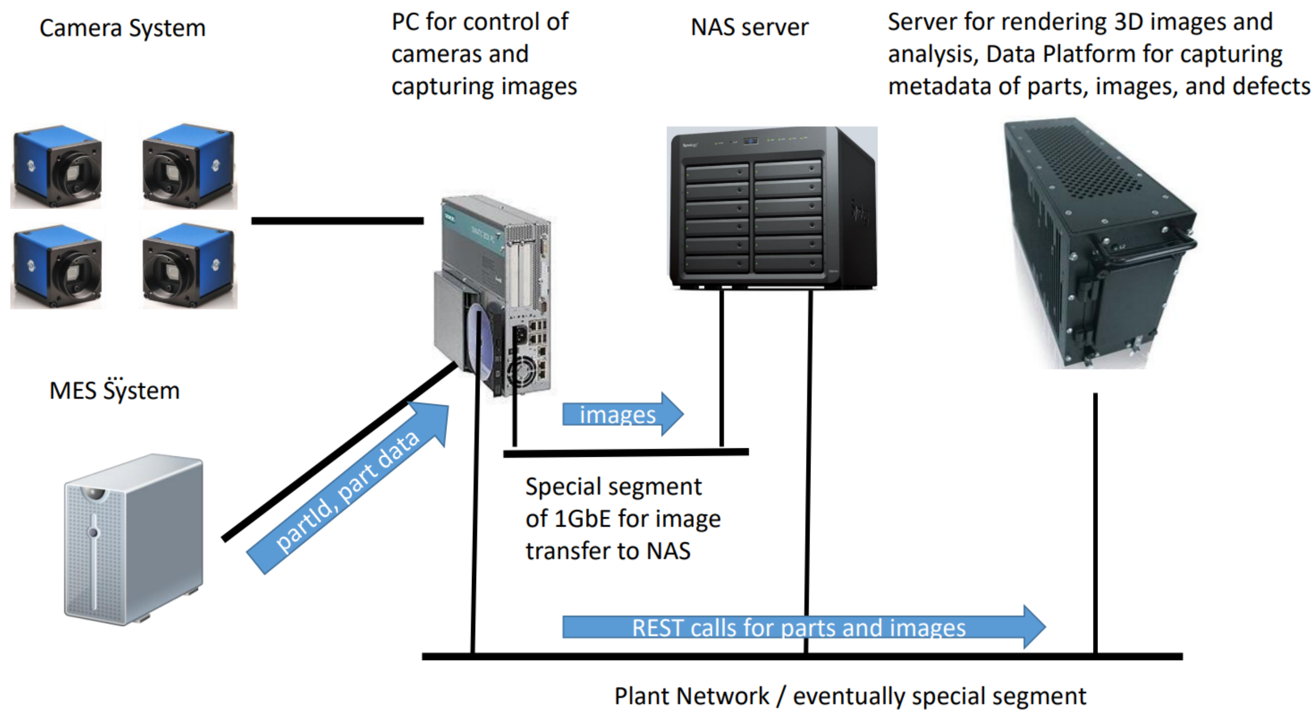
There are four major components of the configuration:

- **Cameras + Camera system**
  Captures raw images and stores them on NAS
- **NAS**
  stores image data
- **Image analysis system**
    - evaluates images and generate failure list
    - It needs to be checked if the function can run directly on camera system
- **Data Platform**
  database with metadata of all parts, images, and failures

Data Flow between Camera System, Image Analysis System and Data Platform

The camera system and the image analysis system send information to the data platform via REST calls. A REST call is a simple HTTP-request containing structured information in JSON-format and can be easily sent via nearly all programming languages like Java, C#, Python,...

- First of all, there must be information to which part the actual images belong. In the use case of voestalpine this part id will be sent from a MES system to the camery system. It his highly recommended to add additional information to the part-Id, like steel grade, heat number etc. The detailed description of this call is in Meta-Data for Produced Parts below.
- For each **captured image, meta data** must be sent in a similar way - see Image Data below. Minimum requirement for these information is, to which part the image belongs, the position on the part and the area of the part the image covers.
- If a **failure** is detected in the image analysis, the information must be sent to data platform as well. See Image analysis Results below.


Camera System
PC for control of cameras and capturing images
NAS server
Server for rendering 3D images and analysis, Data Platform for capturing metadata of parts, images, and defects
MES System
partId, part data
images
Special segment of 1GbE for image transfer to NAS
REST calls for parts and images
Plant Network / eventually special segment

JSON-Interfaces

Meta-Data of Produced Parts (Source: Camera or MES system)

For every part in production - such as a plate or a coil - information needs to be provided to the iMATH data platform - This includes

- Unique Part ID - mandatory
- Technological information - Optional
  - Includes parameters of the raw material, rolling schedule, etc...

The iMATH platform provides a REST interface to which the camera system needs to send part data before the beginning of part production - The following structure is suggested for the JSON body of the request.

Part Data Example Body

```
{
        "timestamp": 1516193959559,
        "part_id": "Part1234",
        "source": "Camera_Control_PC_Garret",
        "part_data": [
                {
                        "key": "steel_grade",
                        "value": "Grade01"
                },
                {
                        "key": "heat_number",
                        "value": "C1234566"
                },
                {
                        "key": "rolling_schedule",
                        "value": "Schedule1"
                },
                {
                        "key": "analysis",
                        "value": [
                                {
                                        "key": "C",
                                        "value": "0.2"
                                },
                                {
                                        "key": "Mn",
                                        "value": "0.02"
                                }
                        ]
                }
        ]
}
```

Description of the Part-Data JSON-Body:

JSON-Property	Type	Constraints	Description
timestamp	integer	required: true	timestamp of the image (milliseconds since epoch)
source	string	required: true	Sender of the message, e.g. Camera control PC
part_id	string	required: true	identifier of the part displayed on the image(s)
part_data	array of objects	optional	Additional technological parameters. The objects in the list are key/value pairs and can be defined and adapted by the sender. So additional informaton can be included on the fly

The content of the REST-call will be inserted in the following database tables.

PART		
PART_ID	**TIMESTAMP**	**SOURCE**
Part1234	1516193959559	Camera_Control_PC_Garret

PART_DATA				
PART_ID	**KEY1**	**KEY2**	**KEY3**	**VALUE**
Part1234	steel_grade			Grade01
Part1234	heat_number			C1234566
Part1234	rolling_schedule			Schedule1
Part1234	analysis	C		0.2
Part1234	analysis	Mn		0.02

Due to the Key-Value-Structure, additional parameters can be added anytime to the part_data list in the REST-interface. They will be saved without requiring changes to the database.

Image Meta Data (Source: Camera System)

For every captured image the camera system needs to provide the following information

- Unique ID of the part
- Position on the part (Center or Upper-Left-Corner...)
- The area of the part displayed on the image

The iMATH platform provides a REST interface to which the camera system needs to send the required information for one or a list of images -The following JSON structure is suggested:

Example Body

```
{
        "part_id" : "Part1234",
    "value_id" : "Camera1"
        "source": "Camera_Control_PC_Garret",
        "values" : [
                { "value": "Part1234_1.1_2.2_1.jpg",
          "timestamp": "1516193959559",
          "position": [346.2,2],
          "dimension" : [5.2,1],
          "quality" : "1"
        },
                { "value": "Part1234_1.1_2.2_2.jpg",
          "timestamp": "1516193959559",
          "position": [246.2,5],
          "dimension" : [1.7,4],
          "quality" : "-1"
        }
         ],
        "qualifying_metadata" : [
                { "key": "key1",
          "value": "value1"
        },
                { "key": "key2",
          "value": "value2"
        }
         ]
}
```

The following table describes the properties of the example body in more detail

JSON-Property	Type	Constraints	Description
part_id	string	required: true	identifier of the part displayed on the image(s)
value_id	string		unique identification of the capture device (camera, sensor,...)
source	string	required: true	creator of the message, e.g. camera system
values	array of objects	required: true, min items: 1	the data of the images it is possible to include data for multiple images in one request as long as they refer to the same part and have the same meta data
value	string	required: true	in this case: name of the image
timestamp	integer	required: true	timestamp of the image (milliseconds since epoch)
position	array of numbers	required:true, min items: 1	position on the part where the image was taken positions in this array are expressed with one element (x coordinate), 2-element arrays (x and y coordinates) or 3-element arrays (x, y and z coordinates) Proposal: Because it is not so easy to define the position on the circumference of a rod, the y-position can be defined with 1,2,3,4 according the camera nummber
dimension	array of number-arrays	required: true, min items: 1	dimension of the area of the part, which is on the image - see position
quality	number	required: false	quality
qualitfying_metadata	array of key-value pairs	required: false	optional additional metadata as key value pairs

The advantage of this data structure is that it is generic and could be applied to other events / measurements too.

By defining values as an array, there is the possibility to send data for multiple images (of the same part and with same metadata) in one request.

Content of the REST-call will be inserted in the following database tables:

PART _IMAGE												
PART_ID	VALUE_ID	VALUE	PKID	SOURCE	POSITION_X	POSITION_Y	POSITION_Z	DIMENSION_X	DIMENSION_Y	DIMENSION_Z	TIMESTAMP	QUALITY

Part1234	Camera1	Part1234_1.1_2.2_1.jpg	1234244444	Camera_Control_PC_Garret	346.2	2		5.2	1		1516193959559	1
Part1234	Camera1	Part1234_1.1_2.2_2.jpg	1234244445	Camera_Control_PC_Garret	246.2	5		1.7	4		1516193959559	1

The unique id PKID is generated during insert.

PART_IMAGE_DETAILS		
PKID	**KEY**	**VALUE**
1234244444	**key1**	**value1**
1234244444	**key2**	**value2**
1234244445	**key1**	**value1**
1234244445	**key2**	**value2**

Image Analysis/ ML-Inference Results (Source: Image Analysis System)

The iMATH platform provides a REST interface to which the image analysis system sends the information for the failures detected on an image - The following JSON structure is suggested for the request:

Example Body

```
{
        "part_id": "Part1234",
        "source": "Analysis System",
        "value": "Part1234_1.1_2.2_1.jpg",
        "timestamp": "1516193959559",
        "failures": [
                {
                "id":           124355435321576
                        "failure": "4711",
                        "position": [
                                44.2,
                                17.4
                        ],
                        "dimension": [
                                5.2,
                                1
                        ],
                        "qualifying_metadata": [
                                {
                                        "key": "xxx",
                                        "value": "1"
                                },
                                {
                                        "key": "yyy",
                                        "value": "2"
                                }
                        ]
                },
                {
                "id":           124355435321578
                        "failure": "4712",
                        "position": [
                                33.2,
                                3
                        ],
                        "dimension": [
                                2.3,
                                1.1
                        ],
                        "qualifying_metadata": [
                                {
                                        "key": "xxx",
                                        "value": "1"
                                },
                                {
                                        "key": "yyy",
                                        "value": "2"
                                }
                        ]
                }
        ]
}
```

The following table describes the properties of the example body in more detail

Property	Type	Constraints	Description
part_id	string	required: true	identifier of the part that is displayed on the image(s)
source	string	required: true	analysis system
value	string	required: true	image name
timestamp	integer	required: true	timestamp of the analysis
failures	array of object	required: true, min items: 1	failure list of image
id	integer	required: true	unique id of failure
failure	string	required: true	failure type from defect catalogue
position	array of numbers	required:true, min items: 1	failure position (x,y,z) in image in [%]
dimension	array of number-arrays	required: true, min items: 1	failure dimension (x,y,z) in image in [%]
qualitfying_metadata	array of key-value pairs	required: false	optional additional metadata as key value pairs

The content of the REST-call will be inserted in the following database tables:

PART_*IMAGE_FAILURES*

PART_ID	VALUE	DEVICE	ID	FAILURE	POSITION_X	POSITION_Y	POSITION_Z	DIMENSION_X	DIMENSION_Y	DIMENSION_Z
Part1234	Part1234_1.1_2.2_1.jpg	Analysis System	124355435321576	4711	44,2	17,4		5.2	1	
Part1234	Part1234_1.1_2.2_2.jpg	Analysis System	124355435321578	4712	33.2	3		2.3	1.1	

PART_IMAGE_FAILURE_DETAILS

PART_ID	ID	KEY	VALUE
Part1234	124355435321576	**xxx**	**1**
Part1234	124355435321576	**yyy**	**2**
Part1234	124355435321578	**xxx**	**1**

Part1234	124355435321578	**yyy**	**2**