SKF @ptitude Observer v13.3 release summary

REQUIREMENTS

Software & FW requirements in Observer v.13.3

- Recommended IMx-1 GW firmware version – 4.0
- Recommended IMx-1 Sensor firmware version – 4.1
- Recommended IMx firmware version – 7.8
- Monitor Service requires 64-bit OS

Deprecated functions

- OPC Classic interface is deprecated and will be automatically converted to OPC UA

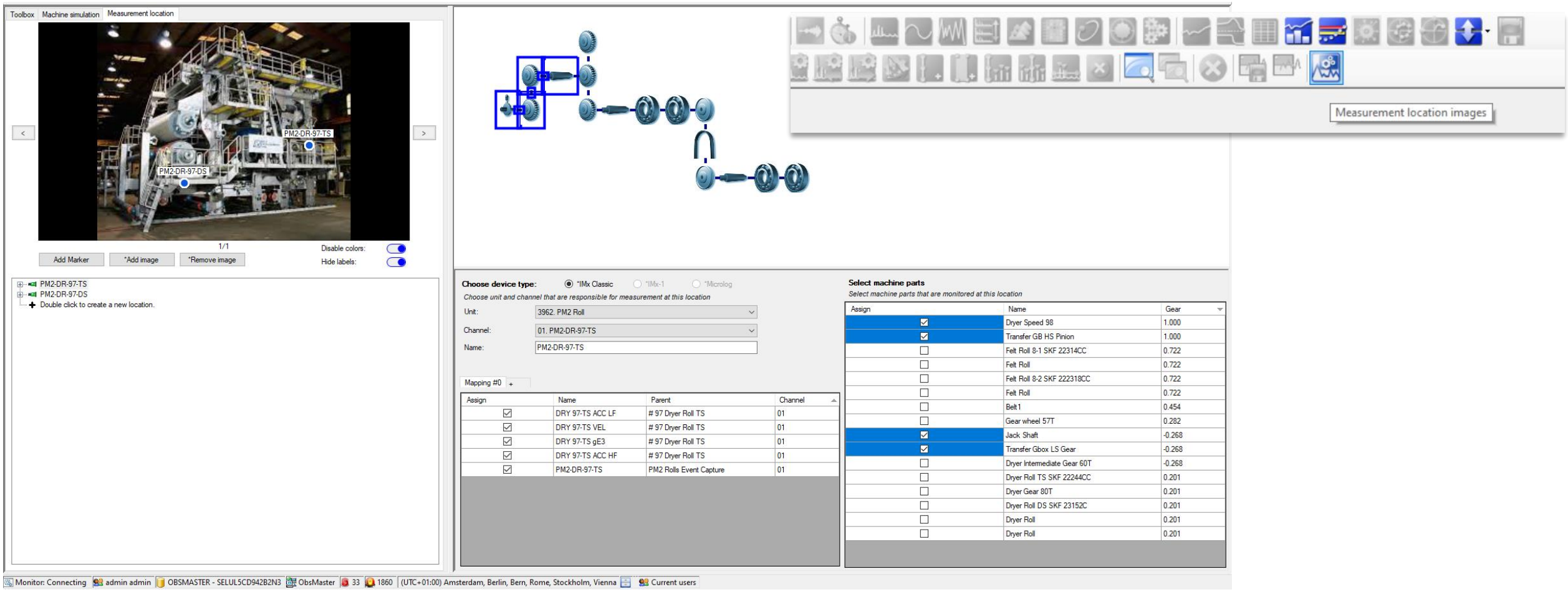
Feature obsolescence plan in future releases

- Message centre (disabling is planned for the next release)
- OPC UA Internal Build-in Server: Publishing Dynamic Data (disabling is planned for next release)

NEW FEATURES

Measurement location

- Measurement location is a new feature for a visual representation of vibration sensor position on the machine's photo/drawings.
- This functionality will help with the mapping of the machine's parts to device measurement channels/points.
- User can upload machine photos and set this mapping from the Machine parts view and later open this Measurement Location view from the toolbar during analysis.
- This feature can be used for mapping IMx Wired, IMx-1 and Microlog measurement points/channels.



Phoenix API: new Measurement Location endpoints

- Information about *Measurement Locations* as well can be extracted via the Phoenix API interface.
- We've introduced two new endpoints:
  - GET v1/measurement-locations – returns a list of all Measurement locations in DB
  - GET v1/machines/{machineId}/measurement-locations – returns all Measurement Locations for a particular machine.

GET /v1/measurement-locations — Returns a list of all Measurement Locations

Implementation Notes

Returns all Measurement Locations in the system

Response Class (Status 200)

OK

Model | Example Value

Inline Model [
MeasurementLocation
]
MeasurementLocation {
id (integer, *optional*): ID of the measurement location,
machineId (integer, *optional*): ID of the machine that this measurement location belongs to,
name (string, *optional*): User defined name of this measurement location,
pointPartMappings (Array[CM.Phoenix.Model.v1.MeasurementLocation.PointPartMapping], *optional*): List of Point Part mappings that belong to this measurement location
}
CM.Phoenix.Model.v1.MeasurementLocation.PointPartMapping {
pointIds (Array[integer], *optional*): IDs of the measurement points that are associated with this mapping,
partIds (Array[integer], *optional*): IDs of the machine parts that are associated with this mapping
}

GET /v1/machines/{machineId}/measurement-locations — Returns all Measurement Locations for a particular machine

Implementation Notes

Returns all Measurement Locations for the given machine

Response Class (Status 200)

OK

Model | Example Value

Inline Model [
MeasurementLocation
]
MeasurementLocation {
id (integer, *optional*): ID of the measurement location,
machineId (integer, *optional*): ID of the machine that this measurement location belongs to,
name (string, *optional*): User defined name of this measurement location,
pointPartMappings (Array[CM.Phoenix.Model.v1.MeasurementLocation.PointPartMapping], *optional*): List of Point Part mappings that belong to this measurement location
}
CM.Phoenix.Model.v1.MeasurementLocation.PointPartMapping {
pointIds (Array[integer], *optional*): IDs of the measurement points that are associated with this mapping,
partIds (Array[integer], *optional*): IDs of the machine parts that are associated with this mapping
}

Response Content Type application/json; v=1.0

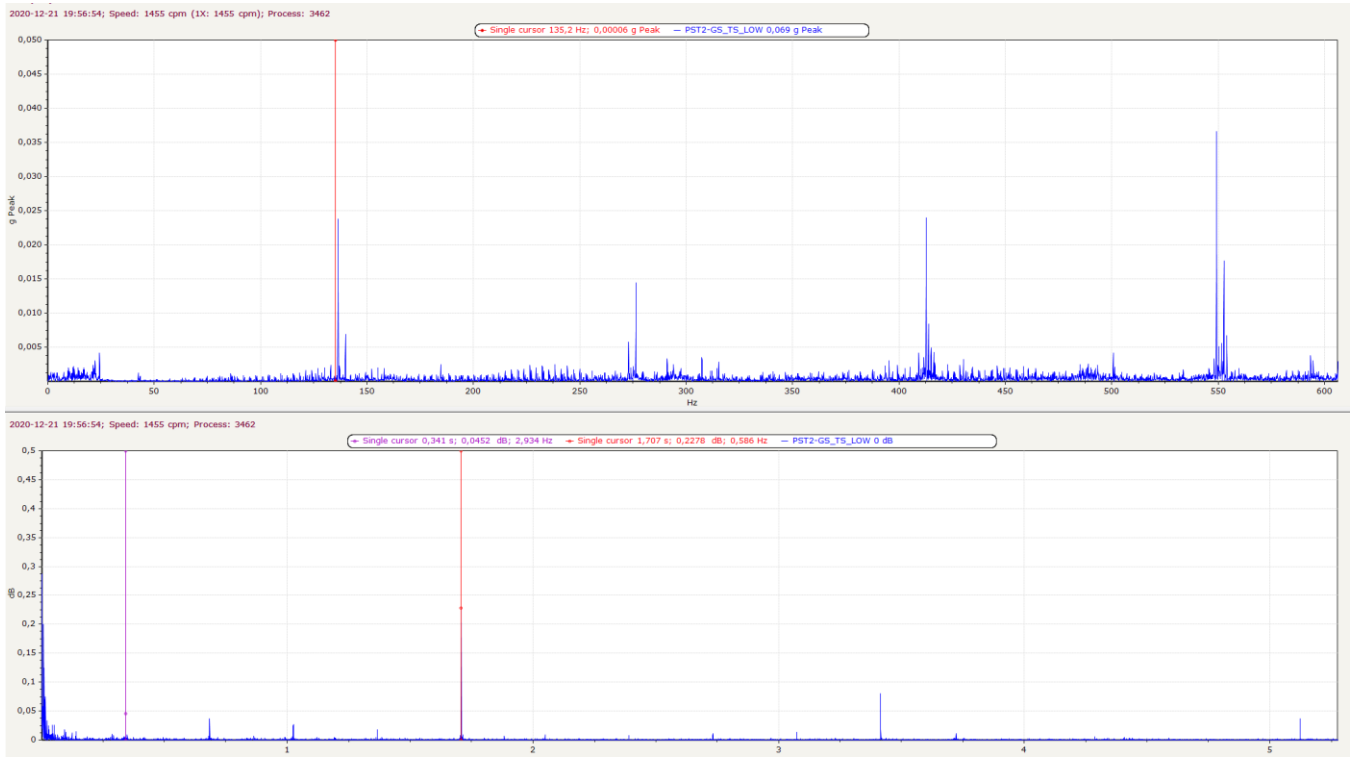
Parameters

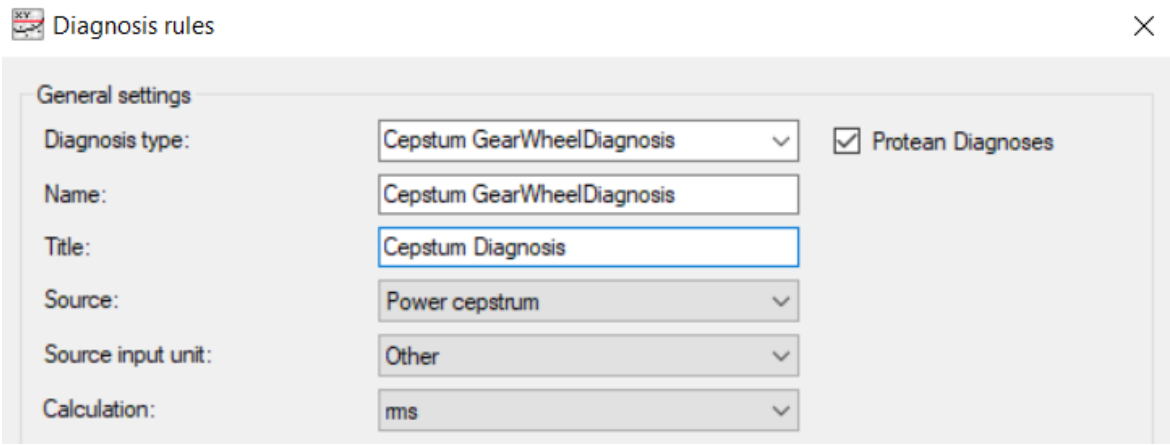
Parameter	Value	Description	Parameter Type	Data Type
machineId	(required)	Id of the machine	path	integer

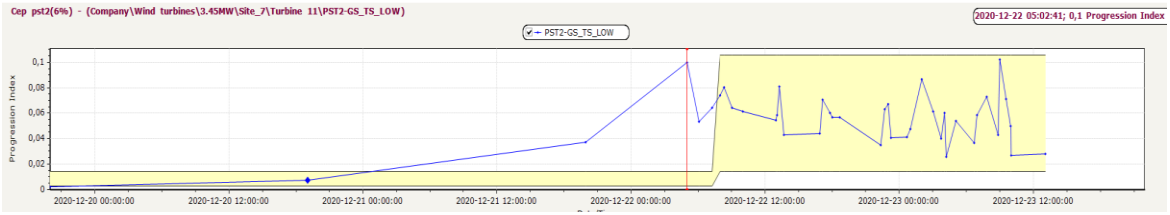
Power Cepstrum

- Power Cepstrum is a method for detection of periodicity in a frequency spectrum. This tool can be very useful for the analysis of machine failure vibration patterns which have multiple harmonics of fundamental frequency and/or sidebands, such as gearbox faults analysis.

- In this release, we added a new combined plot Spectra/Power Cepstrum manual frequency domain analysis and the ability to use Power Cepstrum data as a source for Diagnosis/Protean Diagnosis calculations.





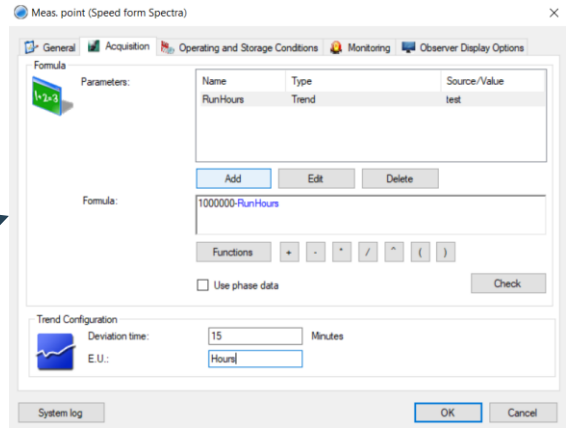


Allow Running hours points to be used as trend source for software derived point

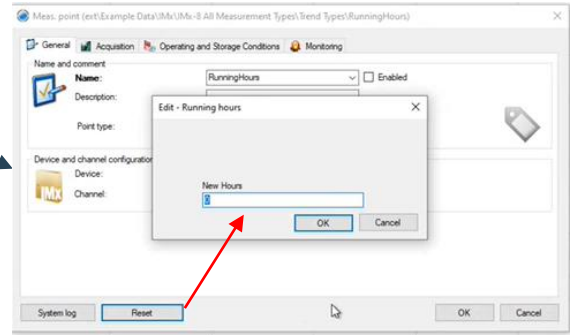
This feature was a customer request to help them calculate the remaining useful life KPI or the time to maintenance KPI.

- Added functionality that allows running hours to be used as a trend source in software-derived points.
- Running Hours point as well has a reset functionality for dropping the hours counter to a certain value.

Running hours as input for SW derived point



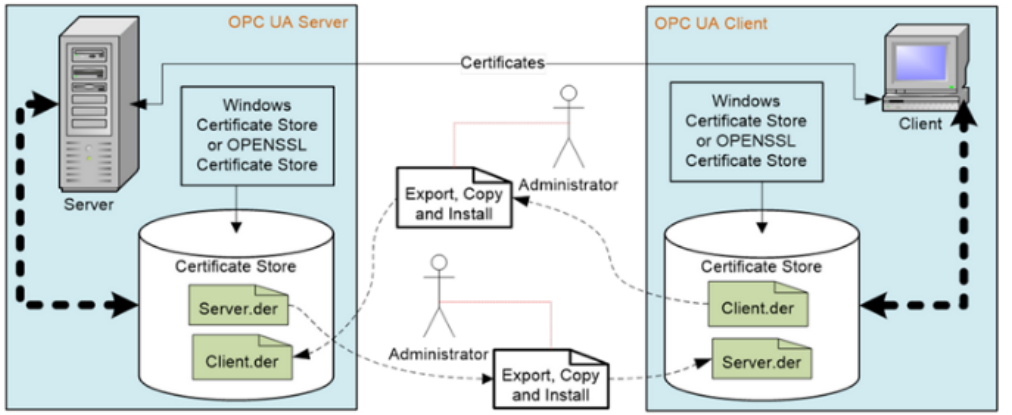
Reset function for Running Hours point

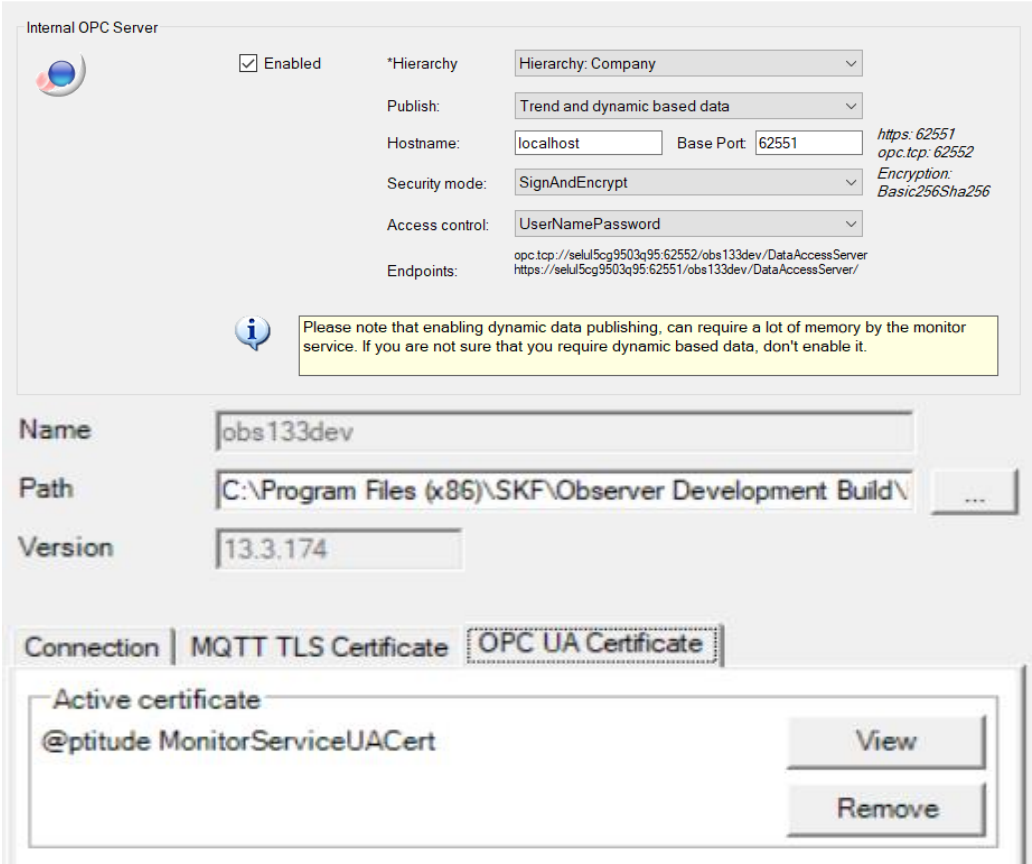


OPC UA – Security modes for Internal OPC UA build-in server

- Added OPC UA Security Modes and Access control for secure encrypted communication between Internal OPC UA server and external OPC UA Clients:
  - Now Observer's OPC UA Server supports: None/Sign/SignAndEncrypt security modes
  - Added supporting of None/UserNamePassword Access control modes
- Added selection of Hierarchy to be published in OPC UA server: Main Hierarchy or selected Workspace. This will help to reduce mount of data to be published and browsed by OPC UA clients for optimal performance and easier integration setup.
- OPC UA X.509 certificate management is arranged via the Monitor Service Manager application.

Note: For cloud installations, SignAndEncrypt security mode and UserNamePassword, access control mode only can be used.



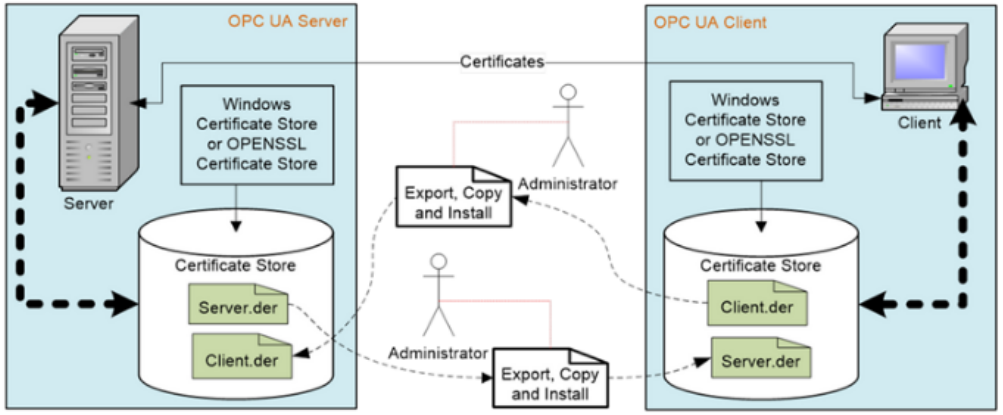


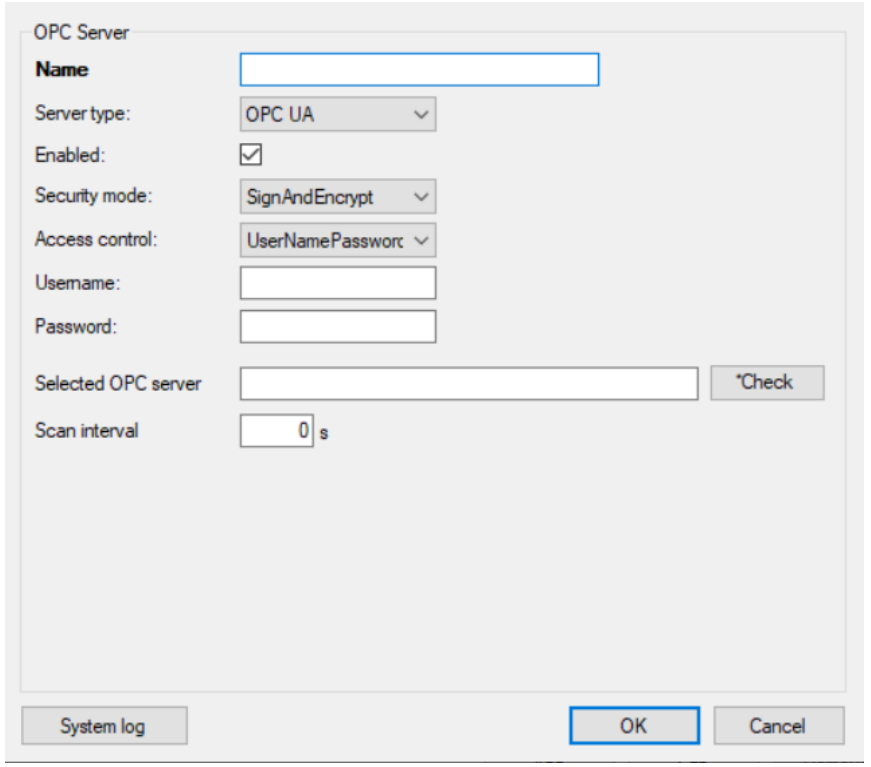
OPC UA – Security modes for Internal OPC UA build-in clients

- Added OPC UA Security Modes and Access control for secure encrypted communication between Internal OPC UA clients to external OPC UA Server

- OPC UA X.509 certificate management is arranged via the Monitor Service Manager application

Note: For cloud installations, SignAndEncrypt security mode and UserNamePassword access control mode only can be used.





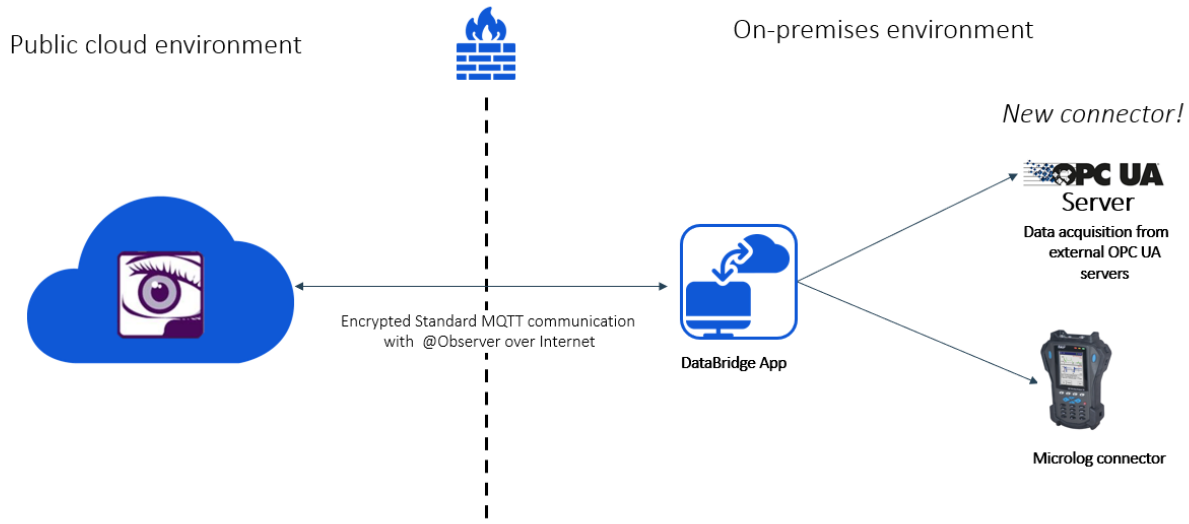
SKF @ptitude Observer Data Bridge: new OPC UA connector

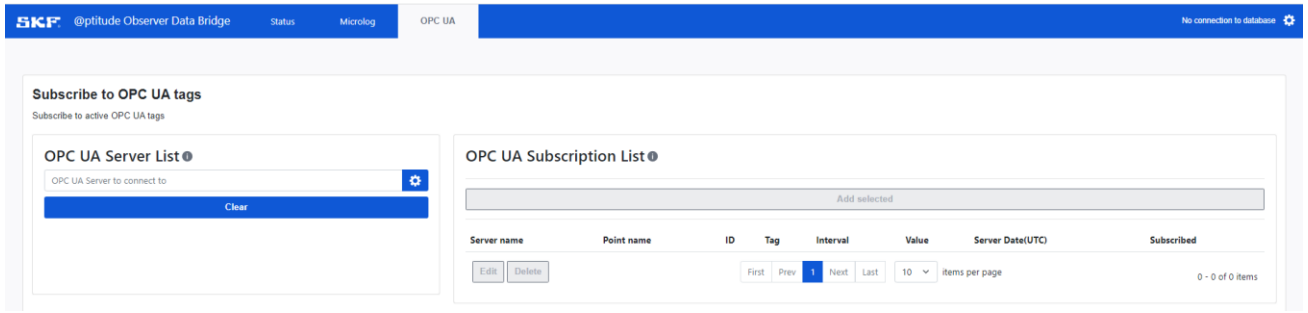
- Data Bridge was initially added to Observer to allow remote connection to Observer databases by Microlog devices but was always intended as a tool to facilitate communication between Observer and many other types of system.

- In this release of SKF @ptitude Observer Data Bridge, we've added an OPC UA client consumer which allows you to collect data from local OPC UA Servers and proxy this traffic via encrypted MQTT protocol to @ptitude Observer backend.

- This option will be useful if the internet connection with the remote @ptitude Observer backend is not stable because the Data Bridge app has a data buffering capability. Or this direct connection via the internet is not allowed, so in Data Bridge can be used as a proxy from the customer network.

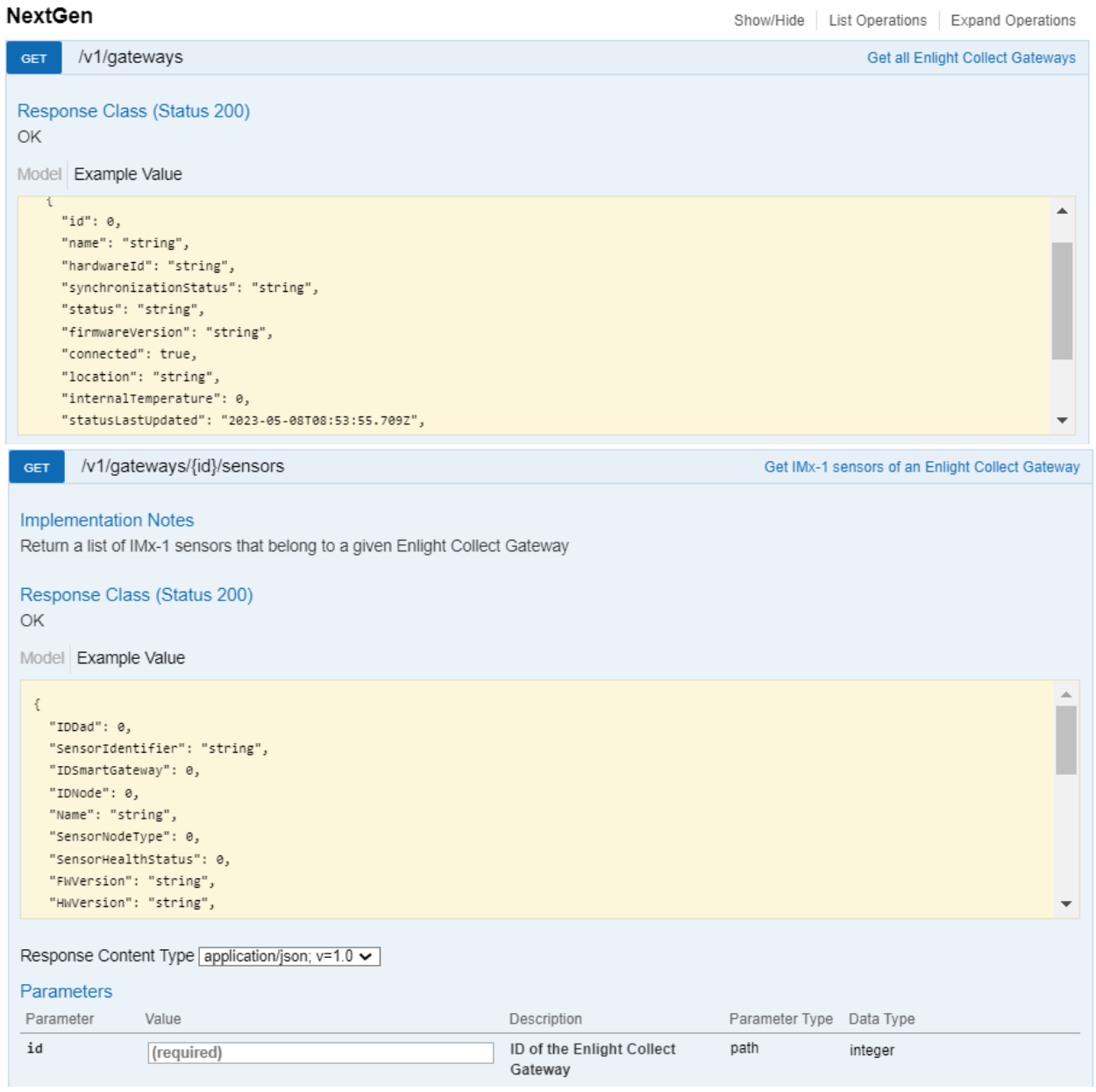
- In this release, we added a Linux version of this application with support for X and Y OS (this is a beta version of the app).





Phoenix API: new endpoint GET v1/gateways, GET v1/gateways/{id}/sensors

- Added new endpoints to help extract IMx-1 sensors information per IMx-1 Gateway



NextGen

Show/Hide | List Operations | Expand Operations

GET /v1/gateways — Get all Enlight Collect Gateways

Response Class (Status 200)

OK

Model | Example Value

```
{
  "id": 0,
  "name": "string",
  "hardwareId": "string",
  "synchronizationStatus": "string",
  "status": "string",
  "firmwareVersion": "string",
  "connected": true,
  "location": "string",
  "internalTemperature": 0,
  "statusLastUpdated": "2023-05-08T08:53:55.709Z",
```

GET /v1/gateways/{id}/sensors — Get IMx-1 sensors of an Enlight Collect Gateway

Implementation Notes

Return a list of IMx-1 sensors that belong to a given Enlight Collect Gateway

Response Class (Status 200)

OK

Model | Example Value

```
{
  "IDDad": 0,
  "SensorIdentifier": "string",
  "IDSmartGateway": 0,
  "IDNode": 0,
  "Name": "string",
  "SensorNodeType": 0,
  "SensorHealthStatus": 0,
  "FWVersion": "string",
  "HWVersion": "string",
```

Response Content Type application/json; v=1.0

Parameters

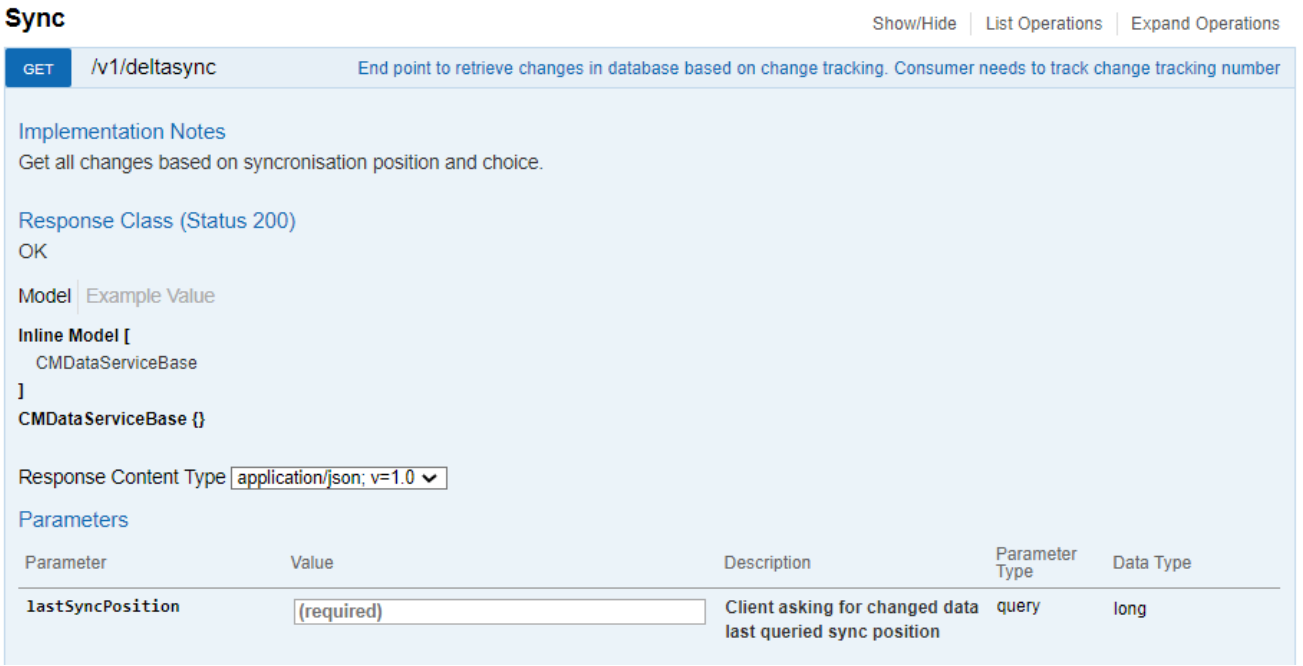
Parameter	Value	Description	Parameter Type	Data Type
id	(required)	ID of the Enlight Collect Gateway	path	integer

Phoenix API: new endpoint GET v1/deltasync

- A new endpoint was added to retrieve changes in DB based on change tracking (lastSyncPosition parameter).
- The data coming back from this new endpoint is exactly the same as the following endpoints (that exist in the documentation) and have the exact same configuration:
  - */v1/points/{pointid/diagnosesMeasurment*
  - */v1/points/{pointid/trendMeasurement*
  - */v1/points/{pointid/dynamicMeasurement*
  - */v1/points/{pointid}*

Example how to use the SyncPosition :

a. Ask *deltasync* , lastSyncposition =1, maxNumberOfRecords 100, typesToInclude = 2 (Trend)
b. The API will reply with a maximum of 100 records.
c. The last record *SyncPosition* is for example 4510. (the last record always has the highest *SyncPosition*)
d. The next query uses 4510 to get more new records if the number of records was 100 (if the number of records was less than 100 no need to ask for more)
e. Please note that the *SyncPosition* is per database not table so the *SyncPoistion* value can increase more than 100 in the example above.



Note:

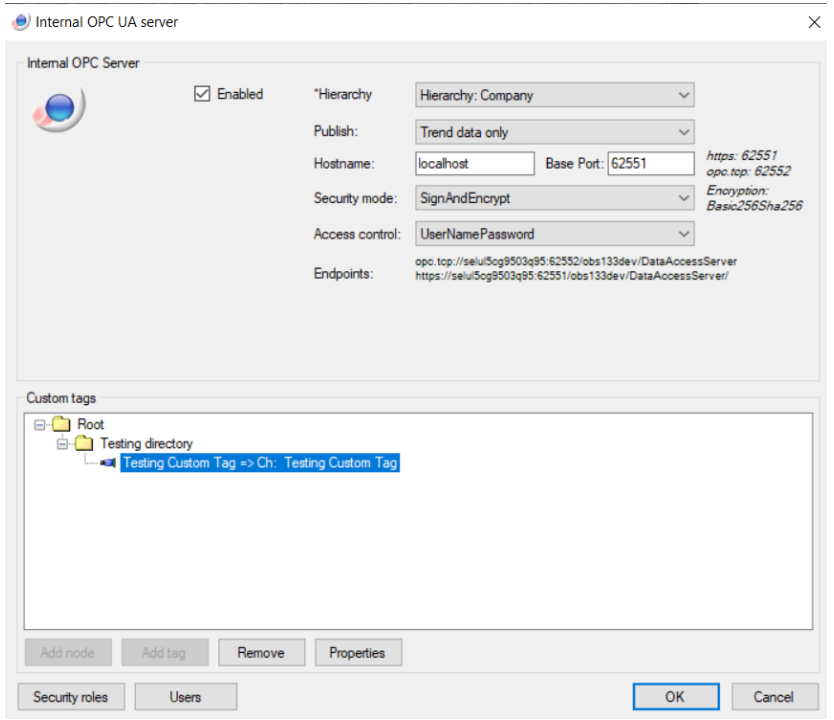
- The retention time for the SQL sync is 7 days recommended to sync once per day.
- The Sync starts the first time anybody accesses the endpoint DeltaSync.
- Recommendation to sync each type by itself, Dynamic, Trend, Nodes, Diagnoses

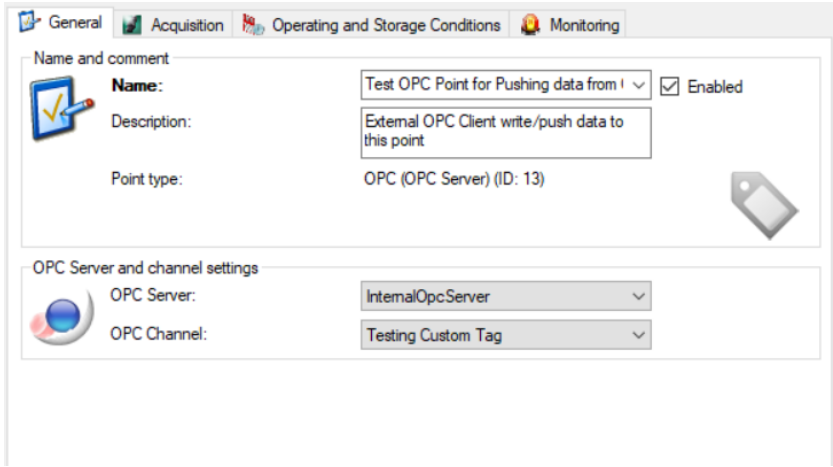
ENHANCEMENTS

OPC UA server’s Custom tags as a source for OPC points

- In previous Observer releases, OPC Points could use as a data source only OPC tags which were subscribed by Internal OPC UA Client. In this release, users can use Internal OPC UA server Custom Tags as a source for SW OPC Points as well. This should simplify the setup of scenarios when an external OPC UA client should push/write data to Observer's internal OPC UA server.

- Use the InternalOpcServer option for the OPC Server attribute in the OPC measurement point.

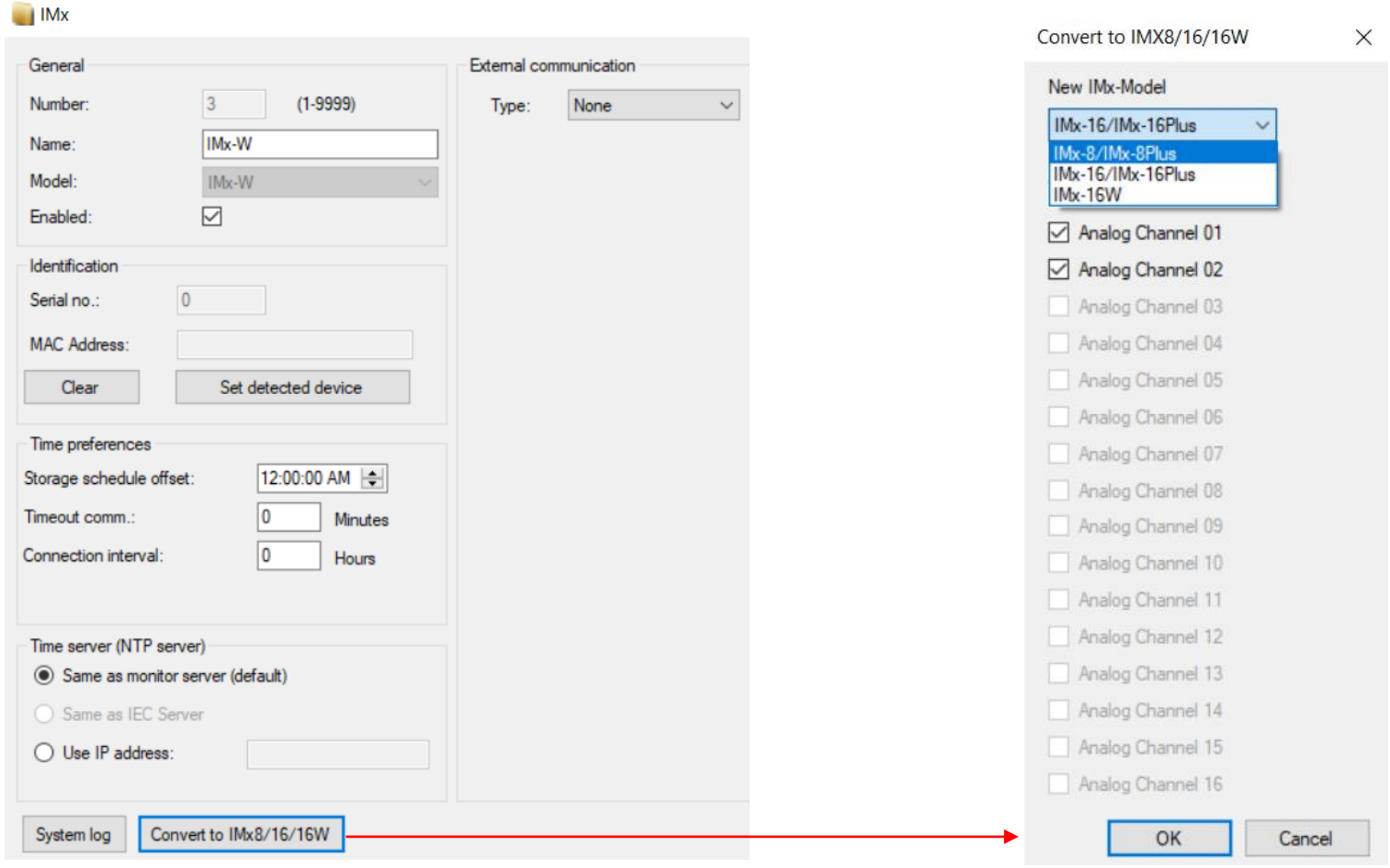




IMx-W/ IMx-S to IMx-8/IMx-16/IMx-16W configuration conversion

- Feature for conversion of old IMx device types configurations for making user's transition to latest IMx devices version smoother.

Note: Once you've done the conversion, there is no functionality to revert it.



Machine speed: Constant machine speed and SW speed handling

- To unify the way of handling configuration of machines with constant speed and making it a bit faster, we added an extra option in ***Machine Properties – Machine Speed tab*** called - ***Constant Speed***. Now the user can enable it, and system automaticaly applies the specified input speed value for each stored vibration measurement.

- New setup for ***Constant Speed*** machines includes two steps:

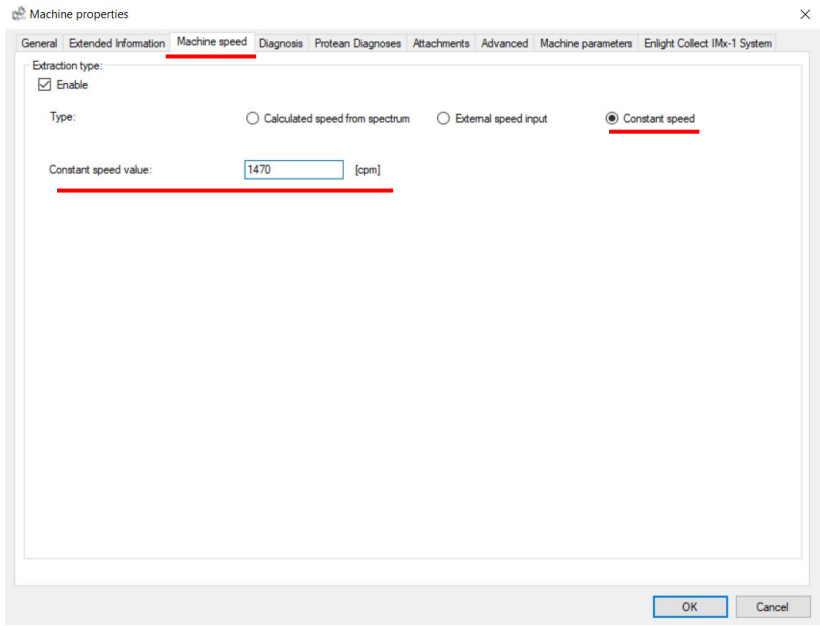
  - In machine parts for ***Speed input location*** component, use default option – ***Associated Speed used***.
  - Enable in *Machine Properties – Machine speed* option Constant Speed and define speed value to be used in (CPM)

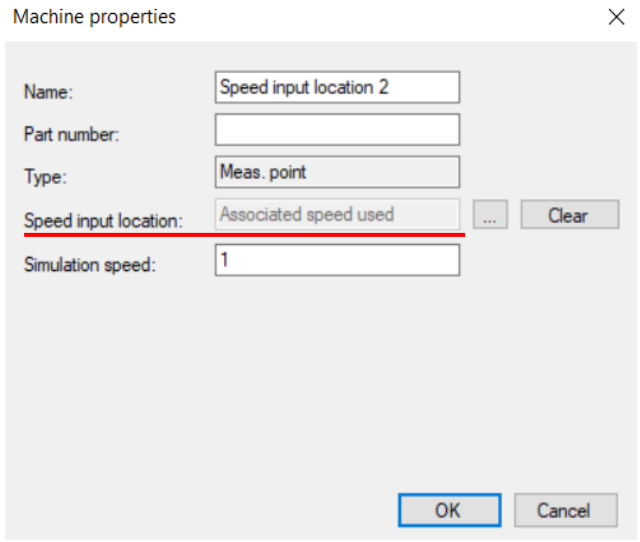
- Constant speed option works with all main supported devices: *IMx-1, Microlog,* and IMx wired device family for both Trend and Dynamic measurements.

- With these changes, it's not mandatory to use associated SW speed points reference for each measurement point in constant speed machine configuration, but this old way of setup will remain for better user transition.

- **Note**: In case some measurement point requires another Associated Speed, the reference user can edit it within Point properties, so after that point's measurement will be excluded from ***Constant speed*** postprocessing and follow to new associated speed reference instead.

- **Note**: Only the machine's drivelines where *the* ***Associated speed*** *used* option is enabled for ***Speed input*** *location* will be calculated by this new feature. If some machine's drivelines have another reference, all related fault frequencies will be calculated by it.





Access to Trend/Diagnosis/Protean plots from full screen mode of Process overview

- In this version, users with *Machine Operator 1* role can choose between different plots in *the Process overview*, including Trend/Diagnosis/ Protean diagnosis plot. This feature should help operators get access to the measurement's history to check trends behavior.

Phoenix API: v1/captures/{captureId} added speed Pulses, associated Process/Digital parameters

- SpeedPulses array, associated Process/Digital measurement for each event capture chunk now included in the API response.

- Issue with incorrect timestamps has been resolved, so now the parameter ***begin*** = start of the first chunk, and parameter ***end*** = end of the last chunk in Event Capture.

For more information and examples, please check Phoenix API specification for SKF @ptitude Observer v13.3

GET /v1/captures/{captureId} Retreives one capture

Implementation Notes

This call returns a list of measurement information for a specific capture

Response Class (Status 200)

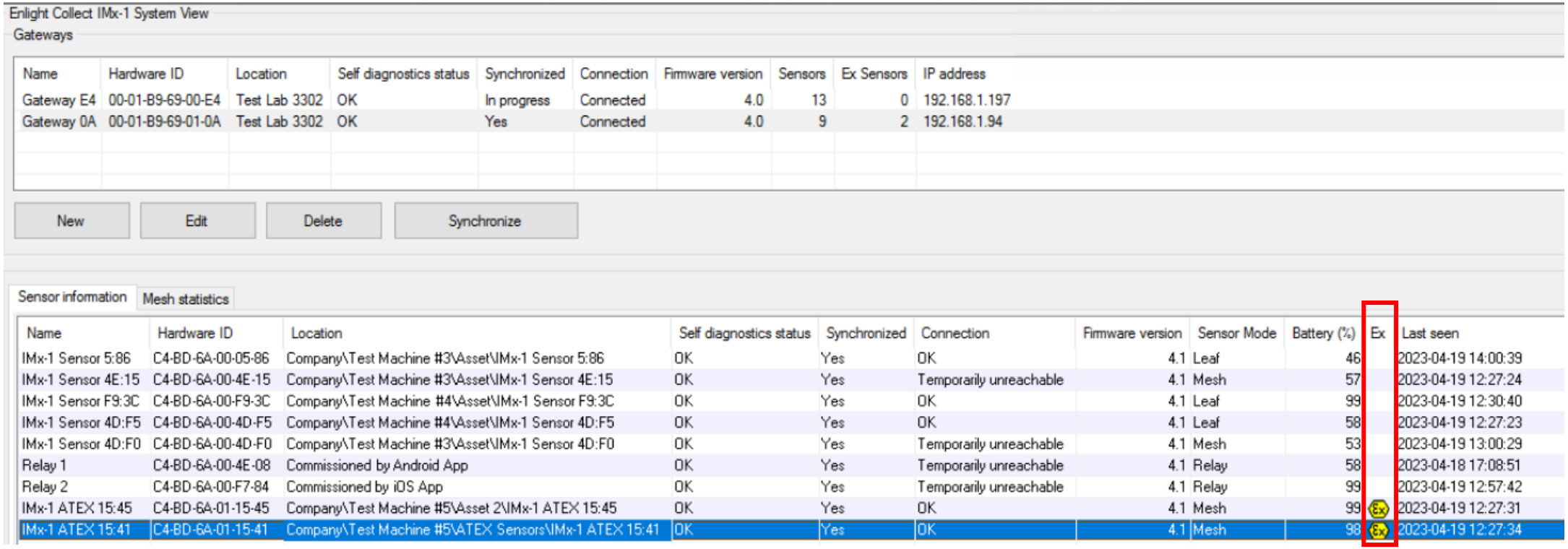
OK

Model | Example Value

Inline Model [
  Measurement
]
Measurement {
  euType (integer, *optional*): Internal point engineering units, see EUType,
  timewave (Array[number], *optional*): List of floats with timewave measurment data,
  speedPulses (Array[number], *optional*): List of integer with speed pulse data,
  speedMeasurements (Array[number], *optional*): List of floats with speed measurement data,
  processMeasurements (Array[number], *optional*): List of floats with process measurement data,
  digitalMeasurements (Array[integer], *optional*): List of integers with digital measurement data,
  scaleFactor (number, *optional*): Timewave measurement scaling factor,
  begin (string, *optional*): Date when measurement was started (UTC),
  triggered (string, *optional*): Date when measurement was triggered (UTC),
  end (string, *optional*): Date when measurement was ended (UTC),
  pointID (integer, *optional*): Database ID of the point,
  name (string, *optional*): Name of the capture point,
  comment (string, *optional*): Comments about the capture point,
  id (integer, *optional*): Datbase ID of the capture point,
  src (string, *optional*): API source string for retreiving measurement point information,
  triggerdBy (string, *optional*): Reason why the capture has triggered,
  lines (integer, *optional*): The configured number of lines for the capture group,
  samplingFrequency (number, *optional*): The configured sampling frequency for the capture group
}

IMx-1 ATEX version sensors

- In this Observer version, the user will be able to identify a newly introduced IMx-1 -EX version sensors (CMWA 6100-EX) within IMx-1 System View, in order to understand future battery-life trends.
- Recommended FW version for CMWA 6100-EX sensor v 4.1



BUGS

*Not all bugs are listed in presentation, list you can find in change log on

IMx-1 : Hierarchy icons not updated in workspace and process overview

- Fixed an issue with IMx-1 points not having their status updated correctly when visualised in a workspace or the process overview.

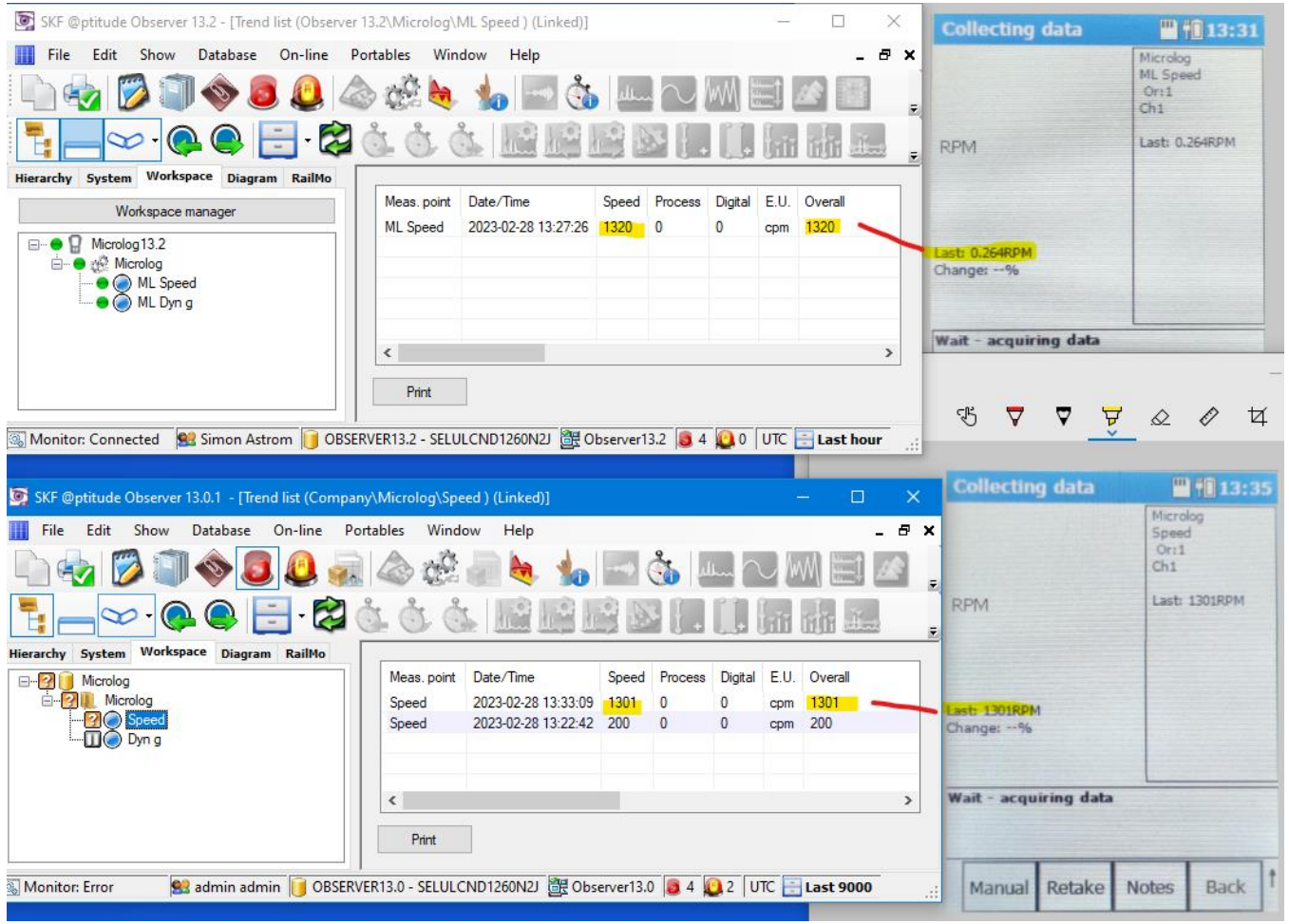
- The process view colour will now follow the hierarchy tree colour.

- IMx-1 points will be updated in a 20-30 seconds interval when shown in the process overview.

Microlog : “last value” not correct

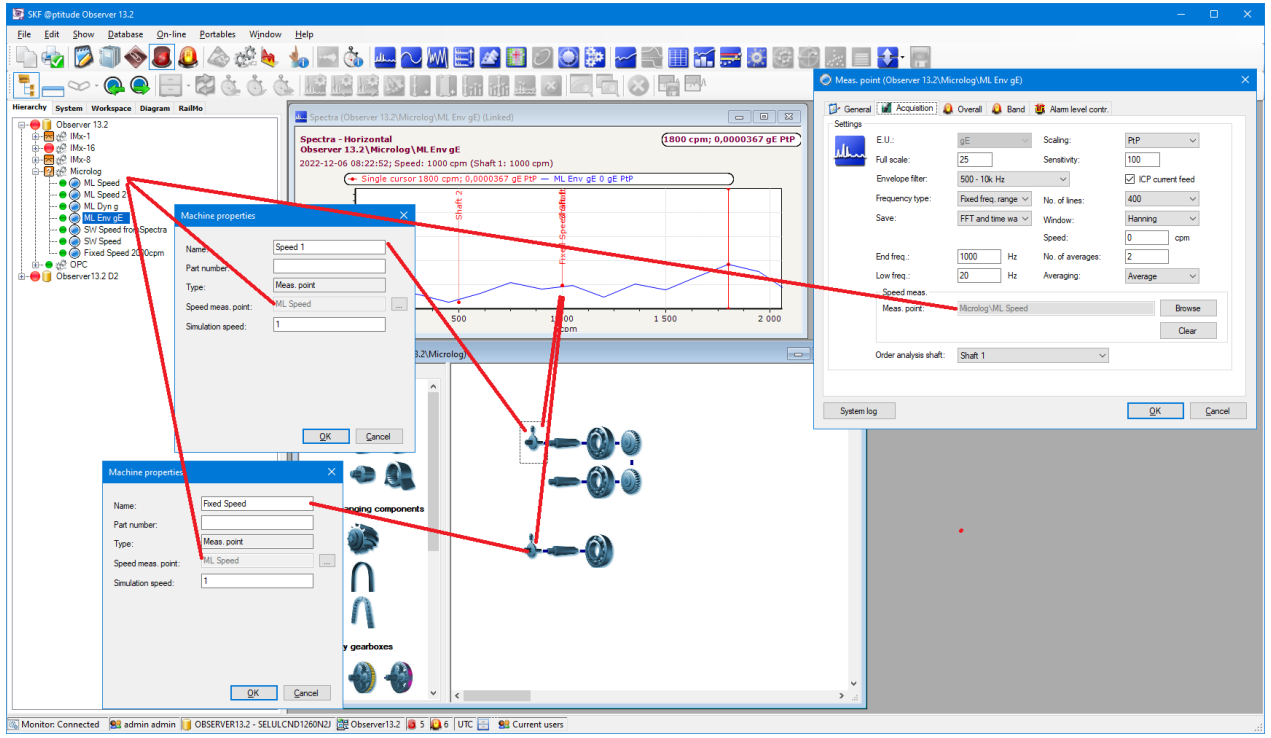
- Fixed an issue that would cause Microlog not to download the correct last value on points.

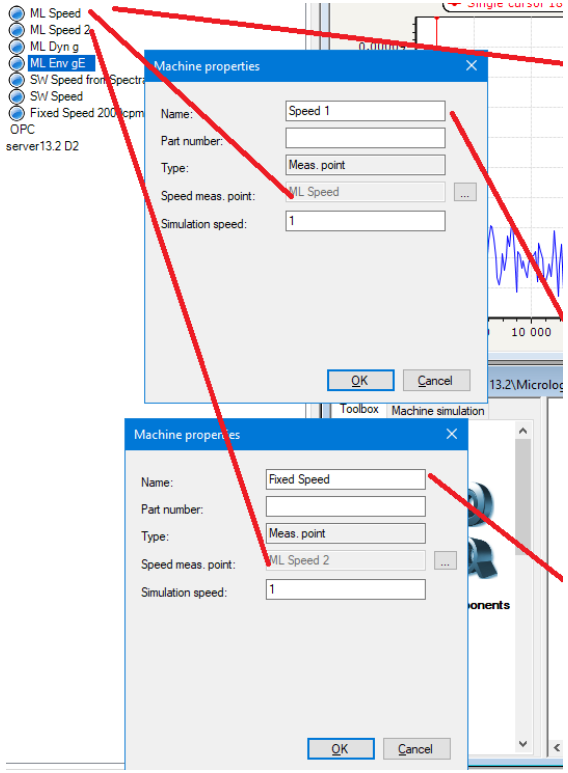
- This issue is now fixed and last value will be displayed correctly (as it did before 13.2).



Microlog: Not possible to have different speed source for two chains

- Fixed a bug where having two machine chains using different speed inputs would break the fault frequency calculations





- The left one worked, while the right one would break the fault frequency calculation.

Microlog: Default speed

- Fixed an issue with Microlog spectra's where using 'set speed' or 'default speed' in combination with 'constant speed' or any associated machine part speed would incorrectly calculate fault frequencies.

- The issue would occur if the machine part input location was set to Microlog tacho and Acceleration point was using the default speed or set speed function. And would cause the diagnosis and plot not to show fault frequency.

Data Bridge route download

- If a user downloaded routes through Data bridge to a Microlog device with existing routes, the data would be overwritten. This posed a risk that data would be unintentionally lost.

- Now, if measurement data exits on the route in the Microlog, a warning is added when trying to update a route to Microlog. This prevents data to be overwritten.

Data bridge: Broken hierarchy on Microlog

- The issue was that when downloading routes to Microlog, the hierarchy on the Microlog was not displayed correctly, and points would be missing from the hierarchy. Editing end freq (from 20-50 Hz, for example) would cause the point to be transferred correctly.

- But this issue is now fixed and now editing should be needed.

Data bridge incorrect timestamp

- Fixed an issue that would cause a mismatch in time zones between Data Bridge and Microlog. This would happen when getting measurement data from Microlog because the UTC time needed by Observer was not converted properly.

MPUW:

- Fixed a crash that would occur when using Multiple Point Update Wizard to edit IMx Process points linked to a Modbus register.
- Fixed a crash in Multiple Point Update Wizard when trying to edit Gear Inspector or Time Waveform Analysis points.
- Fixed a crash in Multiple Point Update Wizard when trying to edit Speed points.
- Fixed an issue which enabled setting incorrect configurations on IMx-1 points through multiple point update wizard (In the report, the user was able to set fmax up to 6400 for IMx-1).

External database cause Observer to enter “not responding” state

- Fixed an issue where Observer would freeze in case any external databases were unreachable.

- For a full list of fixed bugs, see the change log.

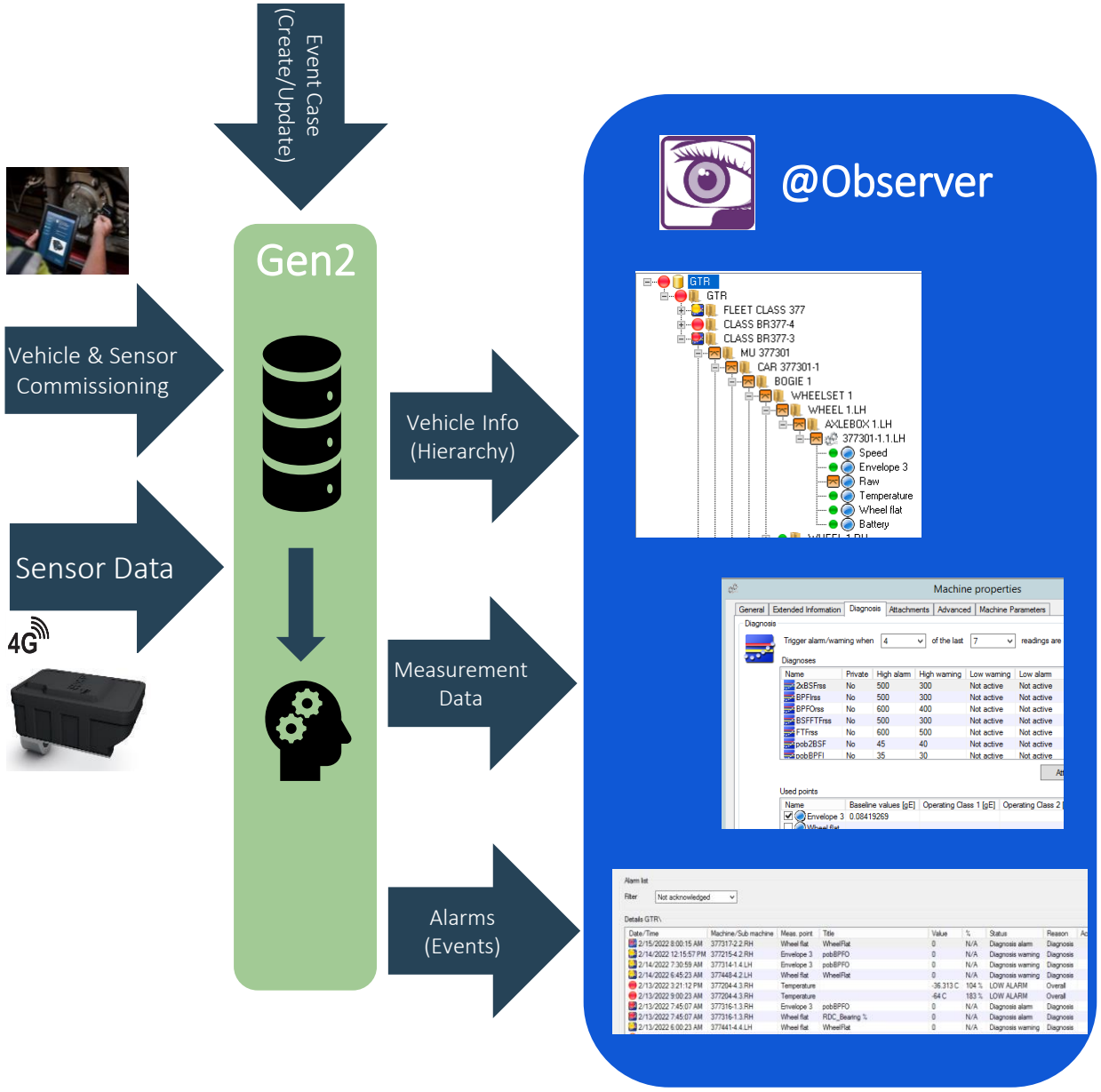
Rail Track Monitoring release summary

NEW FEATURES

Observer

Integration of Observer with Rail GEN2 backend:

- User should be able to select which Rail Gen2 companies they want to see in the Observer hierarchy.
- All vehicles/assets automatically visible in the Observer hierarchy, for selected company(s).
  - Vehicle changes made in Gen2 (naming etc.), automatically propagate to the Observer hierarchy
- Sensor measurement data automatically shown in Observer
  - Battery Voltage
  - Speed
  - Bearing (Env3)
  - Wheel Flat
  - Raw (Acceleration)
  - Temperature
- All sensor measurement data should be visible in Observer, even if not received in time-sequenced order.
- IRAD/IRSA Alarms in Observer.
- Create Alarms on the basis of event cases.



APIs

3 new APIs are added for implementing the features of Televic MVP:

- /v1/insight/vehicle - Create and update vehicles and their information.
- /v1/insight/telemetry – Update sensor measurement data.
- /v1/insight/getNodeIDAndNodeName – Returns ID and names of measurement points under machines with the help of which alarms are created using Alarm API.

Prerequisites:

Before any vehicle can be imported into @Observer, a template for that vehicle must first be created and appended to the template library.

Vehicle API

Create a vehicle/asset or Update vehicle/asset details received from Rail Gen2 companies in Observer.

POST /v1/insight/vehicle

Recieves Vehicle commission CREATE or UPDATE from Insight Rail Cloud -- Example URL for call: http://localhost:50000/api/Insight/vehicle

Response Class (Status 200)

OK

Model | Example Value

```
{
  "message": "string"
}
```

Response Content Type: application/json; v=1.0

Parameters

Parameter	Value	Description	Parameter Type	Data Type
commission	{ "reference": "bdc2d831-fc62-47a4-87d1-e75b7c590h87", "setId": "TRAIN TestDemoNew", "oem": "KVB", "designation": "TRAIN TestDemoNew", "description": "Gen1 imported	**RailCloud Commission object**	body	Model | Example Value

Parameter content type: application/json; v=1.0

```
{
  "SetUIC": "string",
  "SetId": "string",
  "OEM": "string",
  "Designation": "string",
  "Description": "string",
  "Classification": "string",
  "Traction": "string",
  "Coupling": "string",
  "CurrentStatus": "string",
  "Carriages": [
```

Response Messages

HTTP Status Code	Reason	Response Model	Headers
204	The request was successfully completed but no content found.		
400	The request was invalid	Model | Example Value	

POST /v1/insight/commissionVehicle

POST /v1/insight/updateVehicle

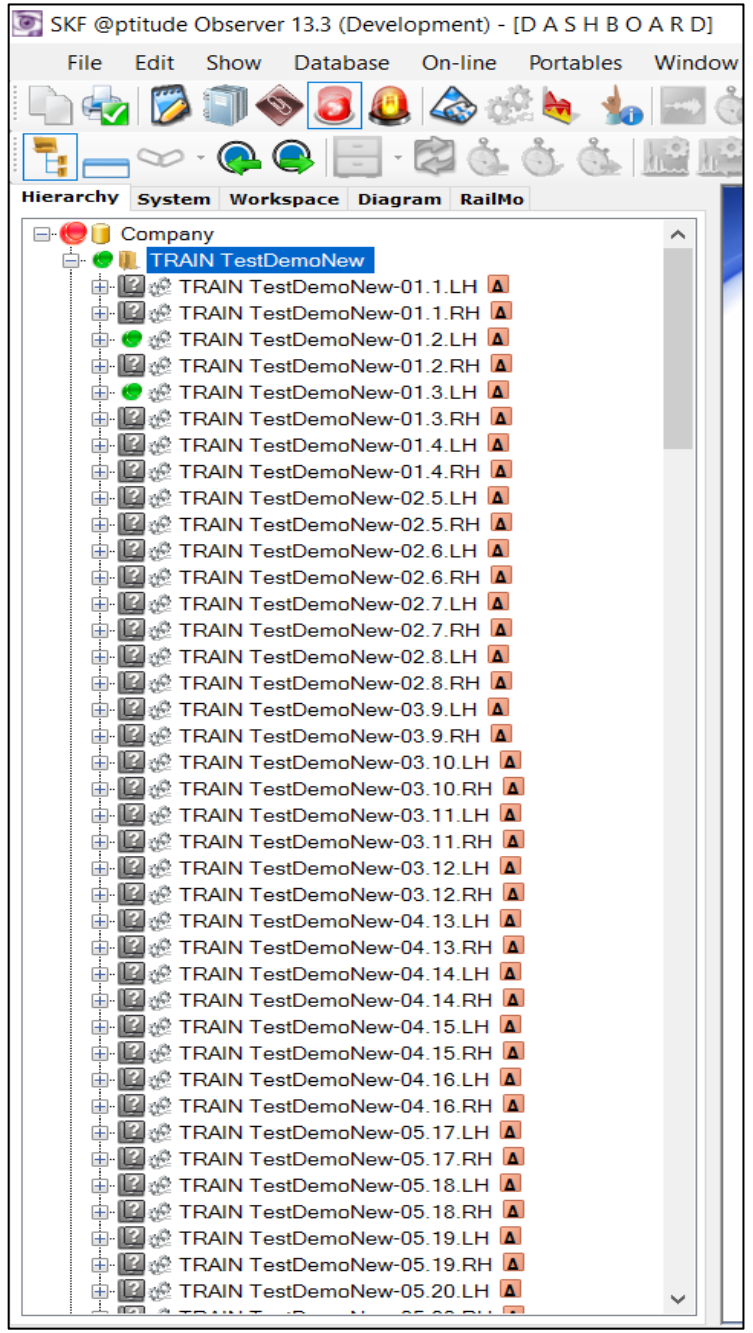
POST /v1/insight/vehicle

JSON


```
{
  "reference": "bdc2d831-fc62-47a4-87d1-e75b7c590h87",
  "setId": "TRAIN TestDemoNew",
  "oem": "KVB",
  "designation": "TRAIN TestDemoNew",
  "description": "Gen1 imported vehicle: 43663",
  "classification": "MultipleUnit",
  "coupling": "Coupled",
  "currentStatus": "InService",
  "carriages": [
    {
      "evn": "D99911",
      "position": 1,
      "vehicleNumber": "TRAIN TestDemoNew-01",
      "wheelSets": [
        "6bc5bd43-fe33-4f7f-a8bd-40ea4abd382f",
        "b5e02100-d858-40db-b41d-8c3de5d52d51",
        "041413d6-c826-4a6d-89ad-180ae540fa7b",
        "16fda0fb-bd0a-46bf-ba15-df2ec7972655"
      ],
      "reference": "c9f4fd18-3ac9-433c-9f11-f7080af62a22"
    },
    {
      "evn": "D84805",
      "position": 2,
      "vehicleNumber": "TRAIN TestDemoNew-02",
      "wheelSets": [
        "4529fb30-88ba-4180-8da3-067f61a3f02f",
        "8e26469f-15f7-4306-9f4e-4586c5fa9cd5",
        "187d5801-ff01-427d-afee-edf4b354c604",
        "e0508413-52a7-4eaf-aaf1-54861c191e1b"
      ],
      "reference": "c01cdab5-1f7f-4c0a-afdb-d18f81d40277"
    },
    {
      "evn": "N5380",
      "position": 3,
      "vehicleNumber": "TRAIN TestDemoNew-03",
```


file length : 117,303 lines : 3,724

Observer



Post Telemetry API

Update Sensor measurement data in Observer.

POST /v1/insight/telemetry — Saves a new dynamic measurement into the database

Implementation Notes

Updates the measurements of the speed, temperature, raw data, envelope, wheelflat and battery points

Response Class (Status 200)

OK

Model | Example Value

```
{}
```

Response Content Type: application/json; v=1.0

Parameters

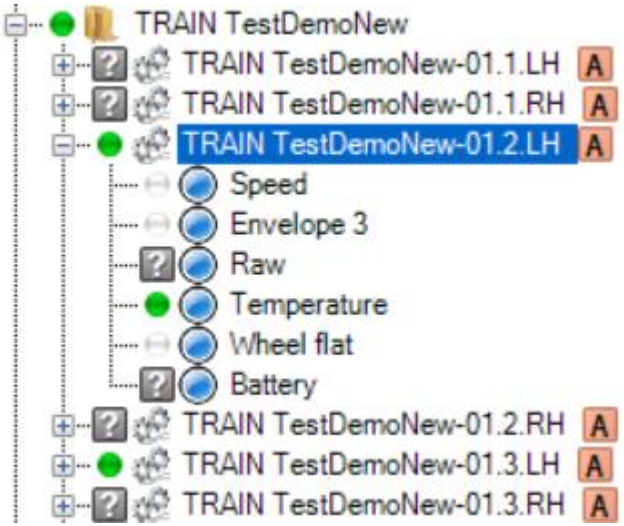
Parameter	Value	Description	Parameter Type	Data Type
measurement	{ "tenantid": "volatiletesttenant", "vehicle": "bdc2d831-fc62-47a4-87d1-e75b7c590h76", "wheelset": "187d5801-ff01-427d-afee-edf4b354c604", "axlebox": "d1cbf86c-2bca-4eed-92ba-	Data Service Measurement object	body	Model | Example Value

Parameter content type: application/json; v=1.0

```
{
  "values": [
    {
      "timestamp": 0,
      "identifier": "string",
      "energy_remaining": {
        "units": "string",
        "value": 0
      },
      "v_0": {
        "units": "string",
```

Response Messages

HTTP Status Code	Reason	Response Model	Headers
204	The request was successfully completed but no content found.		



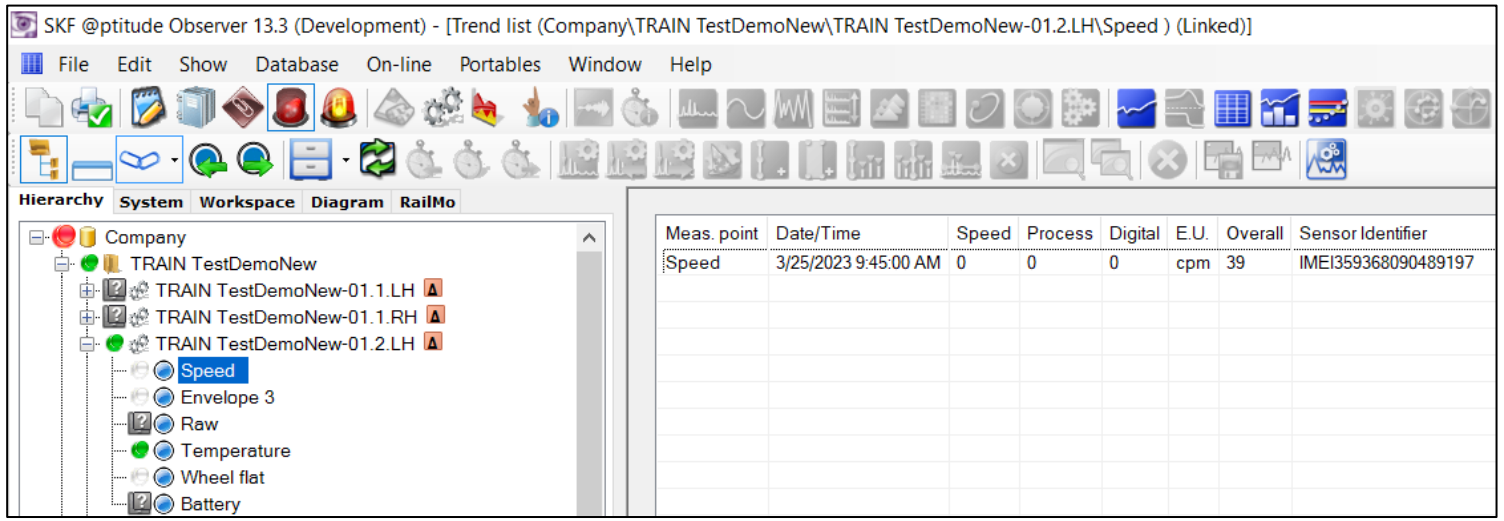
JSON


```
{
  "tenantid": "volatiletesttenant2",
  "vehicle": "bdc2d831-fc62-47a4-87d1-e75b7c590h87",
  "wheelset": "041413d6-c826-4a6d-89ad-180ae540fa7b",
  "axlebox": "66cb16dd-9427-4bdc-a5b2-961feb73c99c",
  "values": [

        {
         "timestamp":1679737500000,
         "gnss_speed_to_rotation_2":{
            "Value":"39",
            "units":"ms"
         }
      }

  ],
  "identifier": "IMEI359368090489197"
}
```


Observer



Generate alarms in Observer using the alarm API – GetNodeIDAndNodeName

To generate alarms in Observer, we use v2/alarms under alarms API. This endpoint accepts pointID as an input parameter, but other systems will not know the exact PointID of the measurement point because the IDs are internal to Observer.

Hence, we use this new API to accept Vehicle ID, WheelsetID and Axlebox ID and return the pointID and pointName corresponding to it so then this returned ID can be used to create alarms in Observer.

GET /v1/insight/getNodeIDAndNodeName — Returns NodeId and NodeName corresponding to the VehicleID, WheelsetID and AxleboxID passed.

Response Class (Status 200)

OK

Model | Example Value

```
{}
```

Response Content Type: application/json; v=1.0

Parameters

Parameter	Value	Description	Parameter Type	Data Type
vehicleID	bdc2d831-fc62-47a4-87d1-e75b7c590h87		query	string
wheelsetID	6bc5bd43-fe33-4f7f-a8bd-40ea4abd382f		query	string
axleboxID	06d68ac9-48cf-42ee-b973-70b811b6f9d7		query	string

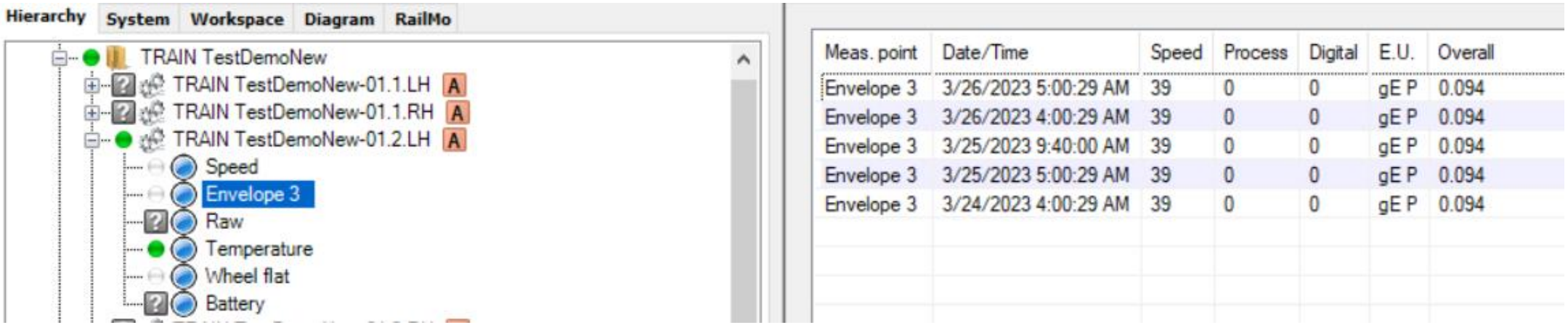
Response Messages

HTTP Status Code	Reason	Response Model	Headers
204	The request was successfully completed but no content found.		
400	The request was invalid	Model | Example Value	

```
{
  "message": "string",
  "messageDetail": "string"
}
```

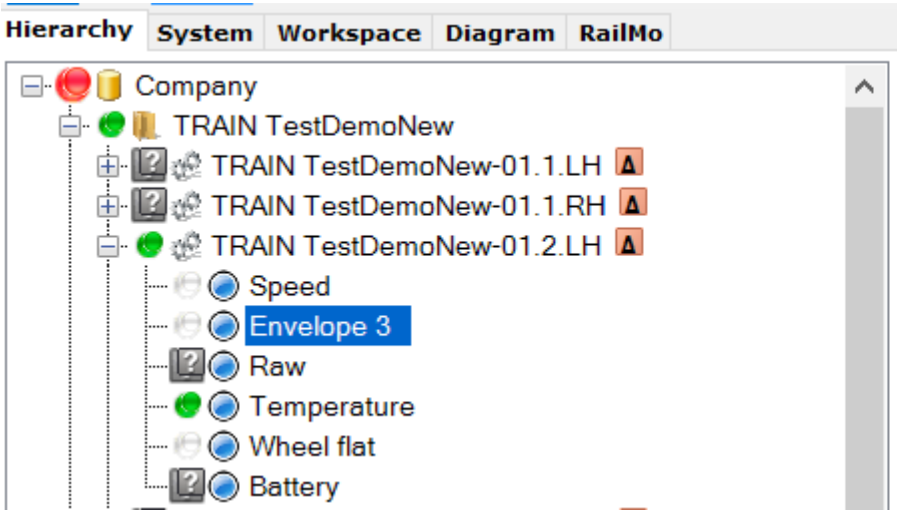
Check for Insight License key for adding Sensor Identifier in TrendList and Meas. Date page.

Observer TrendList without Insight Rail license



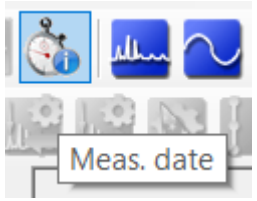
Meas. point	Date/Time	Speed	Process	Digital	E.U.	Overall
Envelope 3	3/26/2023 5:00:29 AM	39	0	0	gE P	0.094
Envelope 3	3/26/2023 4:00:29 AM	39	0	0	gE P	0.094
Envelope 3	3/25/2023 9:40:00 AM	39	0	0	gE P	0.094
Envelope 3	3/25/2023 5:00:29 AM	39	0	0	gE P	0.094
Envelope 3	3/24/2023 4:00:29 AM	39	0	0	gE P	0.094

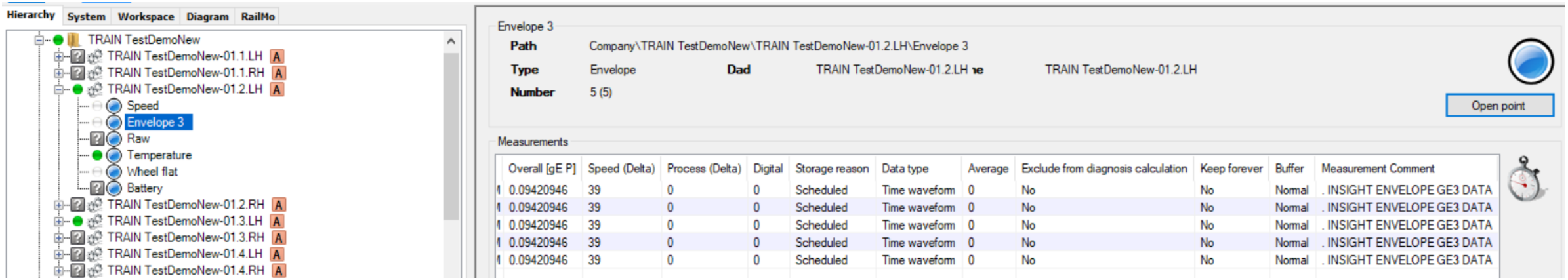
Observer TrendList with Insight Rail license – SensorIdentifier column is added to the TrendList.



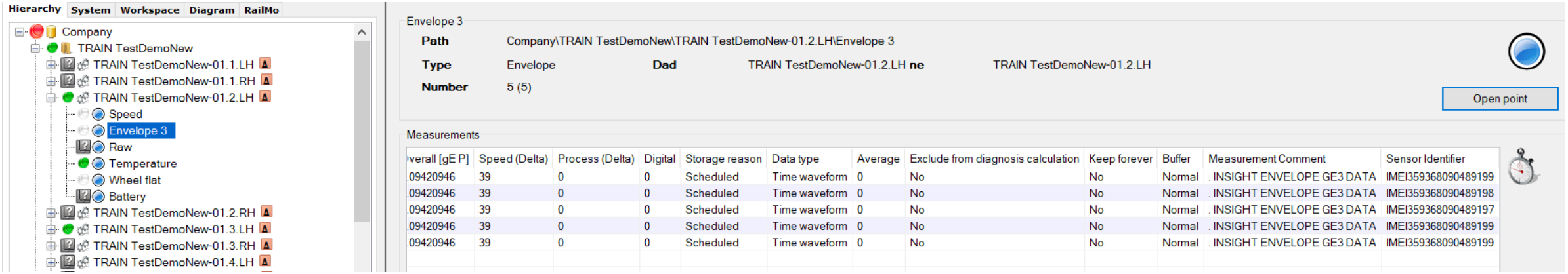
Meas. point	Date/Time	Speed	Process	Digital	E.U.	Overall	Sensor Identifier
Envelope 3	3/26/2023 5:00:29 AM	39	0	0	gE P	0.094	IMEI359368090489199
Envelope 3	3/26/2023 4:00:29 AM	39	0	0	gE P	0.094	IMEI359368090489198
Envelope 3	3/25/2023 9:40:00 AM	39	0	0	gE P	0.094	IMEI359368090489197
Envelope 3	3/25/2023 5:00:29 AM	39	0	0	gE P	0.094	IMEI359368090489199
Envelope 3	3/24/2023 4:00:29 AM	39	0	0	gE P	0.094	IMEI359368090489199

Observer Meas. Date page without Insight Rail license





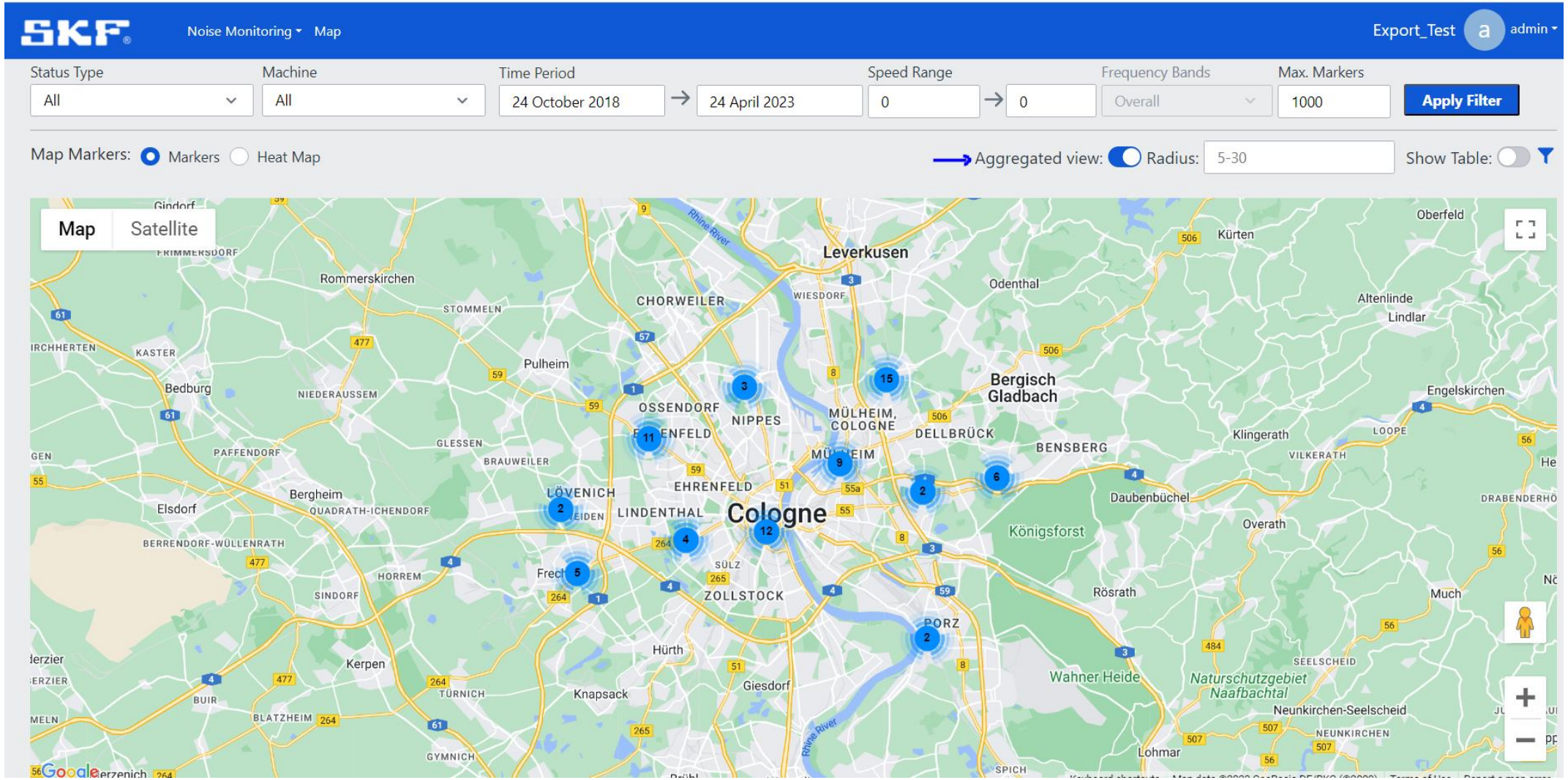
Observer Meas. Date page with Insight Rail license – SensorIdentifier column is added to the table.



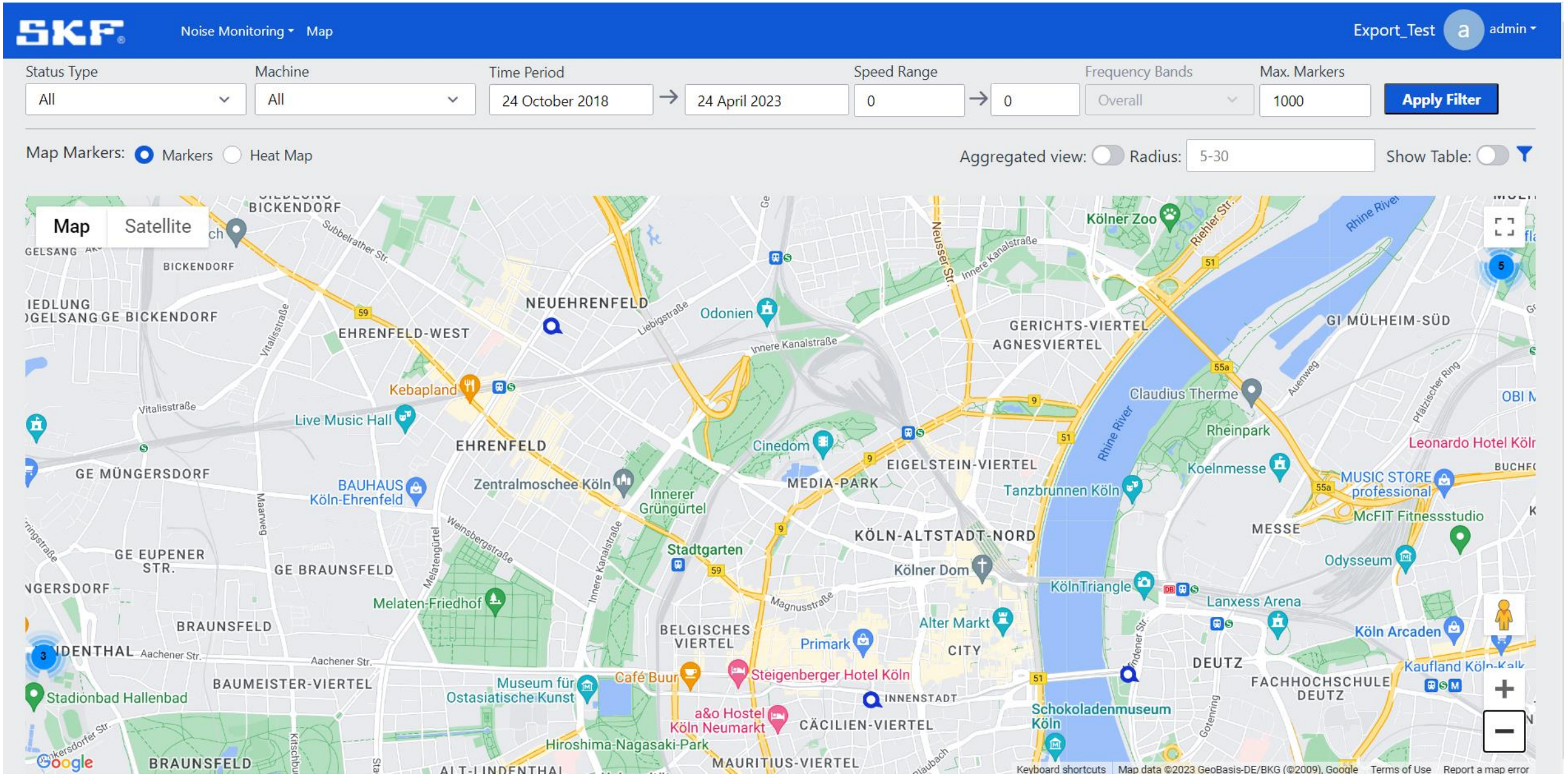
Web Client

Add a switch to toggle between an aggregated (with counter) and non-aggregated view in the Noise Monitoring application.

In the noise monitoring application, we have added a switch to enable/disable aggregated view. If it's aggregated view, it is with counters. Else it will show a single marker with direction.



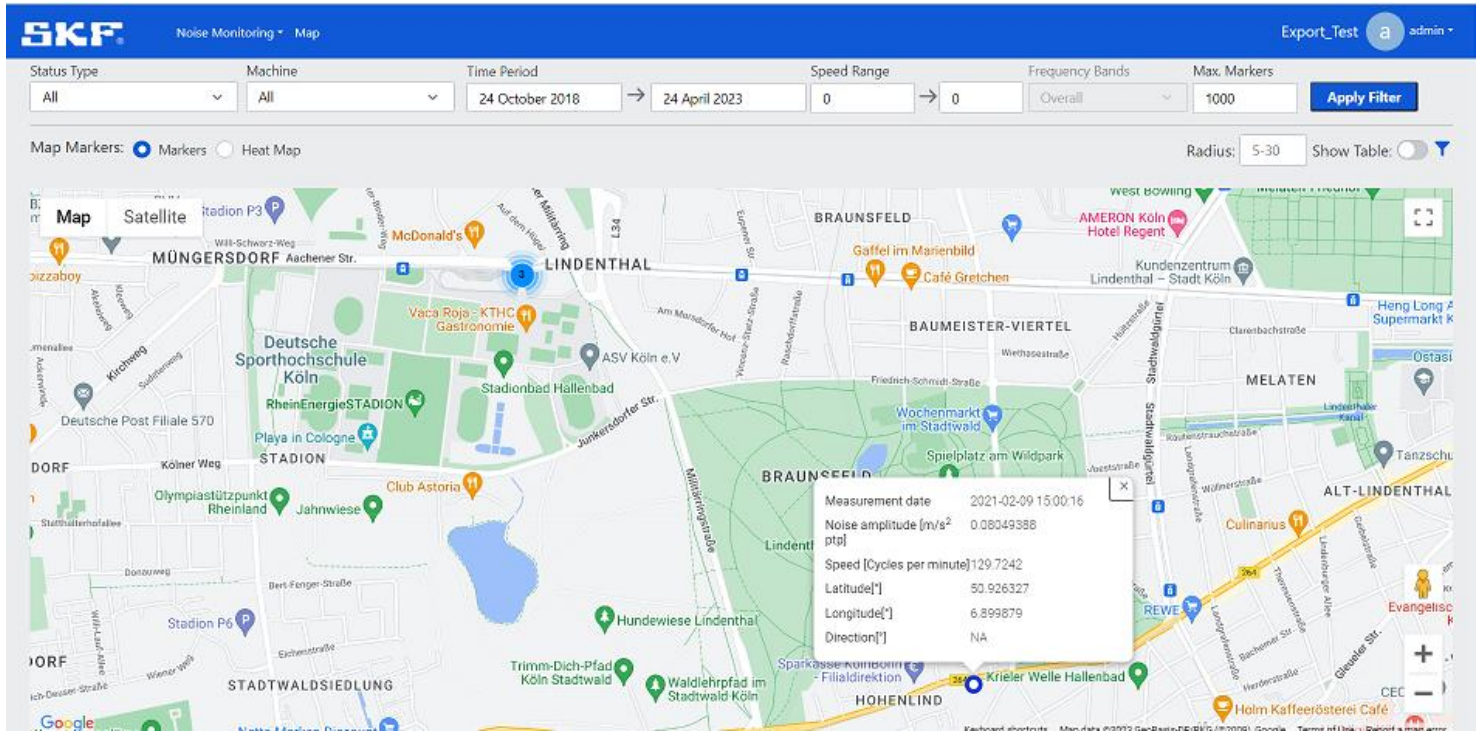
With Aggregated view switched off, we can see individual single markers with direction.

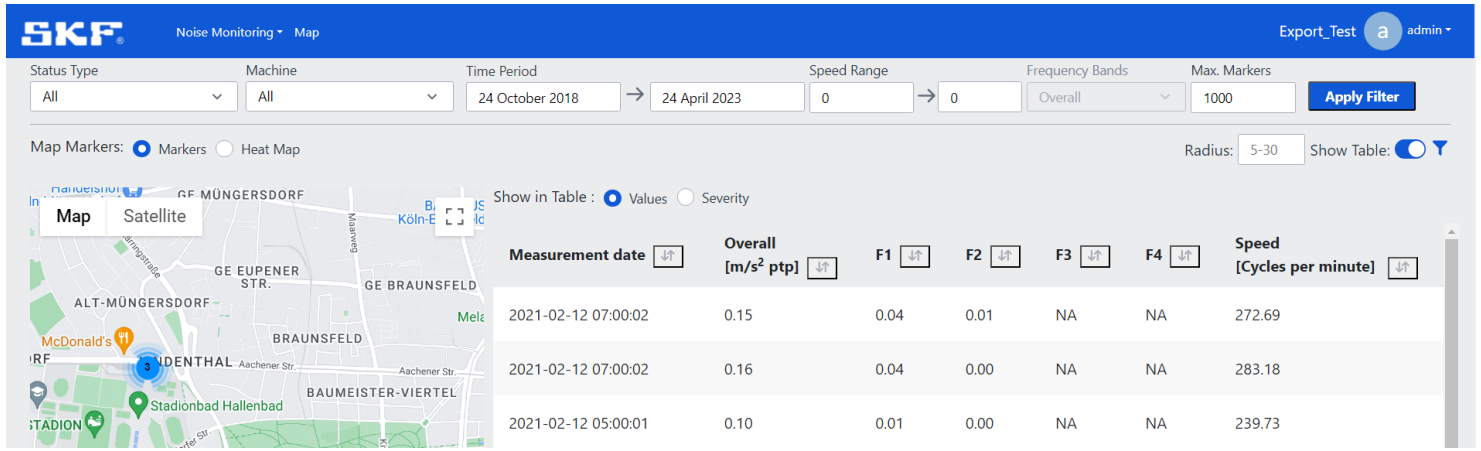


BUG FIXES

Noise Monitoring web UI reporting unreasonable speeds

Web UI is reporting speeds over 500 km/h, which is unreasonable. There is a conversion required from CPM(Cycles per minute) to km/hr. Currently, we have modified the web client UI to show the correct unit as configured in Observer.





Measurement Details

Measurement Details

Measurement date
2021-02-09 15:00:16

Noise amplitude per track side [m/s^2 PtP]
0.08049388

Speed [Cycles per minute]
129.7242

Direction
NA

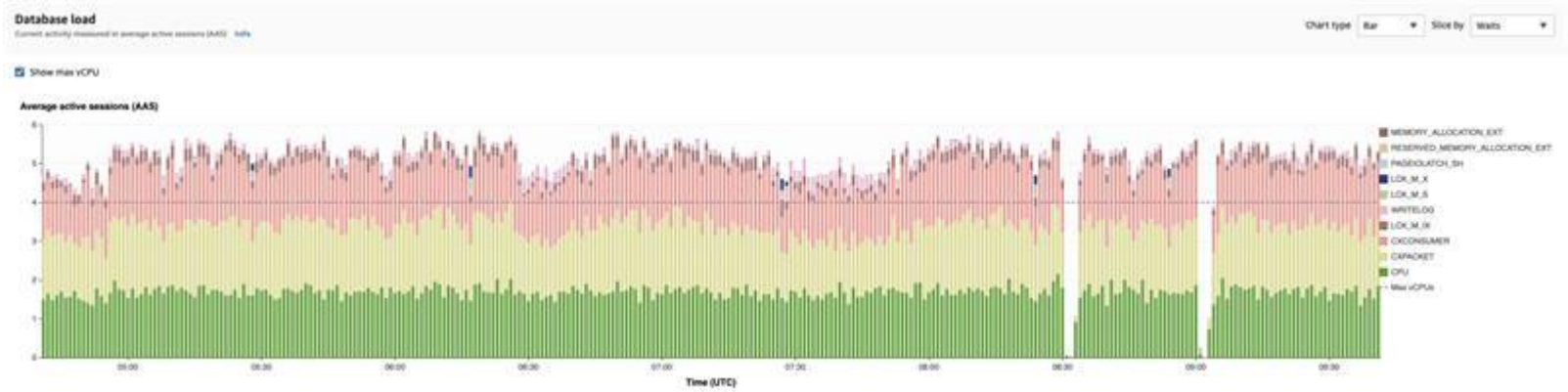
System Information

Mic Point Name
RS3_R_MIC test bitte anlassen

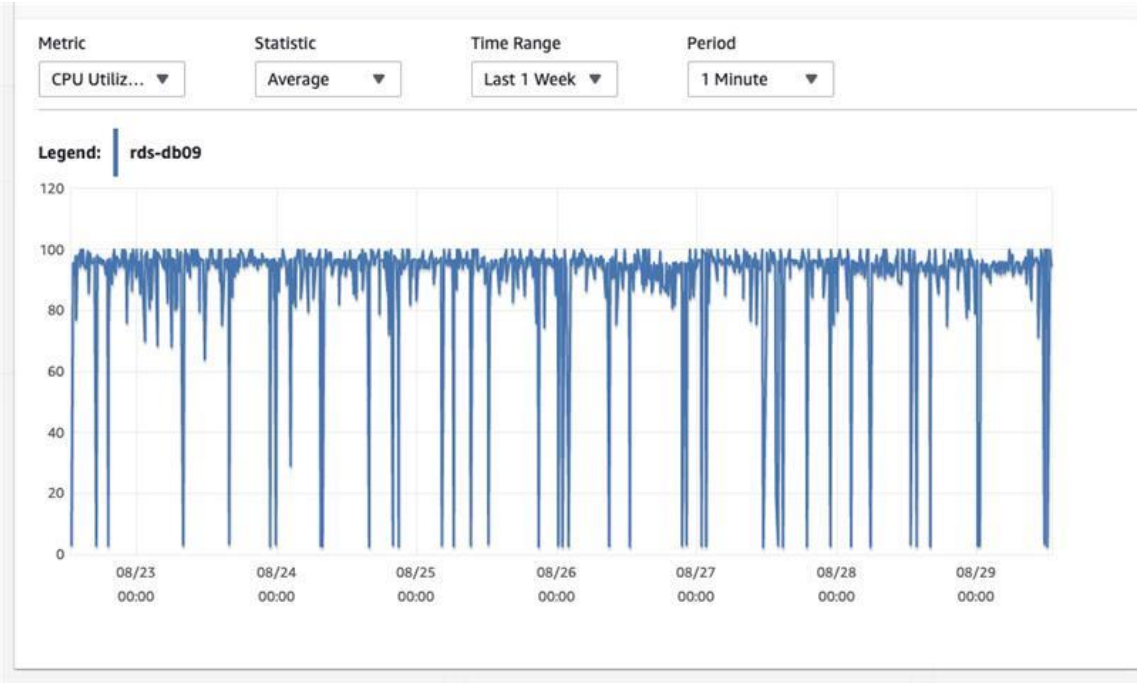
Changes for performance issue fix – Data delay for 3-4 days

This was a customer issue where we found there was a data delay for 3-4 days with the following performance issue. There was a DB Null conversion error, and the measurement buffer was set to 12 months. Buffer is now set to 1 day, and conversion is taken care of.

Performance diagram



CPU load close to 100%



Monitor service stops automatically after some time

Error:

2022-07-14 09:30:04(UTC) Runtime Error. System.OverflowException: Arithmetic operation resulted in an overflow.
  at CM.MonitorCore.CMRailMoCalc.EstimateTimeStampsAndSummarizeDataForMetersTravelled(CMDbProviderFactory dbFactory, CMRailwaySectionCaptureE railwaySectionCapture)
  at CM.MonitorCore.CMRailMoCalc.InitSummaryCalculation()
2022-07-14 09:30:04(UTC) Internal. 13.2.194.0 ***SYSTEM STOPPED*** (Reason : Error)

Quick fix / Workaround:

- SELECT * FROM RailwaySectionCapture where Processed = 0 and CalculatedDistance >0 and RailSettingIDNode >0
- Update RailwaySectionCapture set Processed=1 where Processed = 0 and CalculatedDistance >0 and RailSettingIDNode >0

Fix:

Added null checks and checks for variables used for divided by zero values to fix arithmetic overflow exception.

Observer data transmission delay

Issue:

The captures were “discarded” based on the current day. If buffered data was arriving, a lot was “discarded”.

Not a bug in IMx.

IMx will continue to store as long the memory can store a complete capture.

Example
IMx day 1, storing 300 capture, communication to monitor ok. Monitor stores 50 and truncate 250
IMx day 2, storing 300 capture, communication to monitor Not ok.
IMx day 3, storing 300 capture, communication to monitor Not ok.
IMx day 4, storing 35 capture, memory full, communication Not to monitor ok.
IMx day 5, storing 0 capture, communication to monitor Not ok.
Communication back**, Monitor stores 50 captures form day 2 and truncate the rest**.
IMx day 6, storing 300 capture, communication to monitor ok. Monitor stores 50 and truncate 250

Monitor should store
IMx day 1, Monitor stores 50
Communication back day 5
Monitor stores 50 captures form day 2
Monitor stores 50 captures form day 3
Monitor stores 35 captures form day 4
Monitor stores 0 captures form day 5

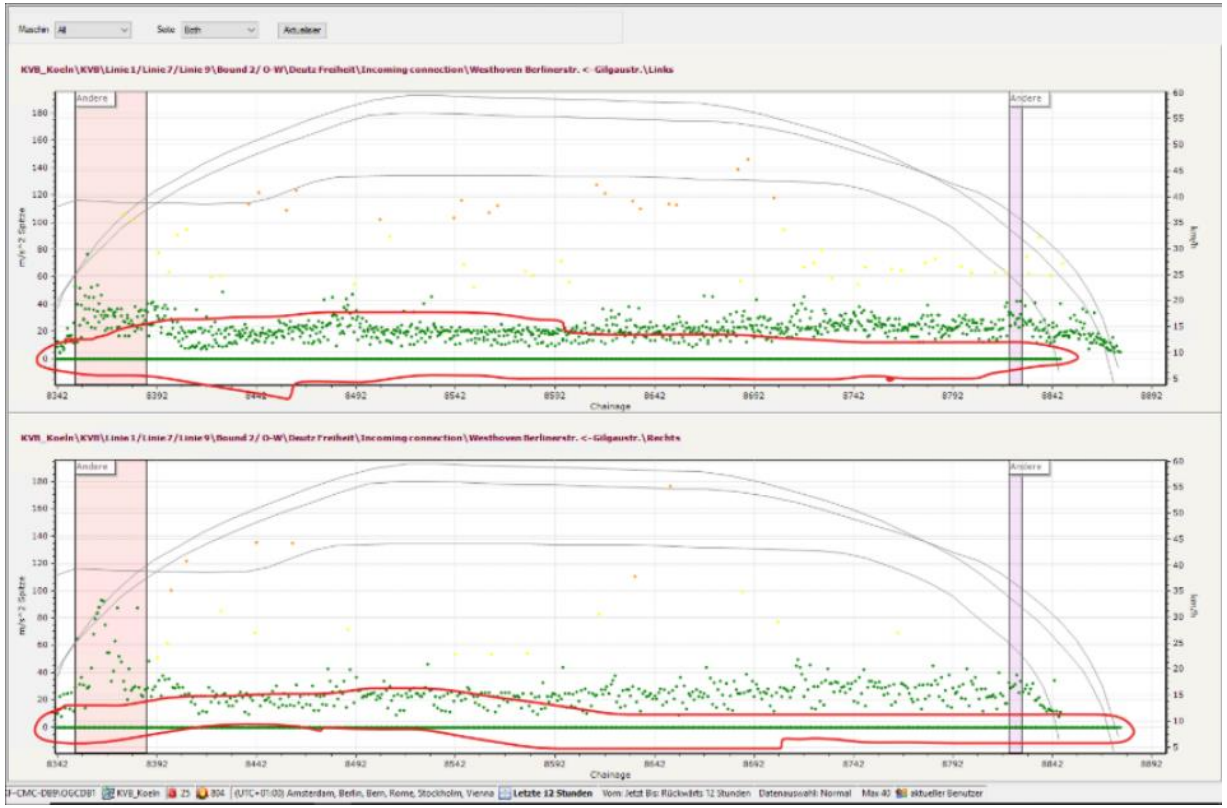
Fix:

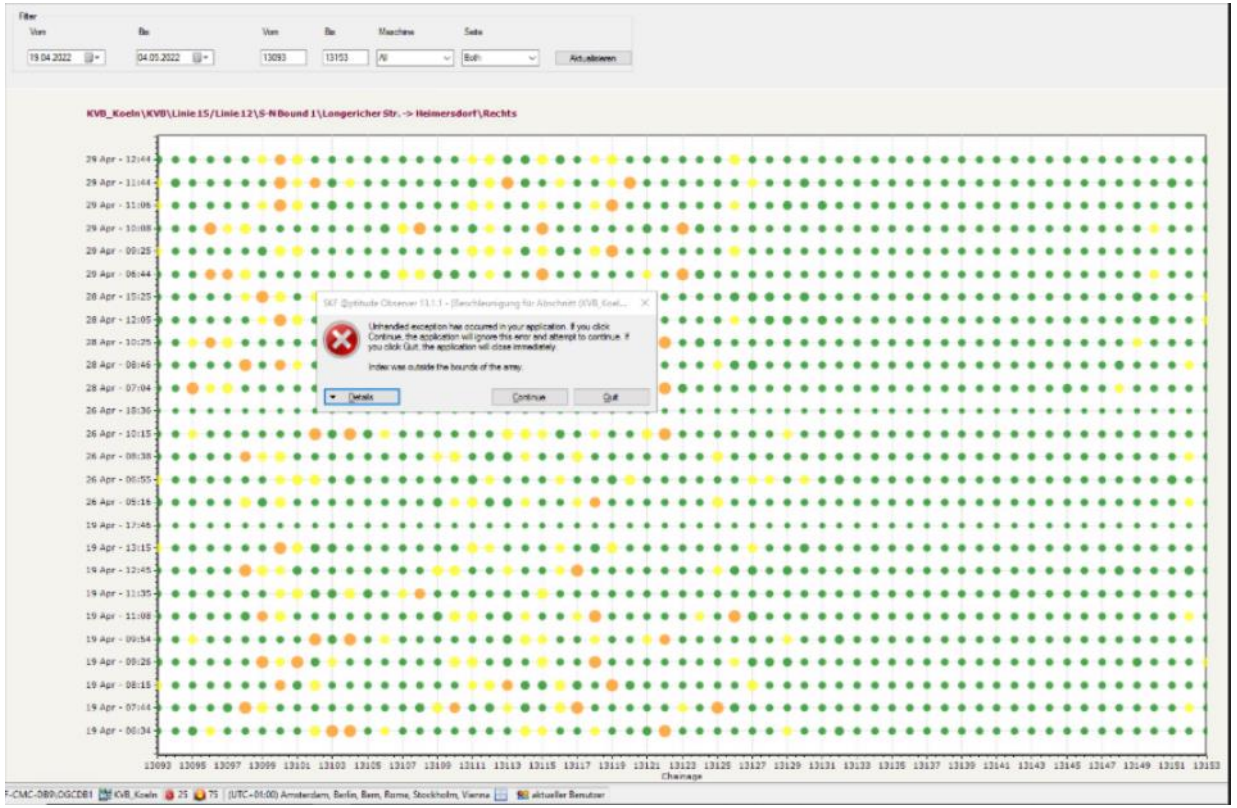
Changes are made in the method used for EventCapture count for the present day. The time was always counted from the present day 12AM and not from the time that monitor had stopped working.

Exception when clicking on Zero values in Bubble chart

Issue:

Error message comes also when the user clicks on all 0 points along one record from left to right, not only in the left or right corner. Furthermore, it looks like there is some faulty "0" data coming into the database.





Fix:

Adding checks in all places where there were properties of objects being accessed. We check for the object being null and only then access the property.

SKF