OPC UA Specifications for Sensoft Multiline. Version 1.0.8

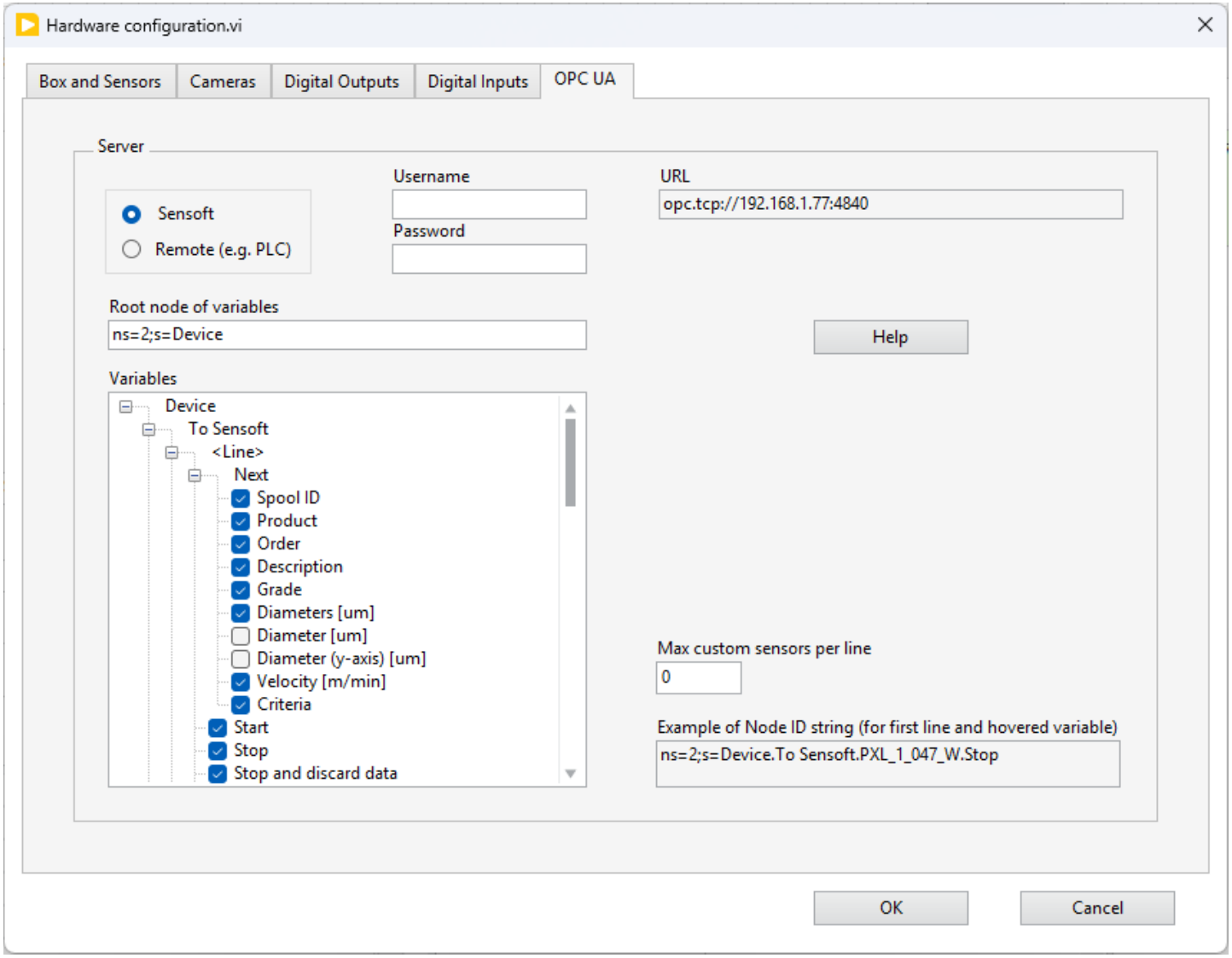


Figure 1: ***OPC UA*** *page of the* ***Hardware configuration*** *dialog window.*

A change log for this document is given at the end.

Node ID structure

Some variables, like Start for starting a measurement, are written by the remote device (typically a PLC). They are all in one OPC folder node, different from the OPC folder node containing the variables only read by the remote device.

The node ID on the OPC UA server of the variables written by the remote device is:

<Root_node>.To Sensoft.<Var_name>

where:

<Root_node> is the field **Root node of variables** in Figure 1

<Var_name> is the variable name as listed in Chapter **To Sensoft**. The variables that are specific to one line start with <Line>.

<Line> is the name of the line the variable applies to (field **Line name** on page **Box and Sensors**). Only the lines (i.e. the sensors) specified as **Active** on page **Box and Sensors** appear and count. If **Line name** contains a dot (the node delimiter character in OPC UA), then **Line name** is escaped by adding double quotation marks around it, e.g. <Line> = "Pos. 1"

Correspondingly, the variables only read by the remote device have node ID:

<Root_node>.From Sensoft.<Var_name>

where now the variable name is listed in Chapter **From Sensoft**.

The resulting variable tree can be seen in the field **Variables** in Figure 1. The variables are the rows that have a checkbox. The field **Example of Node ID string (for first line and hovered variable)** in Figure 1 displays a resulting node ID, based on the values set in the fields.

Sensoft writes to all OPC variables. It writes to **From Sensoft** variables after measurement events (e.g. when a fault was detected). It writes to **To Sensoft** variables to maintain them consistent with its internal state (see peculiarities at the end of Chapter **To Sensoft**), so that the PLC can also read them.

Variables

To Sensoft

This chapter lists all OPC UA variables that can be written by the remote device and are read by Sensoft.

Variable name	Data type	Description
<Line>.Next.Spool ID	String (up to 255 characters)	Corresponds to the field **Spool ID** of spool **Next** of line <Line> on page **Criteria** of Sensoft.
<Line>.Next.Product	String (up to 255 characters)	Corresponds to a hidden field **Product** of spool **Next** of line <Line> on page **Criteria** and identifies the product the spool belongs to. It is shown on page **Faults** and **Spools**.
<Line>.Next.Order	String (up to 255 characters)	Corresponds to the field **Order** of spool **Next** of line <Line> on page **Criteria**.
<Line>.Next.Description	String (up to 255 characters)	Corresponds to the field **Description** of spool **Next** of line <Line> on page **Criteria**.
<Line>.Next.Grade	String (up to 255 characters)	Corresponds to the field **Grade** of spool **Next** of line <Line> on page **Criteria**.
<Line>.Next.Diameters [um]	Array (up to 16 elements) of Double	Corresponds to the field **Nom. diameter [µm]** of spool **Next** of line <Line> on page **Criteria**.The number of elements in this variable is the number of axes of line <Line>. If **Settings.Diam x equal Diam y** is True, then all elements of **Diameters [um]** are equal, and only the first element is displayed on page **Criteria**. If you set **Diameters [um]** to an array whose elements are not all equal, then **Settings.Diam x equal Diam y** is automatically changed to False. The array size **Diameters [um]** must be the number of axes or 1. In the latter case all elements are assumed equal. You can determine the number of axes by reading <Line>.**Next.Diameters [um]** before writing it. The array is initialized by the OPC UA server with size 1, therefore OPC clients with fixed array size may be able to use this variable only for round filaments. Invalid value handling: If the array size is not equal to the number of axes or 1, then the whole array is ignored. If elements are negative, 0 or NaN, then these elements are ignored, i.e. remain unchanged.
<Line>.Next.Diameter [um]	Double	**Legacy**. For new applications use <Line>.**Next.Diameters [um]**. Corresponds to the first element of field **Nom. diameter [µm]** of spool **Next** of line <Line> on page **Criteria**. Invalid value handling: Negative values, 0 and NaN are ignored.
<Line>.Next.Diameter (y-axis) [um]	Double	**Legacy**. For new applications use <Line>.**Next.Diameters [um]**. Corresponds to the field **Nom. diameter (y-axis) [µm]** of spool **Next** of line <Line> on page **Criteria**.This field is visible only if **Settings.Diam x equal Diam y** is False. **Settings.Diam x equal Diam y** is automatically set to False if **Diameter (y-axis) [um]** is different from **Diameter [um]** and 0. **Diameter (y-axis) [um]** = 0 means it follows **Diameter [um]**. Invalid value handling: Negative values and NaN are ignored.
<Line>.Next.Velocity [m/min]	Double	Corresponds to the field **Speed** of spool **Next** of line <Line> on page **Criteria**, i.e. to the velocity of the wire at spool start. If you write the current wire velocity to <Line>.**Velocity [m/min]**, it is recommended to write the wire velocity to <Line>.**Next.Velocity [m/min]** just before <Line>.**Start**. Invalid value handling: NaN is ignored.

<Line>.Next.Criteria	Array (up to 64 elements) of Strings	Corresponds to the field **Criteria** of spool **Next** of line <Line> on page **Criteria**. OPC UA clients with fixed array size can work with an array of size 10, using empty string elements for the excess elements. For this reason the array is initialized by the OPC UA server with size 10.
<Line>.Start	Boolean	Setting it to True corresponds to pressing the **Start** button of line <Line> on page **Main**. The variable is latched back to False by Sensoft. Invalid value handling: False is ignored. The command is not executed if the line is already measuring. The variable is nevertheless latched back. Note: It is good practice to check that From Sensoft.<Line>.**Measuring** is False before issuing the **Start** command. Spool change status is updated by default once per second, so that a fast **Stop** - **Start** sequence can be missed. In addition it may take 1-2 s after the **Stop** command for From Sensoft.<Line>.**Measuring** to be updated.
<Line>.Stop	Boolean	Setting it to True corresponds to pressing the **Stop** of line <Line> on page **Main**. The variable is latched back to False by Sensoft. Invalid value handling: False is ignored. The command is not executed if the line is not measuring. The variable is nevertheless latched back. Note: It is good practice to check that From Sensoft.<Line>.**Measuring** is True before issuing the **Stop** command. It may take 1-2 s until From Sensoft.<Line>.**Measuring** is changed to False.
<Line>.Stop and discard data	Boolean	Setting it to True corresponds to pressing the **Stop** button of line <Line> on page **Main** and discarding all the measured data, i.e. not saving it to disk. The variable is latched back to False by Sensoft. Invalid value handling: False is ignored. The command is not executed if the line is not measuring. The variable is nevertheless latched back. Note: It is good practice to check that From Sensoft.<Line>.**Measuring** is True before issuing the **Stop and discard data** command. It may take 1-2 s until From Sensoft.<Line>.**Measuring** is changed to False.
<Line>.Velocity [m/min]	Double	Current filament velocity of line <Line>. If measured and updated often, the position of faults and mean data will be more accurate. If you write the current filament velocity to <Line>.**Velocity [m/min]**, it is recommended to write the filament velocity to <Line>.**Next.Velocity [m/min]** just before <Line>.**Start**. Invalid value handling: NaN is ignored.
<Line>.Custom sensors.<Index>.Values	Array of Double	Custom data, e.g. the diameter, of line <Line>, measured by an external sensor, e.g. a diameter sensor. You can attach up to 16 custom sensors to each line. Each custom sensor of a line has a number <Index>, ranging from 0 to **Max custom sensors per line** - 1, a parameter in the OPC UA dialog window (Figure 1). Each scalar custom data, e.g. diameter and ovality, needs a separate <Index>, even if they come from the same physical sensor. The array **<Line>.Custom sensors.<Index>.Values** holds temporal elements of the scalar data, e.g. [$\varnothing(t_1)$, $\varnothing(t_2)$, ... , $\varnothing(t_n)$]. Only new data points should be written to **Values**, overwriting previous data. The properties of **Values**, e.g. their name, are specified by the variables **<Line>.Custom sensors.<Index>.Settings.*** . Faults are triggered if elements of **Values** satisfy current criteria. **Values** are recorded in the mean data according to the instruction in **<Line>.Custom sensors.<Index>.Settings.Mean data**. Timing:The timestamp at which Sensoft is notified that the variable has changed is used as measurement time of the last array element. It is typically within 100 ms from the write time and has the advantage that it does not need clock synchronization. The measurement time of other elements is set equidistantly between

		this timestamp and the timestamp of the last change of **Values** of the same line. If the last timestamp is before **<Line>.Measurement start time**, then the start time is used instead. If you measure the custom sensor at a high rate, it is good practice to send a **Values** array for each line about every second and to reduce the array size to about 1000 elements. If measurement restart after a period without measurements, it is advised to send the first data point alone (with a subsequent pause of 100 ms). The Sensoft OPC UA server has a publishing interval of 100 ms: writing the same variable more often leads to value loss.
<Line>.Custom sensors.<Index>.Fault	Array of Double	A fault detected by the custom sensor <Index> and passed as is to Sensoft. There are two ways to pass faults (filament imperfections found based on data from custom sensor <Index>) to Sensoft: 1. The custom sensor analyses its data, detects a fault based on its criteria, and writes the fault data to this OPC UA variable. 2. The custom sensor writes its raw values to **Values**, Sensoft detects a fault based on Sensoft's criteria. If faults are passed by this variable, i.e. by the first of the two ways, it is not needed to write **<Line>.Custom sensors.<Index>.Values**. Note however that only if **Values** are passed, mean data and fault profiles can be saved. The elements of **<Line>.Custom sensors.<Index>.Fault** are: Criterion, Size [in **<Line>.Custom sensors.<Index>.Settings.Unit**], *Time [seconds since 1904-01-01T00:00:00.00 UTC], Position [m], Length [m]*. The elements in *italics* are optional. Criterion is the index of a criterion, which refers to Custom sensor <Index>, i.e. has its **Name** in it. Send **Fault** as soon as possible and avoid mixing with faults coming from **Values**, since Time and Position [m] should be non-decreasing. Between two write commands to the same **<Line>.Custom sensors.<Index>.Fault** there must be at least 100 ms (publishing interval of the Sensoft OPC UA server) or the former writing is lost. Multiple faults can be written in one command by padding each fault to 10 elements and concatenating them. Thus an array of 10*N elements contains N faults, the elements with indices 10*i contain their criterion, those with indices 10*i+1 their size and so on (i = 0..N-1). OPC UA clients with fixed array size can use array size 10, sending one fault at the time. Invalid value handling: An empty array is ignored. If Time is missing, zero, or more than 5 s from system time, the notification timestamp is taken (see description in **Values**). If Position [m] is missing, zero or negative, the position is calculated from Time. If int(Criterion) does not correspond to a criterion about Custom sensor <Index>, the fault is ignored. If Length [m] is missing or negative it is set to NaN.
<Line>.Custom sensors.<Index>.Settings.Name	String (up to 10 characters)	The name, e.g. Diam, of what Custom sensor <Index> measures. If the Custom sensor has no name, i.e. **Name** is an empty string, it is considered inactive, and no data is processed. **Name** is used to specify **Criteria**, in the legends of the graphs displaying the data and in the TDMS file (to indicate the type of fault in the fault list and as column title for the mean value and fault profiles). Default: "". Invalid value handling: Strings longer than 10 characters are truncated. The following names used for built-in fault types and are invalid (ignored): L, LU, Lumps, N, NE, Neck, Neck-Downs, Neckdown, Neckdowns, Necks, R, Ro, Rough, Roughness, Sd, Std Dev, Std Dev., Std. Dev, Std. Dev., Stddev and all its lowercase and mixed-case variants.

<Line>.Custom sensors.<Index>.Settings.Unit	String (up to 10 characters)	The unit, e.g. µm, of what Custom sensor <Index> measures and the unit of **Values** and the element Size of **Fault**. The unit is used in the **Criteria** and in the TDMS file in the column title for the mean value and fault profiles. In Criteria input **Unit** can written as is or abbreviated by the letter "u". Valid criteria are pretty-printed by Sensoft writing out **Unit**. For example, with **Name** = "El. field" and **Unit** ="V/m", both "Warning if El. field > 4 V/m" and "warn if El. field > 4 u" are accepted and pretty-printed "Warning if El. field > 4 V/m". Default: "unit". Invalid value handling: Strings longer than 10 characters are truncated.
<Line>.Custom sensors.<Index>.Settings.Nominal value	Double	A nominal value, i.e. the desired value, of Custom sensor <Index>, in units of **Unit**. The nominal value allows to write criteria relative to it, such as "Alarm if Diam - nom. > 2 µm" or "Alarm if Diam - nom. > 2%". The latter gives an alarm if **Values** - **Nominal value** > 2% of **Nominal Value**. This variable is only used if you use the string "- nom" in **Criteria**. Faults of relative criteria are plotted on page Faults relative to **Nominal Value**. This means e.g. that diameter faults caught with "Alarm if Diam - nom. > 10 µm" are visible together with lump faults, but those caught with "Alarm if Diam > 1510 µm" are not. **Nominal value** is accessed at measurement start and successive changes are ignored. Default: NaN (i.e. relative criteria are ignored). Invalid value handling: NaN is ignored.
<Line>.Custom sensors.<Index>.Settings.Hysteresis	Double	The hysteresis for the threshold of Custom sensor <Index>, in units of **Unit**. Set it approximately to the noise of the signal, or if unknown, a bit higher than the repeatability of the sensor. The hysteresis allows to avoid fake faults due to noise when the signal is near the threshold. At the start and the end of a fault the signal is at the threshold, and noise may make the signal cross the threshold many times. These crossings are merged with the real fault by making it end only when Signal < Threshold - Hysteresis. Default: 0. Invalid value handling: NaN and negative values are ignored.
<Line>.Custom sensors.<Index>.Settings.Max fault length [m]	Double	A fault normally ends when the signal returns below threshold. If the signal does not return below threshold it will end after **Max fault length [m]**. If its stays above threshold indefinitely, every **Max fault length [m]** a fault is returned. Default: 100 m. Invalid value handling: NaN, zero and negative values are ignored.
<Line>.Custom sensors.<Index>.Settings.Fault display	UInt32	How faults of Custom sensor <Index> are shown in the faults graph (in Sensoft on page **Faults**). The values are: 0: None. 1: Point. A point at the fault's max. 2: Length. Point plus the segments Start-Max and Max-End 3: Profile. Point plus a curve of **Values** above threshold. Long profiles are sampled down to 100 values by taking max resp. min (for **Criteria** involving ">" resp. "<"). Default: 3. Invalid value handling: Values not specified here are ignored.
<Line>.Custom sensors.<Index>.Settings.Mean data	UInt32	Specifies how periodic data of Custom sensor <Index> is saved to disc and how it is shown in Sensoft in **Mean graph** on page **Statistics**. The values are: 0: None. No mean data is saved to disc and no data is shown in **Mean graph**. No data is written to

		OPC UA variable **From Sensoft.<Line>.Mean data**. 1: At **Mean data interval [m]**. Mean data (arithmetic average) is written to disc every Mean data interval [m]. If data is missing, NaN is written. 10..11: Like 0..1, but using the **maximal** value in the interval instead of the mean value 20..21: Like 0..1, but using the **minimal** value in the interval instead of the mean value 30..31: Like 0..1, but using the **standard deviation** instead of the mean value Default: 1. Invalid value handling: Values not specified here are ignored.
Start all	Boolean	Setting it to True corresponds to pressing the **Start all** button on page **Main**. The variable is latched back to False by Sensoft. Invalid value handling: False is ignored. The command is not executed if Sensoft is already measuring. The variable is nevertheless latched back.
Stop all	Boolean	Setting it to True corresponds to pressing the **Stop all** on page **Main**. The variable is latched back to False by Sensoft. Invalid value handling: False is ignored. The command is not executed if Sensoft is not measuring. The variable is nevertheless latched back. Note: It may take up to 1-2 s until all **From Sensoft.<Line>.Measuring** are changed to False.
Switch to tab	UInt32	Switches to that page. Affects only what is displayed on the graphical user interface. The values are Main = 0, Criteria = 1, Faults = 2, Photos = 3, Statistics = 4, Spools = 5, Settings = 6, Admin = 7, Debug = 8, Variables = 9. Invalid value handling: Values larger than the number of available elements are ignored.
Switch to line	String	Selects the line with that name (or index). Corresponds to clicking the line on the right pane. Affects only what is displayed on the graphical user interface, e.g. on page **Faults** loads the faults of the current or last spool of that line. The line index is "0" for the first line from the top, "1" for the second and so on. Invalid value handling: Strings that are neither elements of **Line names** nor integer numbers between 0 and Size(**Line names**) - 1 are ignored.
Settings.Mean data interval [m]	Double	Corresponds to the field **Mean data interval [m]** on page **Settings**. Invalid value handling: Negative values and NaN are ignored.
Settings.Data folder	String (up to 255 characters)	Corresponds to the field **Data folder** on page **Settings**. Note: Changes to **Settings.Data folder** are executed only while not measuring, since it affects data saving. Not measuring means that **Stop all** was pressed and the variables **From Sensoft.<Line>.Measuring** are False (which may take 1-2 s). Invalid value handling: The command is reverted while measuring.
Settings.Diam x equal Diam y	Boolean	Corresponds to clicking with the right mouse button on the diameter field and select **Use different diameters for x and y** from the context menu. If True, all wires are considered round: **Next.Diameter (y-axis) [um]** of all lines is set to 0, meaning it is considered equal to **Next.Diameter [um]** and the field **Nom. diameter (y-axis) [µm]** on page **Criteria** is hidden. If False, then flat wires with **Next.Diameter (y-axis) [um]** different from **Next.Diameter (y-axis) [um]** are

		allowed and the field **Nom. diameter (y-axis) [µm]** on page **Criteria** is shown. **Settings.Diam x equal Diam y** is automatically set to False when a line has an y-axis diameter different from that of the x-axis.

Sensoft is subscribed to the variables listed in **To Sensoft**, i.e. receives the variable when its value on the server changes. The publishing interval is 50 ms, which means that changes happening faster are not received. Sensoft treats changes to the values of the OPC UA variables like manual changes to the corresponding fields in Sensoft. This means in particular that manual changes are not disallowed. The variables in **To Sensoft** are updated by Sensoft when the value of the corresponding field changes, both if the change originated from a manual change and from an OPC change. The exceptions are latching Booleans and **Velocity [m/min]**. The **To Sensoft** variables are consistent with the local variables in Sensoft, with the following peculiarities:

- <Line>.**Start**, <Line>.**Stop** and <Line>.**Stop and discard data** do not correspond to a local variable. They are latching Booleans, i.e. written by Sensoft just to latch them back to False after an OPC change. To read if a measurement is going on use From Sensoft.<Line>.**Measuring**
- <Line>.**Velocity [m/min]**, <Line>.**Custom sensors**.<Index>.**Values** and <Line>.**Custom sensors**.<Index>.**Fault** are input-only for Sensoft, and are not written to by Sensoft. To read the velocity use From Sensoft.<Line>.**Velocity [m/min]**, to read faults use From Sensoft.<Line>.**Last fault.*** .

When Sensoft starts in OPC UA mode or changes to OPC UA mode, its updates all **To Sensoft** variables to match their corresponding fields. It updates also the **From Sensoft** variables **Line names** and <Line>.**Measuring**, while the others are only updated while measuring. The variables <Line>.**Last fault.*** and <Line>.**Alerts.*** are reset at spool start.

From Sensoft

This chapter lists all OPC UA variables that can be read by the remote device and are written by Sensoft.

Variable name	Data type	Description
<Line>.Last fault.Nr	Int32	The number of the last fault, i.e. also the number of faults since the beginning of the measurement. Supports Historical access, see note 1.
<Line>.Last fault.Time	DateTime	Time of the last fault. Supports Historical access, see note 1.
<Line>.Last fault.Position [m]	Double	Position of the last fault. Supports Historical access, see note 1.
<Line>.Last fault.Size [um]	Double	Size of the last fault. Supports Historical access, see note 1.
<Line>.Last fault.Type	UInt32	Type of the last fault (Lump = 0, Neck-down = 1, Roughness = 2, Custom sensor <Index> = 10 + <Index>). Supports Historical access, see note 1.

<Line>.Last fault.Severity	UInt32	Alarm = 1, Warning = 2, Alarm and Warning = 3. Supports Historical access, see note 1.
<Line>.Last fault.In criteria	Array of Int32	Indices of the criteria the fault satisfies. Supports Historical access, see note 1.
<Line>.Last fault.Velocity [m/min]	Double	Speed at the moment of the last fault. Supports Historical access, see note 1.
<Line>.Last fault.Length [m]	Double	The length of the fault, i.e. how long the relevant physical property was over the threshold. For faults of **Type** Lump and Neck-down the fault length cannot be measured and the value will be NaN. Supports Historical access, see note 1.
<Line>.Measurement start time	DateTime	Start time of the latest measurement.
<Line>.Measuring	Boolean	True while the line is measuring.
<Line>.File path	String (up to 255 characters)	Path of the TDMS file, relative to **To Sensoft.Settings.Data folder** . The TDMS file contains all measured data and information about the spool.
<Line>.Position [m]	Double	Current position of the wire, i.e. wire length since the start of the measurement. The time of the position measurement is the stored in the Source timestamp of this OPC UA variable. Updated only while measuring, in that case by default once a second [note 2].
<Line>.Velocity [m/min]	Double	Current velocity of the wire. Updated only while measuring, in that case by default once a second [note 2].
<Line>.Signal [%]	Double	Measured signal in % of expected signal. Expected signal is the signal level with clean sensor windows with a filament of nominal diameter (if <Line> is measuring), respectively without filament (if <Line> is not measuring). In the Sensoft GUI **Signal [%]** is shown only in case it deviates from the expected signal (e.g. if the sensor windows are dirty) and only for measuring lines. Nominal diameter is shown in Sensoft GUI on page Criteria in the field Current.Nom. diameter [µm] and can be set for the next spool, e.g. by OPC UA with <Line>.Next.Diameter [um]. Updated by default once a second [note 2].
<Line>.Mean data	Array [up to 32] of Double	Last Mean data values. Mean data is periodic data about the spool, written to disc every **Settings.Mean data interval [m]** and displayed in Sensoft on page **Statistics** in **Mean graph**. This variable is an array containing the following elements: Time [seconds since 1904-01-01T00:00:00.00 UTC], Position [m], Relative diameter [um], Relative ovality [um], Max LU [um], Max NE [um], Max roughness [um], *Future data 0*, ..., *Future data 2, Custom sensor 0, ..., Custom sensor 15, Future data 3, ..., Future data 8*). The values in *italics* are optional, i.e. omitted if not available. This means that the size of the array is variable but not greater than 32. NaN is used for missing elements, e.g. for Relative ovality [um] with uniaxial sensors. For biaxial sensors Relative ovality [um] is signed, so that the the x and y components of Relative diameter [um] can be calculated ($Ø_x$ = Ø + ovality, $Ø_y$ = Ø - ovality). The neck-down value Max NE [um] is negative. Since Custom sensors can send data with a delay, it is not guaranteed that their mean value corresponds to **Position [m]** and **Time**. When new Custom sensor data is available, the digest of that data is sent. Here

		digest means average, max or min, depending on what is set in **<Line>.Custom sensors.<Index>.Settings.Mean data**. Updated only while measuring, in that case every **Settings.Mean data interval [m]** but not more often than once per cycle (1 Hz by default) [note 2].
<Line>.Alerts.Nr of alarms	Int32	Total number of alarms since the start of the measurement. Spool is a PASS if this variable is zero and a FAIL if greater than zero
<Line>.Alerts.Nr of warnings	Int32	Total number of warnings since the start of the measurement.
<Line>.Alerts.Alerts	Array[Nr. of criteria] of Int32	Number of warnings/alarms for each criterion.
Line names	Array[Nr. of lines] of String	The names of the lines that can measure, as they appear in the right-hand side box in Sensoft. The line names are specified on page **Box and Sensors** of the **Configuration...** window. Only **Active** lines are included, and in case of sensor groups, only one line per group. **Line names** can only change if the hardware configuration is changed. If Sensoft is not the OPC UA server, the server should provide corresponding nodes within 200 ms after a change of this variable. **Line names** can be used to get the number of lines and their ordering. Note that contrarily to the <Line> part of the other variable names, dots in line names are not escaped.
Line to head	Array[Nr. of lines] of Int32	The head index of each line. It holds information whether a sensor is active or in a group. Heads are rows in the **Box and Sensors** page of Figure 1, and often correspond to a sensor. Lines are the rows on page **Main** of Sensoft. Indices are 0-based, so the top row has index 0. If there are multiple Sensystem boxes controlled by the same pc, the enumeration of the heads continues where the previous box left off. Active heads are those with a green light in the **Active** column in the **Box and Sensors** page. Only active heads create lines, and if you are using sensor groups, for each active group only the head with the lowest index. Therefore: i) To test if a head is active (and first-of-group, if using groups), test if it is in **Line to head**, ii) If it is, at element i, its line name is **Line names**[i].
Status	UInt32	Status of the Sensoft Multiline software. The values are 0: Not running, 1: Running and ok, ≥ 2: Running with problems. The bits of the variable **Status** codify which problems the software has. For more details, please refer to sensoptic.ch/sensoft/multiline/how-to/status

Clients typically read **From Sensoft** variables by subscribing to them, so that the OPC UA server sends them a notification when the value of variable changes. Note that the variables **Last fault.Type**, **Last fault.Severity**, **Last fault.In criteria** and **Last fault.Velocity [m/min]** often have the same value as for the fault before. To get a notification each time a variable is written to, i.e. even if the value stays the same, at subscription to the variable use DataChangeTrigger = StatusValueTimestamp instead of the default value StatusValue.

Note 1: For accessing data of previous faults, the OPC UA server provides Historical Access to all **Last fault.*** variables. If Sensoft is the OPC UA server, 10'000 old points of each variable are available. A remote server is free on how many samples to give access, if any. Sensoft writes the data of each fault exactly once to the OPC UA server, therefore reading the historical data of **Last fault.*** from the measurement start time (as stored in the variable **Measurement start time**) to now without limiting the number of resulting points, should return the data from the first to the last fault (an array of length **Last fault.Nr**). For exceptions during overload, see Chapter Performance and capping. The OPC UA Source timestamp of the **Last fault.*** variables is the time of the fault, i.e. equal to **Last fault.Time**.

Note 2: The update rate of the **<Line>.*** variables is given by the value **Update rate [Hz]** on page Settings. The default value is 1 Hz.

Performance and capping

If there are many faults, writing them to OPC UA is the most CPU intensive task of Sensoft. For reference, with all OPC UA variables enabled, a typical PC begins to slow down if the total fault rate (i.e. of all lines together) is in the order of 200 - 400 faults/s.

For this reason the number of faults written to <Line>.**Last fault.*** in one second is capped to 10 faults/s for each line. Precisely, if a line has more than 10 faults in one cycle, which by default lasts 1 s, only the 10 with largest size are sent to the OPC UA server.

The workload of OPC UA can be reduced by disabling variables in the OPC UA configuration page. This is done by removing the check-mark near the variable in the **Hardware configuration** dialog (Figure 1) and has the effect that the variable is not written.

The publishing interval of the Sensoft OPC UA server is 100 ms. This means that if a client writes to the same **To Sensoft** variable twice within this period the former writing is overwritten and lost. In practice this affects only <Line>.Custom sensors.<Index>.Values and <Line>.Custom sensors.<Index>.Fault . It also means that clients cannot expect updates of **From Sensoft** variables faster than 10 Hz, even if **Update rate [Hz]** on page Settings was higher. The publishing interval does not apply to Historical Access.

Advanced settings

To access the advanced settings for OPC UA in Sensoft, click with the right mouse button in the OPC UA page of Hardware settings (Figure 2) and choose **Show advanced settings** from the context menu.

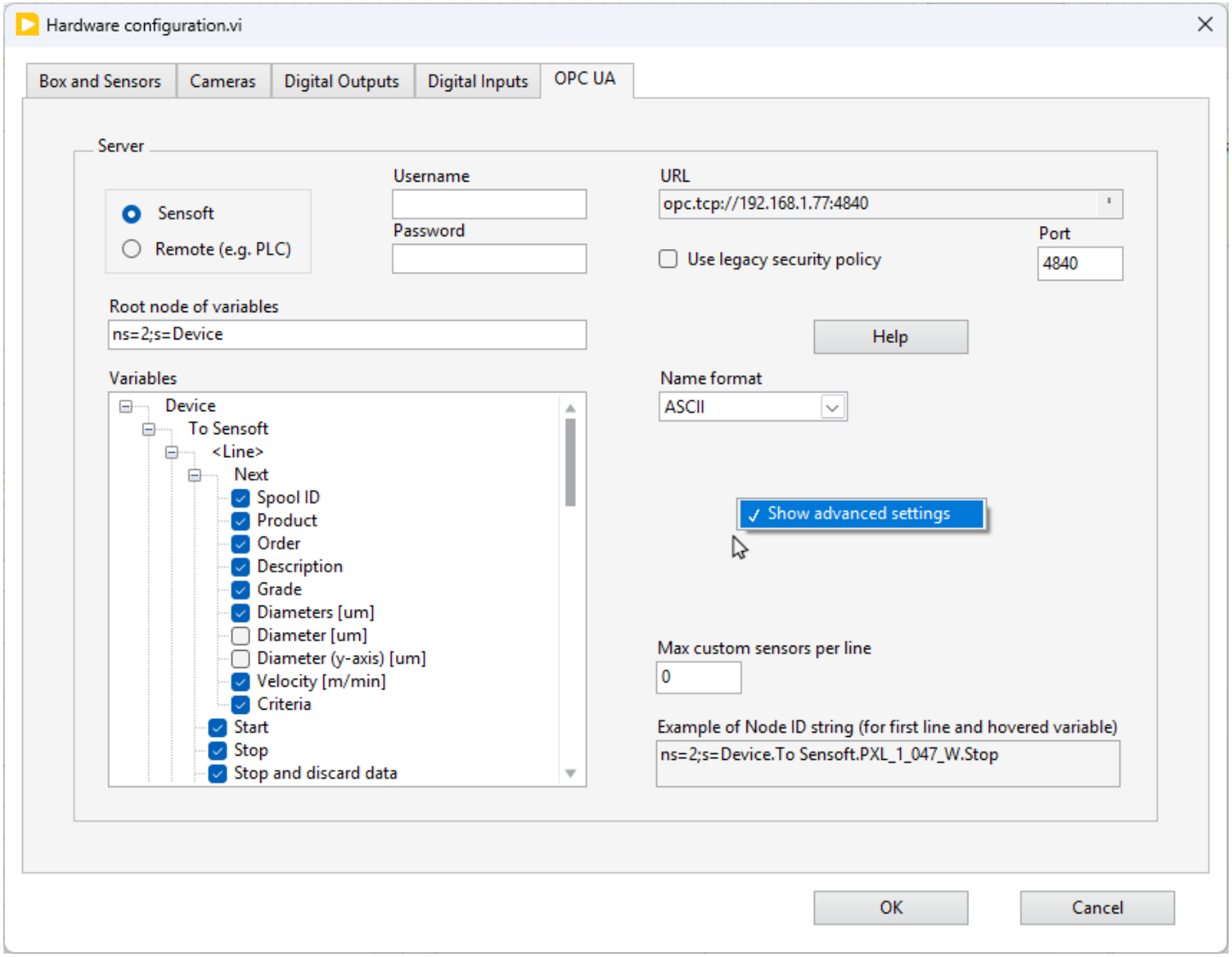


*Figure 2: **OPC UA** page of the **Hardware configuration** dialog window with Advanced settings shown. To show them, click with the right mouse button and select **Show advanced settings** from the context menu.*

Server and client

Sensoft can act either as OPC UA server or as OPC UA client. The user can choose it in the Hardware configuration dialog (Figure 2).

The server has to provide the node structure for at least the activated variables. By default, the server is Sensoft. It provides the node structure for all variables.

Port

If Sensoft is OPC UA server, the advanced setting **Port** can be used to give a non-standard port to the OPC UA server. This allows running multiple OPC UA servers on same computer, e.g. or another instance of Sensoft. The setting Port is not used and remains hidden when Sensoft is OPC UA client, there the port is set in field URL.

IEC 61131-3 names

By default Sensoft uses the OPC UA variable names listed in the tables above. They consist of ASCII characters valid in OPC UA. However, some PLCs only support OPC UA variable names made up of IEC 61131-3 characters (i.e. letters, digits, and underscores). Therefore Sensoft can optionally use IEC 61131-3 variable names, listed in the following table. Use the field **Name format** to select IEC 61131-3 (Figure 2).

If necessary, the line name <Line> will be converted to IEC 61131-3 in the following way: a) Characters that are not allowed will be replaced with underscores; b) If the resulting name is not unique (i.e. already used by a previous line), append the smallest integer greater or equal 2 that makes the name unique. As an example, the variable **To Sensoft.Line 312-1.Next.Spool ID** will be called **To_Sensoft.Line_312_1.Next.Spool_ID**.

To Sensoft	To_Sensoft
Variable name (ASCII)	**Variable name (IEC 61131-3)**
<Line>.Next.Spool ID	<Line>.Next.Spool_ID
<Line>.Next.Product	<Line>.Next.Product
<Line>.Next.Order	<Line>.Next.Order
<Line>.Next.Description	<Line>.Next.Description
<Line>.Next.Grade	<Line>.Next.Grade
<Line>.Next.Diameters [um]	<Line>.Next.Diameters_um
<Line>.Next.Diameter [um]	<Line>.Next.Diameter_um
<Line>.Next.Diameter (y-axis) [um]	<Line>.Next.Diameter_y_axis_um
<Line>.Next.Velocity [m/min]	<Line>.Next.Velocity_mpm
<Line>.Next.Criteria	<Line>.Next.Criteria
<Line>.Start	<Line>.Start
<Line>.Stop	<Line>.Stop
<Line>.Stop and discard data	<Line>.Stop_and_discard_data
<Line>.Velocity [m/min]	<Line>.Velocity_mpm
<Line>.Custom sensors.<Index>.Values	<Line>.Custom_sensors.<Index>.Values
<Line>.Custom sensors.<Index>.Fault	<Line>.Custom_sensors.<Index>.Fault
<Line>.Custom sensors.<Index>.Settings.Name	<Line>.Custom_sensors.<Index>.Settings.Name
<Line>.Custom sensors.<Index>.Settings.Unit	<Line>.Custom_sensors.<Index>.Settings.Unit
<Line>.Custom sensors.<Index>.Settings.Nominal value	<Line>.Custom_sensors.<Index>.Settings.Nominal_value
<Line>.Custom sensors.<Index>.Settings.Hysteresis	<Line>.Custom_sensors.<Index>.Settings.Hysteresis
<Line>.Custom sensors.<Index>.Settings.Max fault length [m]	<Line>.Custom_sensors.<Index>.Settings.Max_fault_length_m

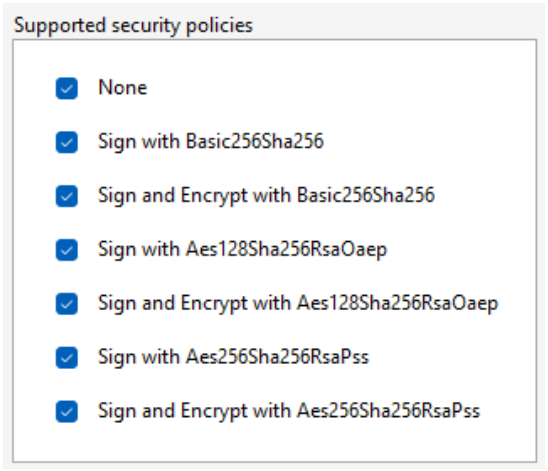
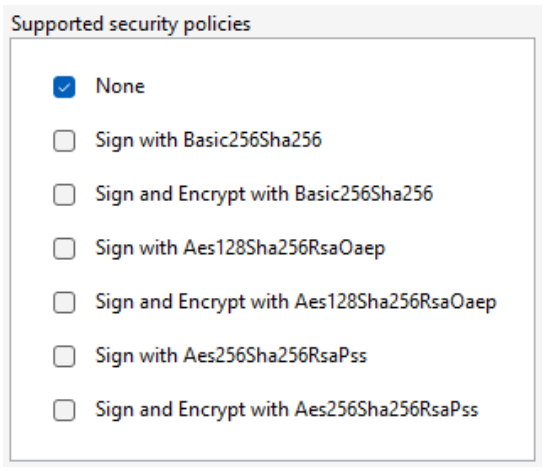
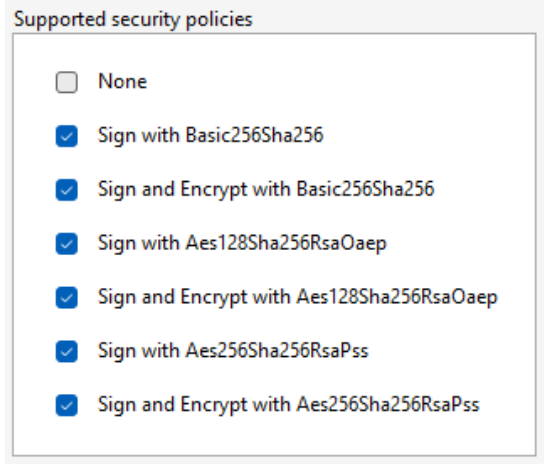
<Line>.Custom sensors.<Index>.Settings.Fault display	<Line>.Custom_sensors.<Index>.Settings.Fault_display
<Line>.Custom sensors.<Index>.Settings.Mean data	<Line>.Custom_sensors.<Index>.Settings.Mean_data
Start all	Start_all
Stop all	Stop_all
Switch to tab	Switch_to_tab
Switch to line	Switch_to_line
Settings.Mean data interval [m]	Settings.Mean_data_interval_m
Settings.Data folder	Settings.Data_folder
Settings.Diam x equal Diam y	Settings.Diam_x_equal_Diam_y

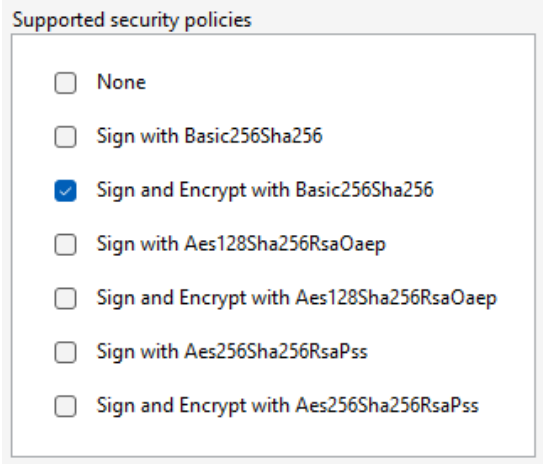
From Sensoft	From_Sensoft
Variable name (ASCII)	**Variable name (IEC 61131-3)**
<Line>.Last fault.Nr	<Line>.Last_fault.Nr
<Line>.Last fault.Time	<Line>.Last_fault.Time
<Line>.Last fault.Position [m]	<Line>.Last_fault.Position_m
<Line>.Last fault.Size [um]	<Line>.Last_fault.Size_um
<Line>.Last fault.Type	<Line>.Last_fault.Type
<Line>.Last fault.Severity	<Line>.Last_fault.Severity
<Line>.Last fault.In criteria	<Line>.Last_fault.In_criteria
<Line>.Last fault.Velocity [m/min]	<Line>.Last_fault.Velocity_mpm
<Line>.Last fault.Length [m]	<Line>.Last_fault.Length_m
<Line>.Measurement start time	<Line>.Measurement_start_time
<Line>.Measuring	<Line>.Measuring
<Line>.File path	<Line>.File_path
<Line>.Position [m]	<Line>.Position_m
<Line>.Velocity [m/min]	<Line>.Velocity_mpm
<Line>.Signal [%]	<Line>.Signal_percent
<Line>.Mean data	<Line>.Mean_data
<Line>.Alerts.Nr of alarms	<Line>.Alerts.Nr_of_alarms
<Line>.Alerts.Nr of warnings	<Line>.Alerts.Nr_of_warnings
<Line>.Alerts.Alerts	<Line>.Alerts.Alerts
Line names	Line_names
Line to head	Line_to_head
Status	Status

Security

Supported security policies

Sensoft supports the following security policies:

<table>
<tr><th></th><th>Sensoft as Server</th><th>Sensoft as Client</th></tr>
<tr><td>Without Username or Password</td><td> 
</td><td> 
</td></tr>
<tr><td>With Username and Password</td><td> 

Important: Username and Password are not verified, since NI does not support this.</td><td> 
</td></tr>
</table>

Legacy security policies (Basic128Rsa15 and Basic256) can be used instead of the current ones, by setting a checkmark near the advanced setting **Use legacy security policy** (Figure 2).

If you need to limit access to the OPC UA server, we recommend to use the Windows firewall to limit the access to port 4840, which is used by the OPC UA server (unless

another port is set in field **Port**).

Server certificate files

As of now, no certificate files are used by Sensoft neither as client nor as server. We can implement it if our partners need it.

Changes

V 1.0.8 (Sensoft Multiline 2.4 and later)

- Chapter Variables.From Sensoft: Added variable **Status**
- Variables of type Array: Made most of them work for PLC clients with fixed array size and updated some of their descriptions

V 1.0.7 (Sensoft Multiline 2.3)

- Chapter Variables.From Sensoft: Added variable **Line to head**

V 1.0.6 (Sensoft Multiline 2.2)

- Added Chapter Advanced settings and Figure 2
- Added Subchapter IEC 61131-3 names
- Chapter Security: Updated Security policies.
- Chapter Variables.To Sensoft, variable Switch to tab: Added new tabs

V 1.0.5 (Sensoft Multiline 2.1)

- Chapter Variables.To Sensoft: Added variable **<Line>.Next.Diameters [um]** and marked variables <Line>.Next.Diameter [um] and <Line>.Next.Diameter (y-axis) [um] as legacy
- Security - Supported security policies: Added a note that Sensoft as OPC UA server does not use Username and Password

V 1.0.4 (Sensoft Multiline 1.2.3 to 2.0)

- Chapter Variables.To Sensoft, variable <Line>.Custom sensors.<Index>.Fault: Added that max. writing speed is 10 Hz and added the possibility to send multiple faults in one command.
- Chapter Performance and capping: During overload, Sensoft now writes the 10 largest faults per cycle to OPC UA instead of the first 10 faults. Added that max. writing speed is 10 Hz.

V 1.0.3 (Sensoft Multiline 1.2.0 to 1.2.2)

- Chapter Variables.To Sensoft: The values "Lump", "Neck-down" and "Position" of **<Line>.Custom sensors.<Index>.Settings.Name** are now valid

V 1.0.2 (Sensoft Multiline 1.0.2 to 1.1.4)

- Chapter Variables.To Sensoft: Implemented **<Line>.Stop and discard data**

V 1.0 (Sensoft Multiline 1.0.0 to 1.0.1)

- The variables To Sensoft are now consistent with the local variables in Sensoft, i.e. can also be read, with the restrictions specified at the end of Chapter Variables - To Sensoft
- Chapter Variables.To Sensoft: Added **<Line>.Next.Diameter (y-axis) [um]** and **<Line>.Settings.Diam x = Diam y**
- Chapter Variables.From Sensoft: Added paragraph after table on subscription to an OPC UA variable

V 0.9.8

- Meaning of **<Line>**: Changed from line index (a number) to line name (a string). The reason is that now the nodes do not change when another line is deactivated.
- Chapter Variables.From Sensoft: Added **Line names**, removed **Number of lines** and **<Line>.Line name**
- Chapter Variables.From Sensoft: **Signal [%]**: Now updated also if <Line>.Measuring is False
- Chapter Variables.From Sensoft: **<Line>.Last fault.Length [m]**: Now implemented, for all fault types but Lump and Neck-down
- Chapter Variables.To Sensoft: **Switch to Line** was changed from Int32 to String. It accepts both Line name, or Line index as string

V 0.9.5

- Chapter Variables.To Sensoft: Added **<Line>.Custom sensors**, removed **<Line>.Absolute diameter [um]**
- Chapter Variables.From Sensoft: Added **<Line>.File path** and **<Line>.Mean data**
- Chapter Variables.From Sensoft: Unit of **<Line>.Last fault.Length [m]** changed from mm to m