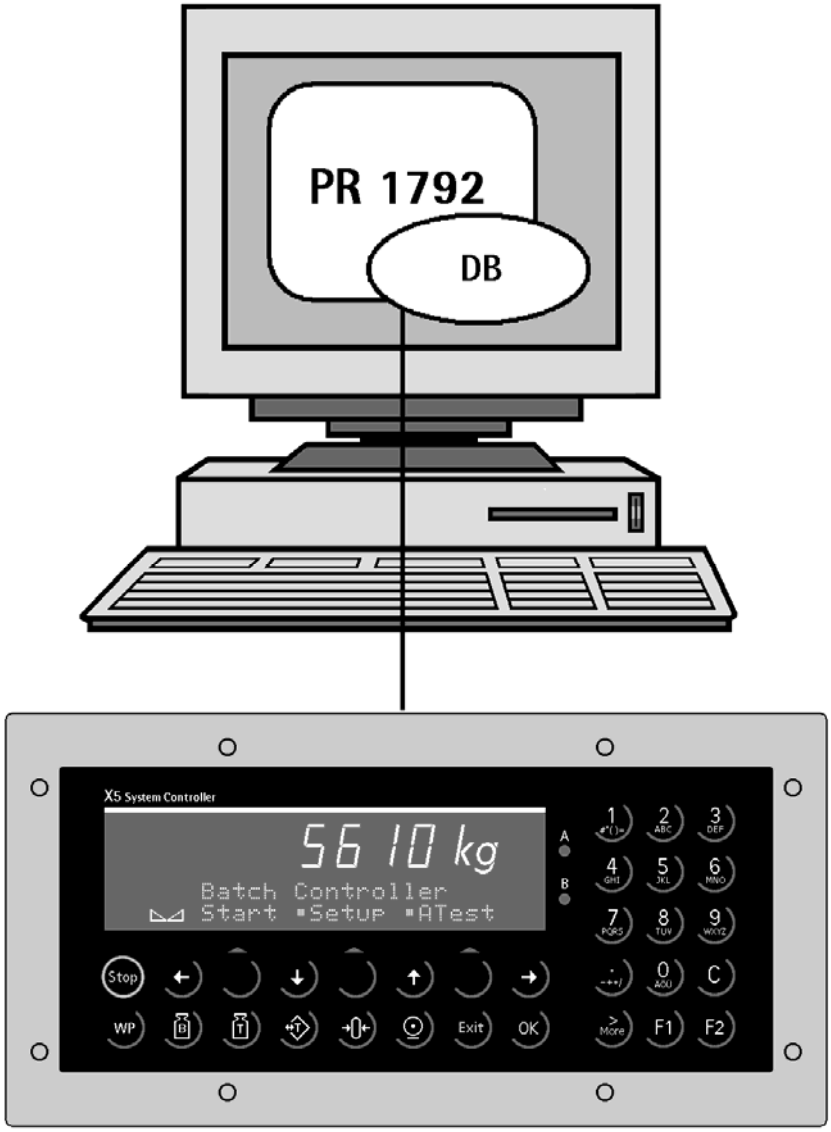
OPC Server

PR 1792/00

Operating Manual



Operating Manual for PR 1792	9499 050 79202 Rel. 2.2.0.14	Edition 2	11.07.2011

Sartorius Mechatronics T&H GmbH, Meiendorfer Str. 205, 22145 Hamburg, Germany
Tel:+49.40.67960.303 Fax:+49.40.67960.383

Please note
Any information in this document is subject to change without notice and does not represent a commitment on the part of SARTORIUS. This product should be operated only by trained and qualified personnel. In correspondence concerning this product the type, name and release number as well as all license numbers in relation to the product have to be quoted.

Important

Bitte beachten
Alle Angaben in diesem Dokument sind unverbindlich für SARTORIUS und stehen unter Änderungsvorbehalt. Die Bedienung des Produktes darf nur von geschultem, fach- und sachkundigem Personal durchgeführt werden. Bei Schriftwechsel über dieses Produkt bitte Typ, Bezeichnung und Versionsnummer sowie alle mit dem Produkt in Zusammenhang stehenden Lizenznummern angeben.

Wichtig

Contents

1 Introduction

OPC Server PR 1792 is part of a system for transmission of current data between Sartorius instruments and PC programs which can operate as an OPC client. The system comprises the connected weighing- / batching Instruments, a PC with programs and the connection of all components. The platform for the system is the PC under operating system Windows NT, Windows 2000, Windows XP or Windows 7. For some system components, separate manuals are available to which only reference is made in this documentation.

The OPC server permits access to the data of instruments connected to the PC by OPC client application. Depending on the instrument the connection can be done via serial interface and/or via Ethernet. With instruments provided with serial connection to the PC, protocol EWCOM V3 is used. Due to differences related to application, communication and memory repartition, transmitters/indicators and batching systems are discussed separately in this manual, wherever necessary.

1.1 Brands, trade names

Windows NT, Windows 2000, Windows XP, Windows 7 are registered trade names of Microsoft Corporation. All other brands or product names are brands or registered trade names of the relevant manufacturer or holder.

1.2 Exemption from liability

The PR 1792 program is a state-of-the-art development. No warranty is taken for correctness, in particular, in connection with third-party software and hardware components required for program operation. Liability for damage due to other system components or faulty handling of this program is precluded by the manufacturer. The use of the program implies recognition of the above-mentioned stipulations.

1.3 Product description

1.3.1 Data carrier

PR 1792 is delivered on the X5 Power Tools CD.

Product PR 1792 comprises the data carrier with documentation in PDF format (please, use Adobe Acrobat 5.0 or a higher version) and installation programs on the CD.

The set-up program for the installation and documentation is given in the PR 1792 directory.

For the latest information and hints on error corrections, see file history.wri. The exact version number is given in file PR 1792.

1.3.2 Program package survey

The PR 1792 program package comprises:

PR 1792.exe	OPC server
EwDrv_01.exe	Communication driver
OPC_Client.exe	OPC client example for testing and debugging
pr1792.chm	Help file
REGALL.BAT	Unless registration of the OPC servers in the Windows Registry was successful, it can be repeated hereby without reinstalling PR 1792.
UNREG.BAT	Deletion of OPC server registration from the Windows Registry
OPCInst.doc	Hints for registration

Additionally, some common DLLs of the OPC foundation are included.

1.3.3 License information

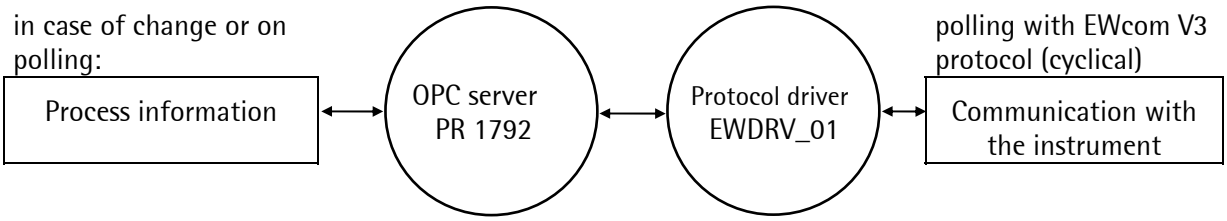
All parts of the PR 1792 program package are copyright-protected and must not be used for commercial purposes without license. For information interchange with the instrument by means of the OPC server, a license is required. A license and a license certificate belong to every instrument. The license is related to the board number.

Instrument	OPC Server	OPC Database access
PR 1713, X5 (PR 5610), X6 (PR 5710)	PR 1792/13	PR 1792/20
X4 (PR 5510)	PR 1792/13	PR 1792/20
Combics Pro	OPC-Server license	OPC-Server license
X3 (PR 5410)	No license required	Not possible
PR 5220	No license required	Not possible
PR 1710, PR 1711	PR 1792/10 *	Not possible
PR 1720	PR 1792/10 *	Not possible

* PR 1792/10 to be entered at [License number] in the mask [Group Configuration] in PR 1792-Program (only PR 1710, PR 1711 and PR 1720).

1.3.4 Operating principle

Data communication to the instrument is built up only, provided that the license is valid and the instrument was configured correctly (see instrument documentation). Unless a license number was entered, or if an invalid license was specified, only access to some items is possible (for DisplayIt).



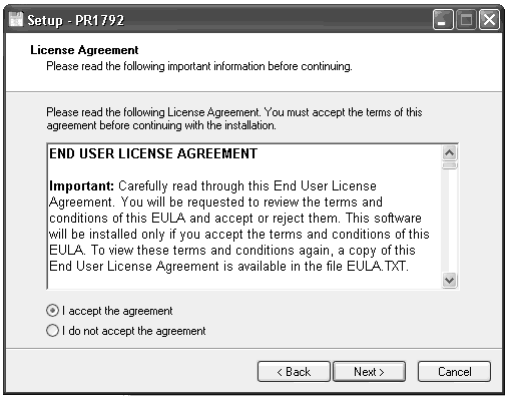
An instrument is addressed by the EW protocol driver EWDRV_01 via a serial PC interface or via Ethernet. Data communication (polling) is automatic by PR 1792. Plausibility checking of software configuration and actually connected instrument is provided, whereby localization of errors (such as confusion of connectors at the PC interface) is possible.

OPC server PR 1792 moves the data in blocks of 64 bytes. Data throughput is increased, if you group the data in blocks of 64 bytes, e.g. 64 – 127, so that all inquiries with md, mx, mr, ... are transmitted in a block.

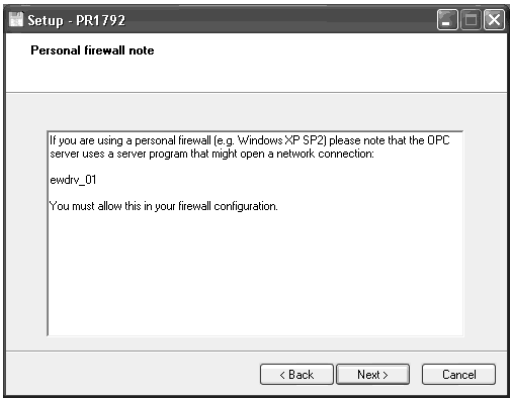
2 PR 1792 OPC server installation

For installation, you must be logged in with administrator authority. If necessary, consult your system administrator. To prevent conflicts of access to system files during installation, **close all other Windows applications** – as usual during installation under Windows.
For OPC server installation, insert the supplied PowerTools CD-ROM into the drive of your PC and start (unless this is done automatically by the system) program **index.html**.
Select the folder of the PR 1792 installation and start **PR1792 2.2.0.14. setup.exe**.

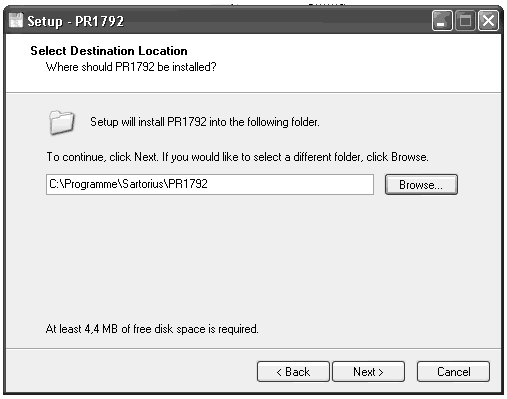
For installation click [Next]



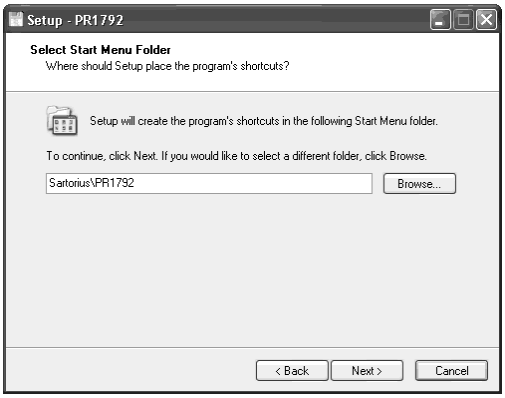
Read the license agreement and activate [I accept the agreement], click [Next]



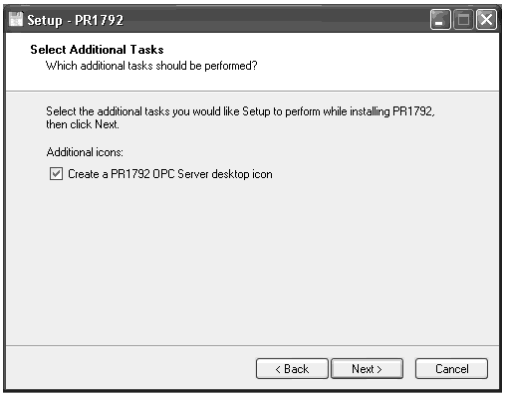
If an Ethernet connection to the instrument is used, the PC operating system must allow the network connection. Firewalls have to be switched off, respectively the transmission of the connected instrument has to be allowed.
Continue with [Next].



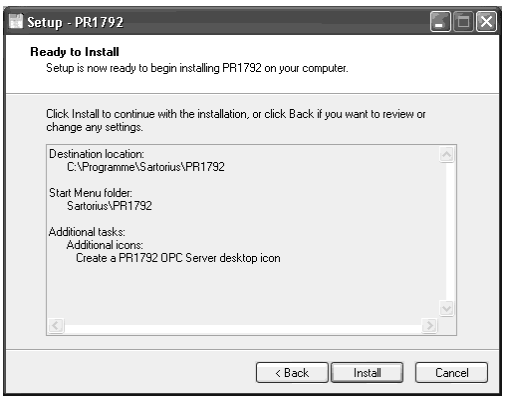
The target directory for the installation can be changed.
We recommend using the proposed directory.
Unless the installation directory exists already, it is created by the installation program.
Continue with [Next].



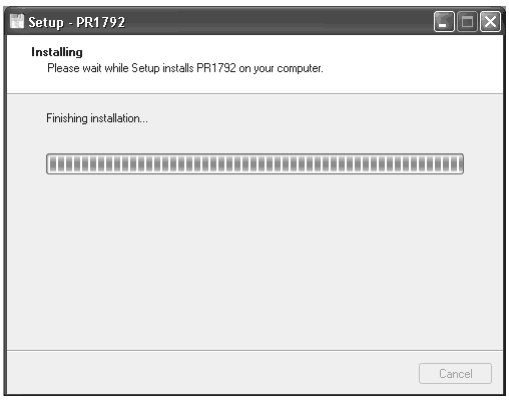
Define the 'shortcut' for the Windows Start-Menu, continue with [Next].



When activated, an icon will be created on the desktop for easy starting of the OPC-Server, continue with [Next].

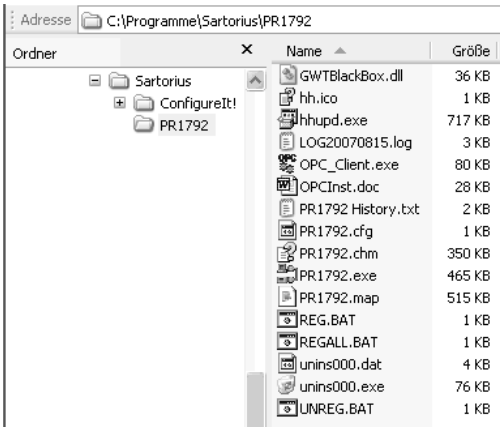


Information about the entered data, continue with [Install].



Installation in progress

Finishing the installation with [Finish]



After installation the program elements are filed in the selected directory.

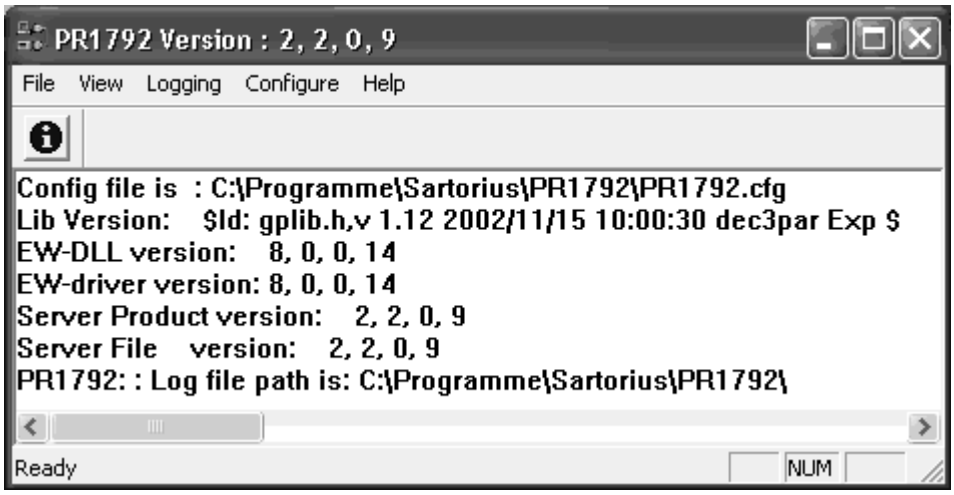
The icon to start the OPC-Server is provided on the desktop.
The OPC-Server can be started directly by the user, an OPC-Client can start the OPC-Server too.

After starting the OPC-Server the symbol will appear in the task bar.

3 Configuration of the OPC server

3.1 Survey

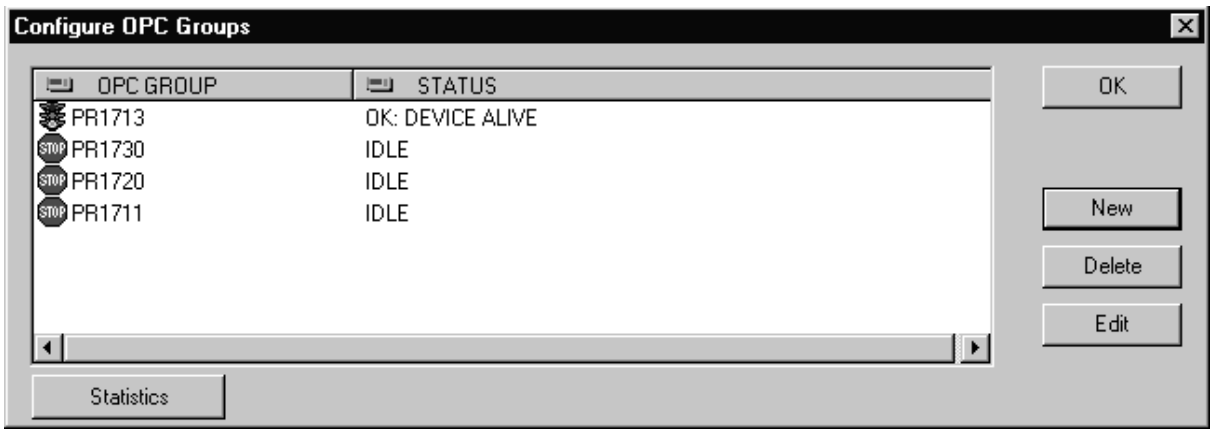
In conjunction with the installation, the OPC server must be configured.
When starting program 'PR 1792 OPC Server', the EWDRV_01.EXE communication process is started, unless this was already done. The communication process starts only once. If it is already running, the start is canceled. The communication process is minimized into an icon in the task bar and started.
PR 1792 starts always with a window in the size used last. Both programs can be terminated as usual under Windows.



File	[File]–[Exit] terminates the server program.
View	[Toolbar] and [Status Bar] can be activated separately.
Logging	For reporting various trouble shooting messages. See chapter 'Options for trouble shooting'.
Configure	Used for OPC server configuration.
Help	OPC server online help and version information

3.2 Configuration menu

Click on ⓘ or select [Configure]-[Show/Edit Group Status] to open the configuration menu:

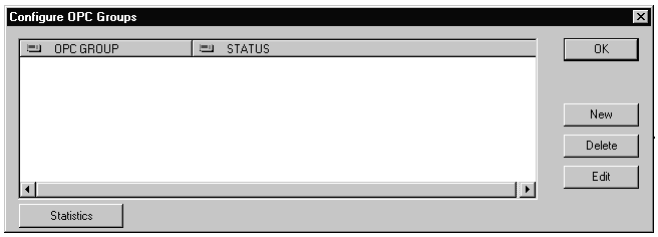


All 'OPC Groups' with the current status are listed in this window.
The three possible statuses are:
• IDLE – The 'Group' is not in operation and not connected with the instrument. This is normal unless an OPC client program is connected with the server. The group can be edited or deleted only in this status.

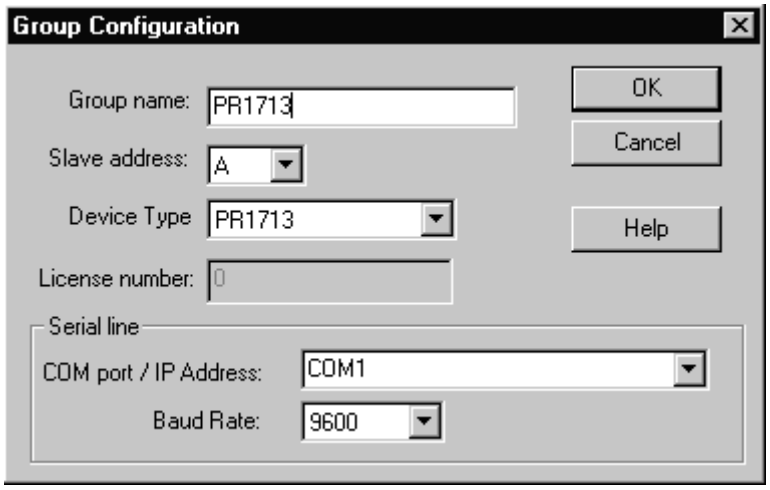
- OPENING / INITIALIZING – an attempt to connect the group with the instrument is made.
- OK: DEVICE ALIVE – The 'Group' is connected with the instrument and active.

3.2.1 Creating a new 'Group'

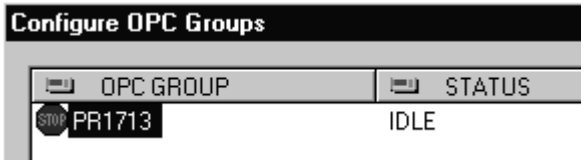
Start by defining all instruments with which the OPC server communicates. Each instrument represents a group.
Click on ⓘ or select [Configure]-[Show/Edit Group Status].



This menu permits creation, editing or deletion of 'groups', click on [New].



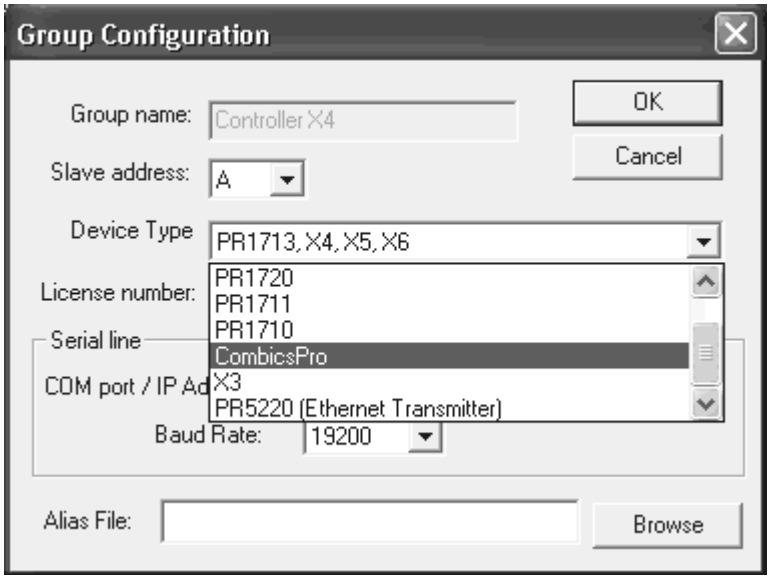
Group name	Free name, which represents an individual instrument. (e.g. Indicator X3)
Slave address	Only for serial communication link. Each instrument on a serial line (e.g. RS485 – not RS232) requires a unique address (A, B, C...), which must correspond to the address configured on the instrument.
Device Type	Select the instrument from a list, see chapter 3.2.2
License number	Enter the PR 1792 license number, only for PR 1710, PR 1711 and PR 1720.
COM port /	PC-COM-Port where the serial line is connected.
IP Address	IP-Adresse des angeschlossen Gerätes (Netzwerkparameter)
Baud Rate	Required for serial communication only, it must correspond with the address configured in the instrument.



After click on [OK] the instrument is configured. As it is not addressed by an OPC client program so far, it is in status 'Idle'.

3.2.2 Device Type

The instrument to be connected is selected from a list:



Depending on the instrument, a serial and/or Ethernet connection can be used:

Instrument	Serial RS-232	Serial RS-485	Ethernet
PR 1713, X5 (PR 5610), X6 (PR 5710)	Standard	Option 1713/04	Option PR 1713/14
X4 (PR 5510)	Standard	Option 5510/04	Option PR 5510/14
Combics Pro	Standard	Option 5510/04	Standard
X3 (PR 5410)	Not possible	Not possible	Standard
PR 5220	Not possible	Not possible	Standard
PR 1720	Option PR 1602	Option PR 1604	Not possible
PR 1710/02, /12	Standard	Not possible	Not possible
PR 1710/04, /14	Not possible	Standard	Not possible
PR 1711/62	Standard	Not possible	Not possible
PR 1711/64	Not possible	Standard	Not possible

3.2.3 Serial interface

The instruments must be equipped with a suitable serial interface, depending on the instrument eventually an option card has to be plugged-in (see list in chapter 3.2.2.). Select protocol EWCOM V3 (ASYCOM slave). Configure identical interface parameters (e.g. Ewcom V3, 9600 Baud, 7 bits, even parity, ...), as described in the relevant instrument documentation.

3.2.4 Ethernet

Some instruments (see list in chapter 3.2.2) can be connected via Ethernet TCP/IP.
Enter the network address in the mask at [COM port / IP Address] (e.g. **\\130.143.1.5**, see Manual for the respective instrument / Ethernet-card). The address is defined by the local network administrator or if DHCP is used, automatically given from the Server. Take care, that double definition of IP addresses does not happen!

3.2.5 Editing

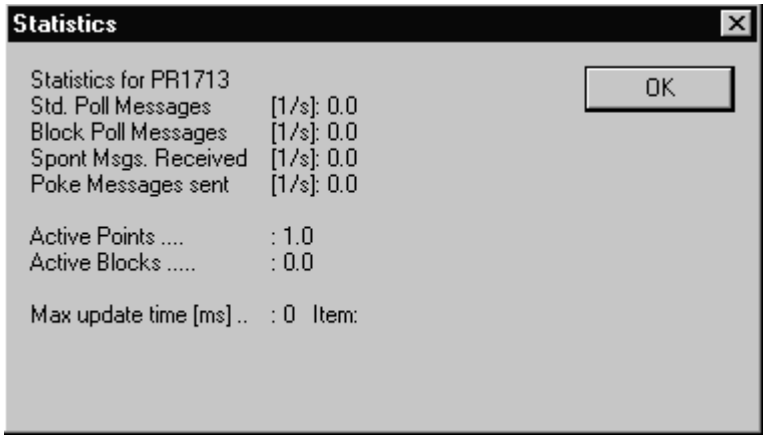
Click on ⓘ or select [Configure]-[Show/Edit Group Status].
For this, the 'Group' status must be 'IDLE'.
Select an instrument and click on [Edit].
In this menu, the parameters can be changed as described in the previous chapter.

3.2.6 Deleting

Click on ⓘ or select [Configure]-[Show/Edit Group Status].
For this, the 'Group' status must be 'IDLE'.
Select an instrument and click on [Delete].
The instrument is deleted.

3.3 Statistics

For diagnosis purposes, the actual data volume per second can be displayed.
Click on ⓘ or select [Configure]-[Show/Edit Group Status] for access to the configuration menu.
Select a 'Group' and click on [Statistics].



As long as this window remains open, all data are updated continuously.

There are four different groups of commands:
'Std. Poll Messages', **'Block Poll Messages'**, **'Spont Msgs. Received'** and **'Poke Messages sent'**.
The average number of commands per second is given with each value. The total of all four values is the overall number of commands per seconds.

'Std. Poll Messages': number of commands and replies of type 'mbxxxlxx' (reads a string)
'Block Poll Messages': number of block transmission orders.
'Spont Msgs. Received': number of spontaneous messages. E.g. value ST_WGT_A is sent at every weight change.
'Poke Messages sent': number of write commands.

'Active Point': number of active 'Items' which were connected by the OPC client.
'Active Blocks': number of active blocks. For efficiency reasons, the 'mx', 'mr' and 'md' commands are not transmitted individually. Data blocks of 64 bytes (512 bits) are transmitted instead.

All data belonging to one of the data blocks are transmitted in a telegram. Detection and management of the data affiliation to a block are by the OPC server.

64-byte data blocks
Bit 1024 ... bit 1535
Bit 1536 ... bit 2047
Bit 2048 ...
...

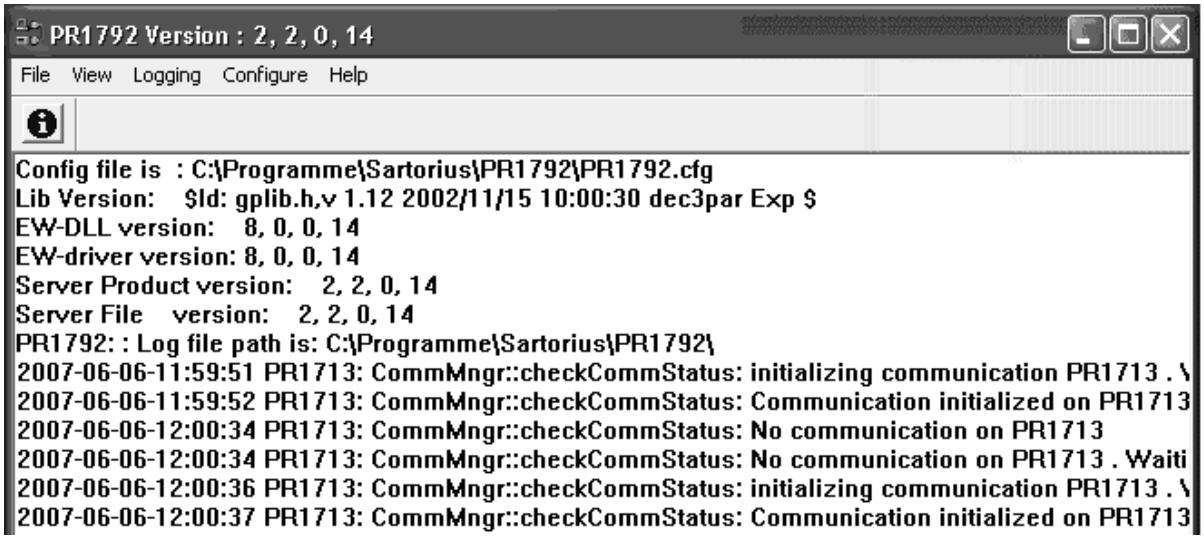
For this reason, compactness of memory locations is purposeful. If data are distributed over the complete SPM memory, a large number of data blocks with few useful contents must be transmitted. Therefore, the number of active blocks must be as low as possible.

'Max update time [ms]': This is the maximum time required by an 'Item' for data updating. This time increases with the number of 'Items' and 'Groups', because the data communication increases, i.e. the time remaining for each individual 'Item' decreases.
Priority over other data transmissions is given to write commands. Very intensive data writing by the client program can impair the data read rate.

3.4 'Logging' menu – options for trouble shooting

3.4.1 Survey

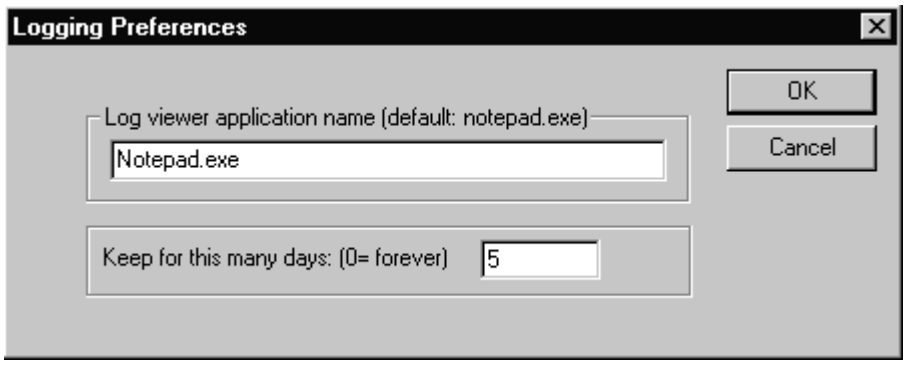
During OPC server operation, various error messages are possible. These are displayed in the OPC server main window:



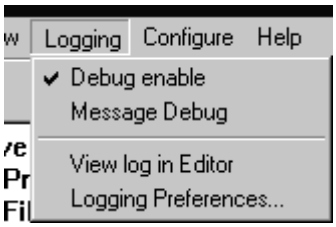
Every day, the OPC server creates a log file with the date of creation in the file name in format: LOGyearMonthDay.log. e.g.. "LOG20070815.log".
The current log file can be opened via menu [Logging]-[View log in Editor] in an editor.

3.4.2 Logging settings

Adjust the editor to open the log file for logging. The default setting is "Notepad.exe".
Moreover, the time during which log files have to be kept can be adjusted. With "Keep for this many days:" set to '0', all log files are kept for an unlimited period of time.
The OPC server uses the file name and not the date as criterion for deleting.
[Logging]-[Logging Preferences]

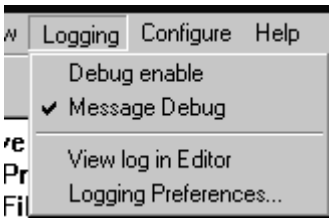


3.4.3 Debug enable



This option activates the debug mode: in addition to the other stored error messages, further information on internal sequences is generated in the OPC server. This mode is intended for diagnosis purposes. During normal operation, it should not be activated, because a large quantity of data consumes a lot of memory space on the hard disk and computer capacity.

3.4.4 Message Debug



For extended diagnosis purposes. messages which extend partly to communication level are generated. The data quantity is extremely high, which is a considerable load for the overall computer.

WARNING: Please, use this mode only for diagnosis purposes!

4 Operation with an OPC client application

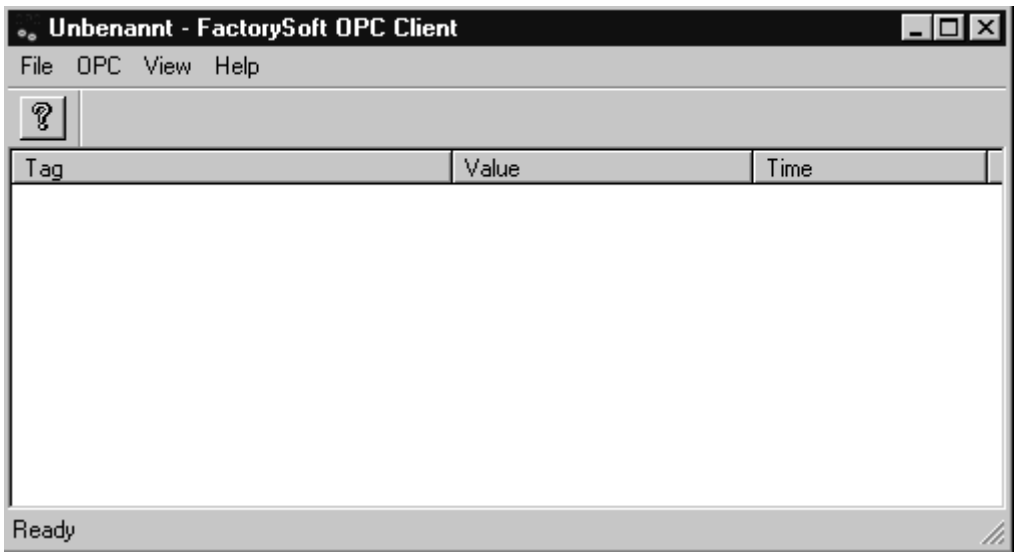
4.1 General

The PR 1792 OPC server is delivered with a program example of an OPC client:
OPC_Client.exe

The description of an OPC client is based on the following program. This application example can be used for testing the OPC server functions and finding errors. In case of problems with a client program written by yourself, e.g. a Visual Basic Program, check first, if the relevant functions can be tested with the OPC client.

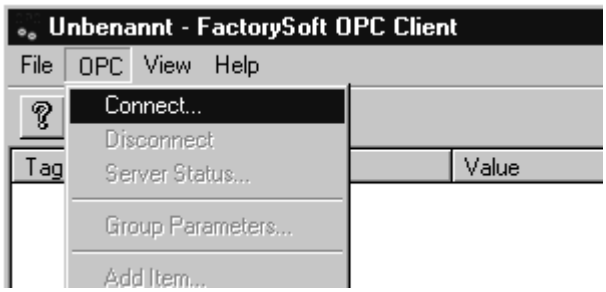
4.2 Starting OPC client

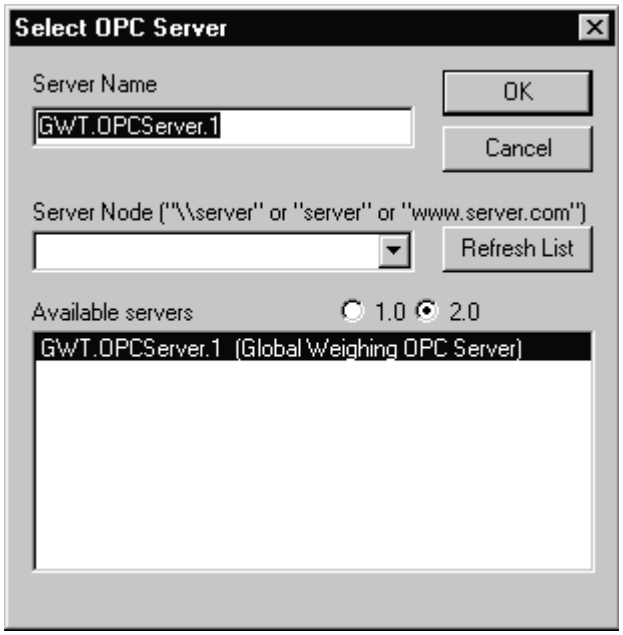
Start program OPC_Client.exe.



4.3 Connecting client and server

Now, connect the client to the server.
[OPC]-[Connect]





Server Name:
The selected server name must be specified in this field.

Server Node:
A server node must be specified only, if the server runs on another computer.

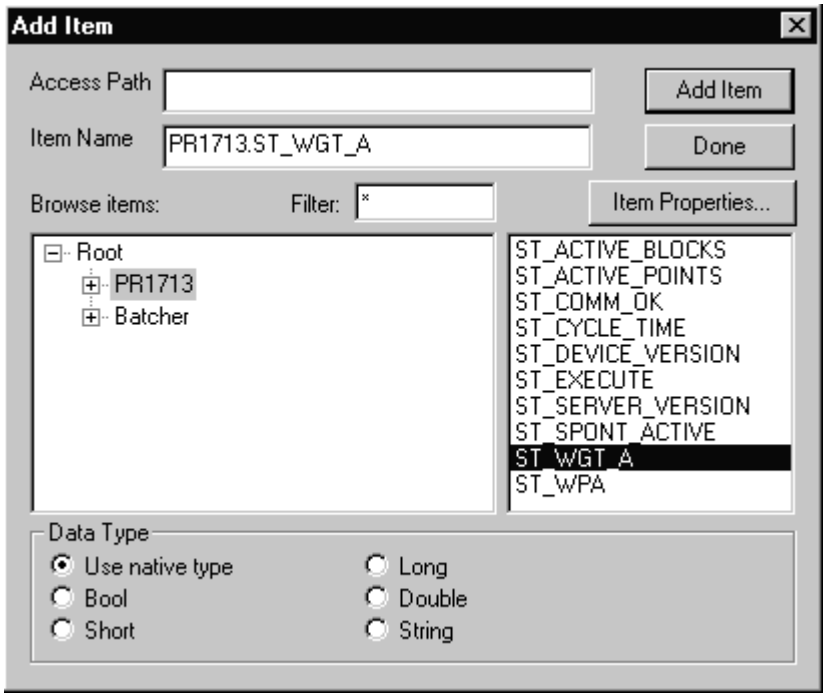
Available servers:
The list of all available servers.

Unless an error message is displayed after pressing "OK", the client is connected with the server.

4.4 Adding items

Now, data from all instruments connected to the OPC server can be read and written.
In this example, two instruments are made known to the OPC server: "PR 1713" and "Batcher".

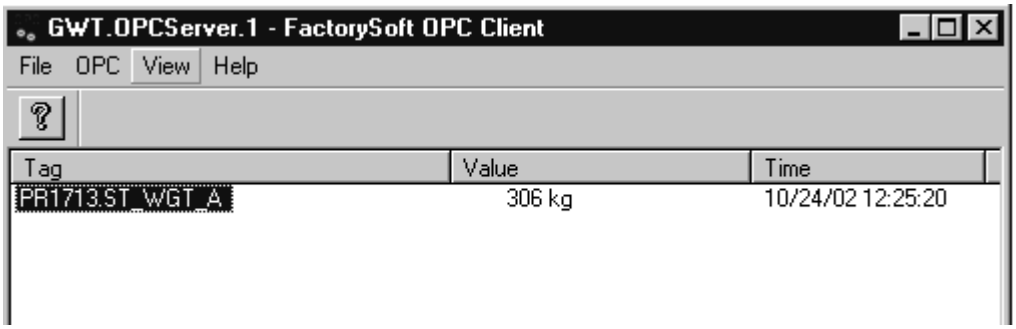
[OPC]-[Add Item]



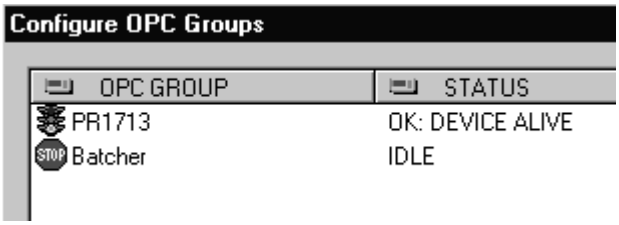
You should always activate 'Use native type'.

A complete 'Item' name is a combination of the 'group' name, followed by a dot '.' and the 'Item' name, e.g. PR 1713.ST_WGT_A.
Click on an instrument ('group') first and then on one of the logic 'Items'. Store this 'Item' in the list by clicking on [Add Item].
Click on [Done] to complete adding of new 'Items'.

Now, the weight value from an instrument is read and continuously updated. The time stamp shows date and time of the read value.



The OPC server status changes from 'IDLE' into 'OK: DEVICE ALIVE' and the traffic light changes to green.
Click on the 'Info' button or select [Configure]-[Show/Edit Group Status]

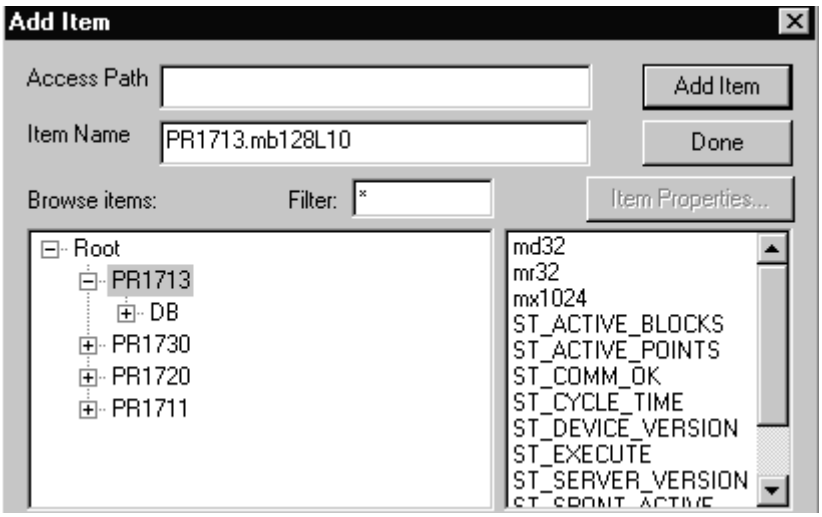


In this example, data communication was built up with OPC group 'PR 1713' and not with OPC group 'Batcher'.

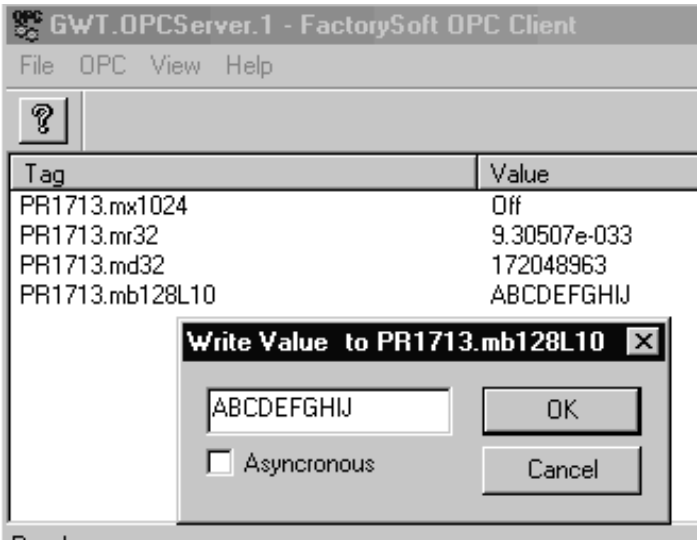
OPC server PR 1792 supports OPC clients and provides a data channel between the instrument specified with the *group name* and the application. Start the relevant application. The variables are read or written in the application (e.g. marker bit).

4.5 Read/write SPM memory area (standard variables)

Standard variables can be used for access to the SPM ('Scratch Pad Memory') memory area.

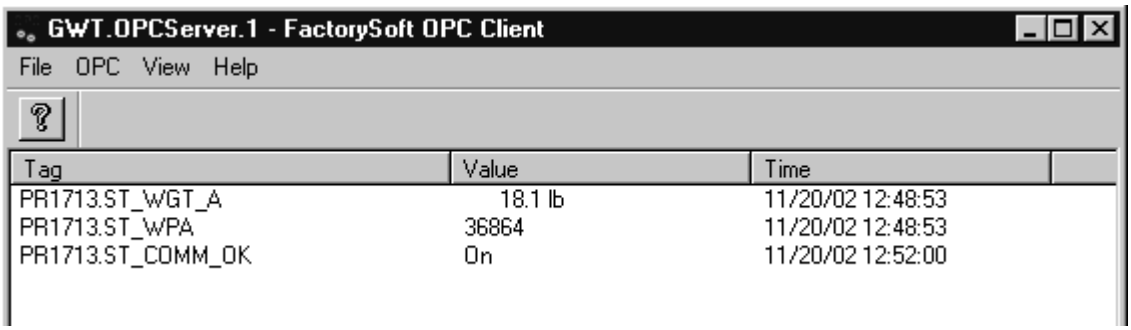


Select an 'Item' (e.g. PR1713.mb128L10) and click with the right mouse key.
With e.g. 'Write Value', new data can be sent to PR 1713.



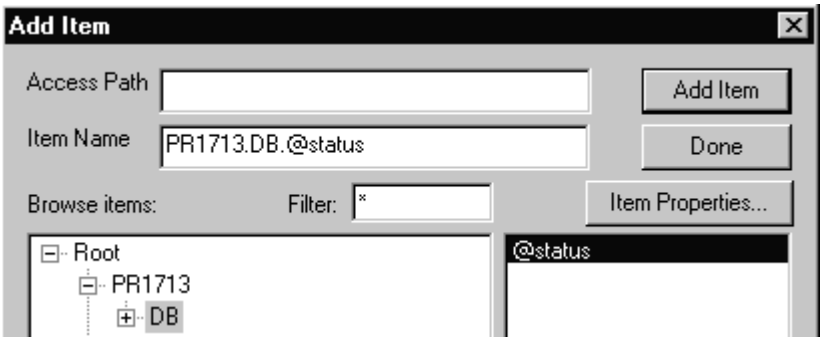
4.6 Reading weights (system variables)

System variable ST_WGT_A contains the actual gross weight.
The status of weighing point A is given in ST_WPA..

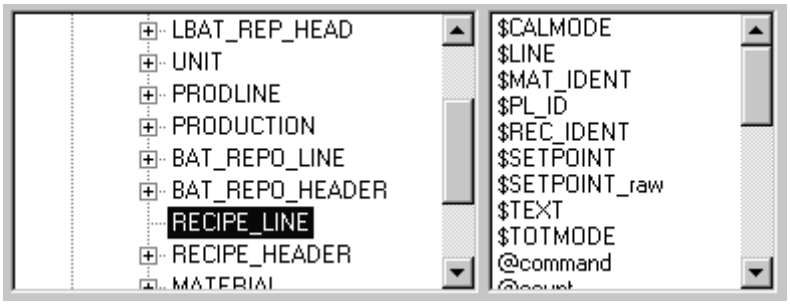


4.7 Reading a table on a PR 1713

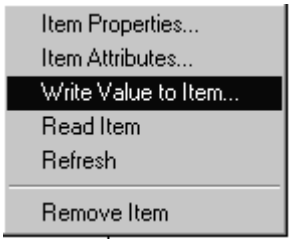
The following example shows how an access to a table in PR 1713 can be made. For this, licence PR 1792/20 is required additionally.
Click on the PR 1713 'DB' sub-group and add 'Item' @status [Add Item].
The server loads the structure of all tables from the instruments.



With 'Item' @status 'OK', loading is terminated.
Open sub-group 'DB' and select recipe line table 'RECIPE_LINE'.



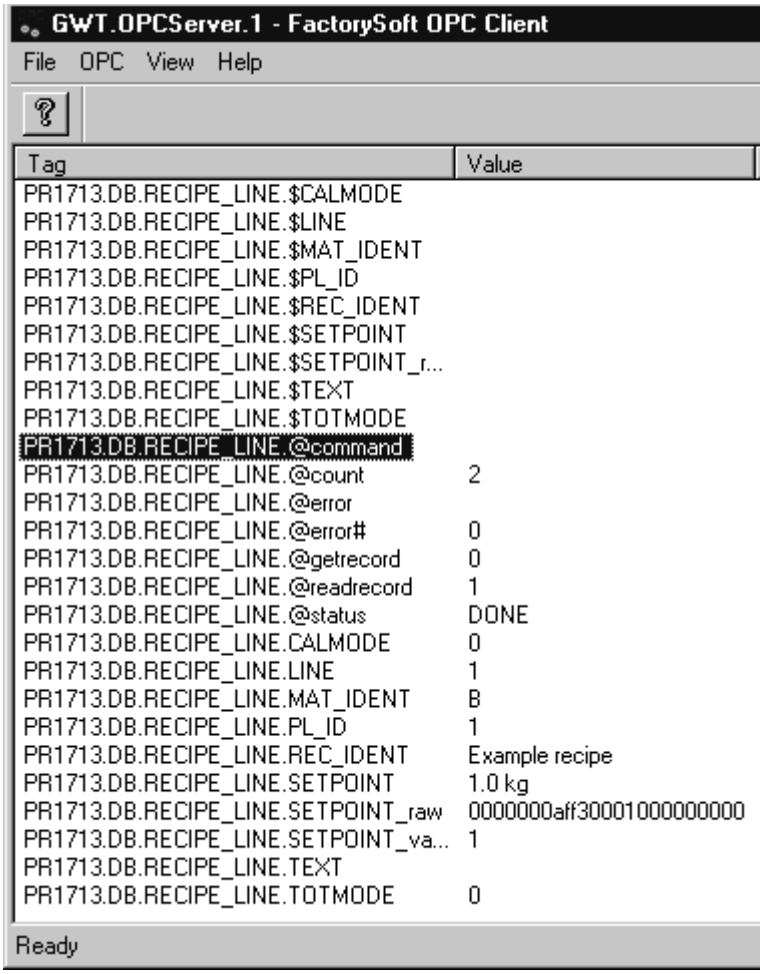
Add all 'Items' of this table. [Add Item] or double-click.
Write 'SELECT' into 'Item' @command:
Click with the right mouse key on @command and select 'Write Value to Item...'.



As the $ fields were left empty, all data sets (recipe lines in this example) are loaded from the instrument into a buffer of PR 1792. The first data set found is shown, i.e. the first recipe line in this example: Line=1.
@status is set to DONE.
The number of all data sets is 2: 'Item' @count = 2.

To complete polling and to set the server ready for another selection, command 'RESET' must be written into @command, whereby @status goes to 'OK'.

For loading line 2 into the buffer purposefully, write 2 into 'Item' $LINE and restart polling.
$-fields can contain search criteria with comparison operators (<,>,=,<=,>=,<> and *) .
They don't contain table values.
If the data set is available, 'Item' LINE (a table element) is also 2.
The number of all data sets with this search criterion is 1: 'Item' @count = 1.
Unless the data set exists, an error message is displayed.



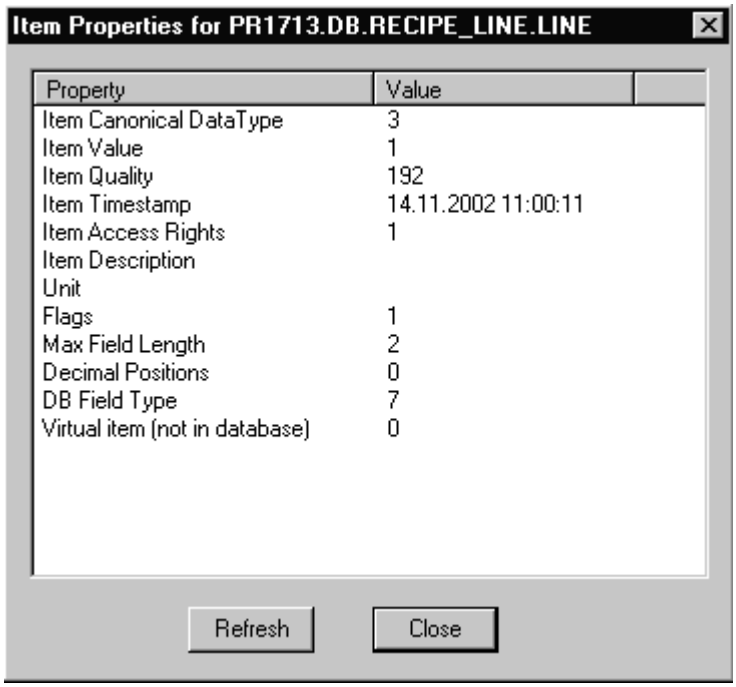
4.8 Item Properties

Additional information can be polled from every 'Item'. For this purpose, select an 'Item' from the list and click on 'Item Properties' in the menu with the right mouse key.

With 'Items' from the database table, parameter **'Flags'** indicates, if this 'Item' is (1) or is not (0) a database key field.
'Virtual item (not in database)' indicates, if this 'Item' is a database element (0) or an artificially generated 'Item'. In the example of the recipe line table, 'Item' PR1713.DB.RECIPE_LINE.SETPOINT is the recipe setpoint and a real element. 'Item' PR1713.DB.RECIPE_LINE.SETPOINT_raw is the artificially added raw value.
'Max Field Length' defines the item length in bytes. With datatype STRING, the maximum text length plus the length byte is specified. I.e. maximum length+1. If the maximum length of the string is 21, an internal variable with length 20 is read out.
'Item Canonical Data Type' indicates the VARIANT data types and
'DB Field Type' indicates the type of database fields.
For further information on OPC interfaces and programming interfaces, see http://www.opcfoundation.org.

Data type	Code
BOOL	1
BYTE	2
WORD	3
DWORD	4
LWORD	5
USINT	6
UINT	7
UDINT	8
ULINT	9
SINT	10
INT	11
DINT	12
LINT	13

Data type	Code
REAL	14
LREAL	15
TIME	16
DATE	17
TIME_OF_DAY	18
DATE_AND_TIME	19
STRING	20
WEIGHT	21



Example: RECIPE_LINE.LINE

Flags = 1: key field

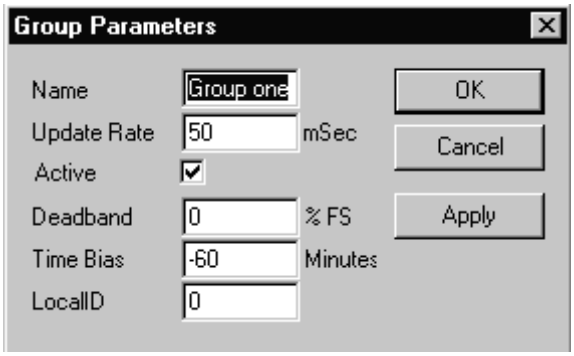
Max field length = 2:
Length of integer data type

Virtual item (not in database) = 0: real element

4.9 Group Parameters

Various group parameters can be defined.
In this client application example, changing the default values is not necessary.

[OPC]-[Group Parameters...]



Please, note that the update rate affects only the rate between Client and OPC server and not the update rate between instrument and OPC server.

5 OPC service syntax

```
<Node>.<Group>.<Item>
```

Node is the name of the computer in the network. If the OPC server runs on the same computer, the node name need not be specified.

Group *Group Name* is the name of an instrument in the system and must be specified in the OPC client in compliance with the definition in OPC server PR 1792/00. Example for a group name: PR 1720_A.

Item *Standard Item* is the name of the variable itself. Dependent of the variable function, name conventions can be defined. Names beginning with ST_ are predefined by the OPC server for system services and cannot be defined freely.

5.1 Variables (Items, Standard Item Names)

There are specific 'Item' names to be used by OPC server PR 1792:

SPM Item names. SPM means 'Scratch Pad Memory' (PLC memory area in the instrument). SPM data can be read and written with SPM 'Items'.

System variables special variables

Database variables access to instrument databases

Controllers support the two classes, transmitters / indicators support only the SPM 'Items'.

Commands to controllers and transmitters / indicators are different in structure and SPM address memory area (1024_{dec} bytes or only 128_{dec} bytes with transmitters / indicators). With transmitters / indicators data definition in the SPM address area is fixed (see instrument manual).

With transmitters / indicators please note:

- It may be necessary to adapt the address information specified in the PR 1792 help file to the 128-byte SPM address area, the most significant address is 27_{dec}.
- Only SPM items (access to the SPM) are permitted. An OPC polling request for *Items* which are not permitted is rejected, a corresponding message is stored in the log file.
- Spontaneous telegrams are not provided.

5.2 Standard variables (SPM Item names)

Caution: The SPM size is 128 bytes with transmitters / indicators and 1024 bytes with controllers. All addresses point to the above mentioned memory area, e.g. bit address mx992 points to the same memory location as md31 (mx992 is actually the most significant bit of md31).

Please, refer to the instrument (system) documentation for exact memory allocation. The SPM area of the instruments is used e.g. for communication between instrument application and external instruments. The OPC server can be used to read and to write data in this area from a Windows PC.

mx<bit address>

Type: discrete (bit)
The most significant bit is dependent of the instrument (see instrument manual):
Controllers: mx8191
Transmitters / indicators: mx1023
Length: 1 bit

mr<double word address>

Type: real
Can be used only for controllers:
Controllers: mr255
Length: 4 bytes

md<double-word-address>

Type: integer
The most significant double word address is dependent of the instrument (see instrument manual):
Controllers: md255
Transmitters / indicators: md31
Length: 4 bytes

mb<byte-address>L<length>

Type: string (OPC message)
Can be used only with controllers:
Controllers: mb1022L.. (minus string length)
for example: mb1000L20, mb100L10
Length: string length in bytes + 1 (byte address)
The first byte of an SPM string describes the length of a string. The length byte is NOT transmitted in the OPC message. When poking e.g. "AB" to mb128L10 , 02 65 $66_{(hex)}$ is written into byte address 128 of the SPM memory.

Note:

For SPM memory area addressing, various methods are possible.
Bit address "mx" is used to address individual bits, e.g. mx512.

Byte address "mb" makes an access to byte addresses of the same memory. I.e. mb64 is used for access to eight bits (corresponding to bit addresses mx512... mx519).

Double word address md16 addresses the 16^{th} double word, bytes mb64... mb67 and bits mx512...mx543.

Addressing example

(Marker area reserved for the firmware **bytes 0 to 127.** Reserved bits are underlined. The rest is not reserved.)

LWort	DWort	Wort	Byte	Bit																Bemerkung
0	0	0	0,1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Enable
		1	2,3	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
	1	2	4,5	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	
		3	6,7	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	
1	2	4	8,9	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	Material
		5	10,11	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	outputs
	3	6	12,13	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	
		7	14,15	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	
2	4	8	16,17	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	WP A
		9	18,19	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	WP B
	5	10	20,21	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	WP C
		11	22,23	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	WP D
3	6	12	24,25	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	(WP E)
		13	26,27	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	(WP F)
	7	14	28,29	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	(WP G)
		15	30,31	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	(WP H)
4	8	16	32,33	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	Mirror WP A
		17	34,35	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	Mirror WP B
	9	18	36,37	288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303	Mirror WP C
		19	38,39	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	Mirror WP D
5	10	20	40,41	320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335	(WP E)
		21	42,43	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351	(WP F)
	11	22	44,45	352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	(WP G)
		23	46,47	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383	(WP H)
6	12	24	48,49	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	Recipe Contr
		25	50,51	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	Mirror RC
	13	26	52,53	416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	
		27	54,55	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447	
7	14	28	56,57	448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	(WP I)
		29	58,59	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479	(WP J)
	15	30	60,61	480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495	(mirror WP I)
		31	62,63	496	497	498	499	500	501	502	503	504	505	506	507	508	509	510	511	(mirror WP J)
8	16	32	64,65	512	513	514	515	516	517	518	519	520	521	522	523	524	525	526	527	WGA
		33	66,67	528	529	530	531	532	533	534	535	536	537	538	539	540	541	542	543	
	17	34	68,69	544	545	546	547	548	549	550	551	552	553	554	555	556	557	558	559	
		35	70,71	560	561	562	563	564	565	566	567	568	569	570	571	572	573	574	575	
9	18	36	72,73	576	577	578	579	580	581	582	583	584	585	586	587	588	589	590	591	WGB
		37	74,75	592	593	594	595	596	597	598	599	600	601	602	603	604	605	606	607	
	19	38	76,77	608	609	610	611	612	613	614	615	616	617	618	619	620	621	622	623	
		39	78,79	624	625	626	627	628	629	630	631	632	633	634	635	636	637	638	639	
10	20	40	80,81	640	641	642	643	644	645	646	647	648	649	650	651	652	653	654	655	WGC
		41	82,83	656	657	658	659	660	661	662	663	664	665	666	667	668	669	670	671	
	21	42	84,85	672	673	674	675	676	677	678	679	680	681	682	683	684	685	686	687	
		43	86,87	688	689	690	691	692	693	694	695	696	697	698	699	700	701	702	703	
11	22	44	88,89	704	705	706	707	708	709	710	711	712	713	714	715	716	717	718	719	WGD
		45	90,91	720	721	722	723	724	725	726	727	728	729	730	731	732	733	734	735	
	23	46	92,93	736	737	738	739	740	741	742	743	744	745	746	747	748	749	750	751	
		47	94,95	752	753	754	755	756	757	758	759	760	761	762	763	764	765	766	767	
12	24	48	96,97	768	769	770	771	772	773	774	775	776	777	778	779	780	781	782	783	(WP E)
		49	98,99	784	785	786	787	788	789	790	791	792	793	794	795	796	797	798	799	
	25	50	100,101	800	801	802	803	804	805	806	807	808	809	810	811	812	813	814	815	
		51	102,103	816	817	818	819	820	821	822	823	824	825	826	827	828	829	830	831	
13	26	52	104,105	832	833	834	835	836	837	838	839	840	841	842	843	844	845	846	847	(WP F)
		53	106,107	848	849	850	851	852	853	854	855	856	857	858	859	860	861	862	863	
	27	54	108,109	864	865	866	867	868	869	870	871	872	873	874	875	876	877	878	879	
		55	110,111	880	881	882	883	884	885	886	887	888	889	890	891	892	893	894	895	
14	28	56	112,113	896	897	898	899	900	901	902	903	904	905	906	907	908	909	910	911	(WP G)
		57	114,115	912	913	914	915	916	917	918	919	920	921	922	923	924	925	926	927	
	29	58	116,117	928	929	930	931	932	933	934	935	936	937	938	939	940	941	942	943	
		59	118,119	944	945	946	947	948	949	950	951	952	953	954	955	956	957	958	959	
15	30	60	120,121	960	961	962	963	964	965	966	967	968	969	970	971	972	973	974	975	(WP H)
		61	122,123	976	977	978	979	980	981	982	983	984	985	986	987	988	989	990	991	
	31	62	124,125	992	993	994	995	996	997	998	999	1000	1001	1002	1003	1004	1005	1006	1007	
		63	126,127	1008	1009	1010	1011	1012	1013	1014	1015	1016	1017	1018	1019	1020	1021	1022	1023	

5.3 System variables

The OPC server has some special variables ('Item' *names*): all start with ST_.

ST_ACTIVE_BLOCKS: integer 'Item'. Number of active data blocks.

ST_ACTIVE_POINTS: integer 'Item'. Number of active variables ('*Items'*).

ST_COMM_OK: discrete 'Item'. Is true, when communication with the instrument is running (poll inquiry send reply). You can use this 'Item' to check the communication (e.g. for cable break etc.).

ST_CYCLE_TIME: integer 'Item'. The cycle time in ms elapses, before data for *Topic* name (not the 'Item' variable) are updated again.

ST_DEVICE_VERSION: instrument ('Group') version

ST_EXECUTE:

ST_SPONT_ACTIVE: integer 'Item'. Number of spontaneous telegrams

ST_SERVER_VERSION: OPC server version

ST_WGT_A .... ST_WGT_J: string 'Items'. These variables (*tags*) contain the unit and the current **gross weight** (independent of displayed value) of weighing points A to J (e.g.: '97.10 kg'). Not valid with transmitters/ Indicator (where only *SPM 'Items'* should be used).

ST_WPA ... ST_WPJ: integer 'Items'. With controllers these variables contain the status bits of weighing points A to J. Not valid with Transmitters / Indicator (where only *SPM 'Items'* should be used).

Bit 15: standstill
Bit 14: zero
Bit 13: dimmed
Bit 12: measured value
Bit 11: entered value
Bit 10: calculated value
Bit 9: tare value
Bit 8: negative value
Bit 7: coarse active
Bit 6: fine active
Bit 5: discharging active

5.4 Database variables

Access to the databases is similar to the access to normal items.
In 1713, only database names the first character of which is not a special character are permitted.
I.e. @TABELLE is not permitted in the 1713. These names are used in the OPC server, in order to mark special items.

All values in <> stand for group or item names.
For each item of the OPC server, the client can poll for properties
(GetItemProperties for automation clients):

- `#define OPC_PROP_CDT 1 data type (VT_..)`
- `#define OPC_PROP_VALUE 2 the value(Variant)`
- `#define OPC_PROP_QUALITY 3 quality`
- `#define OPC_PROP_time 4 time stamp`
- `#define OPC_PROP_RIGHTS 5 access authorities / R,W,RW`
- `#define OPC_PROP_DESC 101 description`

(See also OPCClient – Click with the right mouse key on 'Item' in List - Properties.)
For polling properties, see: http://www.opcfoundation.org. "OPC Data Access Automation Interface Standard" or "Data Access Custom Interface Standard".

5.4.1 Reading the database type / status

- <group>.DB. @status (R)
  - IDLE
  - OPENING
  - OK
  - NO_LICENSE
- <group>.DB. <dbname> (R)
- <group>.DB. <dbname2> ....
  - e.g.: PR1713.DB.MATERIAL, PR1713.DB.RECIPE

5.4.2 Table structure

<group>.DB.<dbname>.<fieldname> (R)

- The client can poll additional properties for these fields:
  - `#define Sartorius PROP_FLAGS 6000 VT_I4 Flags:`
    - `Bit 0 set : is a key field`
  - `#define Sartorius PROP_LEN 6001 VT_I4 maximum field length (only with strings) incl. length byte`
  - `#define Sartorius_PROP_CHILD 6004 VT_BOOL true, if it is an artificial field (e.g. weight_raw)`

5.4.3 Selecting / overwriting the current field values:

- <group>.DB.<dbname>.@command (R/W)
  - 0
  - „SELECT“
  - „UPDATE“
  - „DELETE“
  - „CANCEL“
  - „RESET“
- <group>.DB.<dbname>.$<fieldname> (R/W) for example:
  - „<group>.DB.MATERIAL.$DATE
    - „> 1.3.2007“
  - „<group>.DB.MATERIAL.$VALUE
    - „“
  - „<group>.DB.MATERIAL.$ID
    - „4711“
  - For these fields, the client can poll the same additional properties as for the relevant fields without $ (see 5.4.2).
  - CAUTION: The OPC type of these fields is always VT_BSTR. This is necessary, because otherwise, it would not be possible to formulate conditions. This means e.g. for the weight value, that the client must provide the values (the string) in the unit as specified in OPC_PROP_UNIT. Actually, this should not be a problem, because the values are provided in this format by PR 1792.
  - With float values, „.“ is always used, e.g. „> 17.3“

5.4.4 Access to Select result:

- <group>.DB.<dbname>.@count (R)
  - number of records in the Select result
- <group>.DB.<dbname>.@status (R)
  - OK
  - WORKING,
  - DONE
- <group>.DB.<dbname>.@error (R)
  - contains an English error text, e.g. "String is too long"
- <group>.DB.<dbname>.@error#(R)
  - contains the following error numbers (only the number)
  - ERR_NONE = 0; ""
  - ERR_SELECT_SYNTAX = 1; "Error in select / update syntax"
  - ERR_STRING_TO_LONG = 2; "String is to long"
  - ERR_COULD_NOT_SEND_SELECT = 3; "Could not send select message, please repeat"
  - ERR_ACTION_TIMEOUT = 4; "Didn't get an answer"
  - ERR_SELECT_COMM = 5; "Had some communication problems"
  - ERR_DO_SELECT_FIRST = 6; "Please do a complete SELECT first"
  - ERR_WRONG_RANGE = 7; "Wrong range of values"
  - ERR_PR17XX = 8; "Got error message from PR17XX: "
  - ERR_COULD_NOT_SEND_DELETE = 9; "Could not send delete message, please repeat"
  - ERR_UPDATE_NOT_ALL_FIELDS = 10; "Please enter all fields before UPDATE"
  - ERR_FIELD_SYNTAX = 11; "There is an error in the field syntax"
  - ERR_COULD_NOT_SEND_UPDATE = 12; "Could not send update message, please repeat"
  - ERR_CANCELLED_BY_CLIENT = 13; "Action was cancelled by client"
  - ERR_DEVICE_BUSY = 14; "Device Database is busy"
- <group>.DB.<dbname>.@getrecord (R/W)
  - The record no. of the record set, which should be seen in the <fieldname> fields
- <group>.DB.<dbname>.@readrecord (R)
  - The record number of the record shown in RS
- <group>.DB.<dbname>.<fieldname> (R)
  - „<group>.DB.MATERIAL.DATE
    - 27.3.2007
  - „<group>.DB.MATERIAL.VALUE
    - 25.5
  - „<group>.DB.MATERIAL.ID
    - „MAT001“
- The data of a Select result are buffered in an internal OPC server data structure.
  - Contains the unchanged raw data of the record selected with @getrecord incl. CRC for the Record.
  - CAUTION: Nevertheless, only databases which are actually enabled by PR 1713 can be read. All those which don't start with @ or $ and which are enabled (not hidden) and which have at least one index field.
  - The data in the reply buffer (see below) are available in this format and are converted only for presentation via @readrecord.

5.5 Using the database variables

5.5.1 PR 1792 restart

For each configured group, PR 1792 creates the 'Item' <group>.DB.@status
and sets the status to "IDLE".

5.5.2 Finding out database names

Client "opens" the 'Item' <group>.DB.@status and waits, until @status = OK or NO_LICENSE.
For this access, license Pr1792/20 is required.

The **OPC server** checks, if a client is already connected with DB.@status. If yes, E_FAIL is fed back (method AddItems). See also 5.5.7.
The **server** starts the communication with group <group> and sets item @status to OPENING.
The **server** checks, if a DB license is provided in the instrument. Unless this is the case, @status is set to NO_LICENSE .
The server reads the database names and the table information from the controller and generates the following items for each existing database:

<group>.DB.<dbname>.@count
<group>.DB.<dbname>.@status
<group>.DB.<dbname>.@getrecord
<group>.DB.<dbname>.@readrecord
<group>.DB.<dbname>.<fieldname 1>
.....
<group>.DB.<dbname>.<fieldname n>

In addition to the actual string value, weight fields are provided with the (artificial) field xxxxx_raw with the raw value coded as a string in hexadecimal format.

and the command interface items
<group>.DB.<dbname>.@command
<group>.DB.<dbname>.$<fieldname 1>
...
<group>.DB.<dbname>.$<fieldname n>
PR 1792 setzt das 'Item' @status auf OK

Now, the **client** can find out the databases and the database fields via the OPCBrowser interface

Caution: As the database structure (table, fields, types) is read in this step, tables which were generated or deleted in the instrument subsequently cannot be read/written. An error message ‚no such table' is output in the Log window.
Tables which are continuously generated and deleted again should be declared as 'hidden' by the application programmer.

5.5.3 Reading a database (table)

The **client** has opened 'Item' <group>.DB.@status successfully (@status = OK) and knows the databases and their fields.
The **client** opens the following items for the selected databases:

<group>.DB.<dbname>.@count
<group>.DB.<dbname>.@status
<group>.DB.<dbname>.@getrecord
<group>.DB.<dbname>.@readrecord
<group>.DB.<dbname>.@command
and fields
<group>.DB.<dbname>.<fieldname 1>
.....
<group>.DB.<dbname>.<fieldname n>
<group>.DB.<dbname>.$<fieldname 1>
.....
<group>.DB.<dbname>.$<fieldname n>

The **client** checks, if <group>.DB.<dbname>.@status = OK.
Unless this is the case, the client sets @command to RESET. The **server** sets @status to OK; but the data are not available in this case.

The **client** writes the search conditions [1] into fields $<fieldname> ... and writes value „SELECT" into 'item' @command.
The search conditions are dependent of SQL conditions. The following conditions are possible:
> value: The instrument provides all data sets where the field content exceeds the specified value.
< value: The instrument provides all data sets where the field content is lower than the specified value.
The following comparison operators are permitted:
<,>,=,<=,>=,<> and *
If several $<fieldname> fields are filled with values, the conditions are interpreted as connected by an AND function.
All empty $<fieldname> fields, and fields which contain „*" are ignored.

The **server** sends the inquiry to the instrument, sets @status to WORKING, sets the <fieldname> fields to status "OPC_BAD" and waits for the reply from the instrument.
When the action is terminated, the **server** sets @status to DONE. The selected data sets are available.

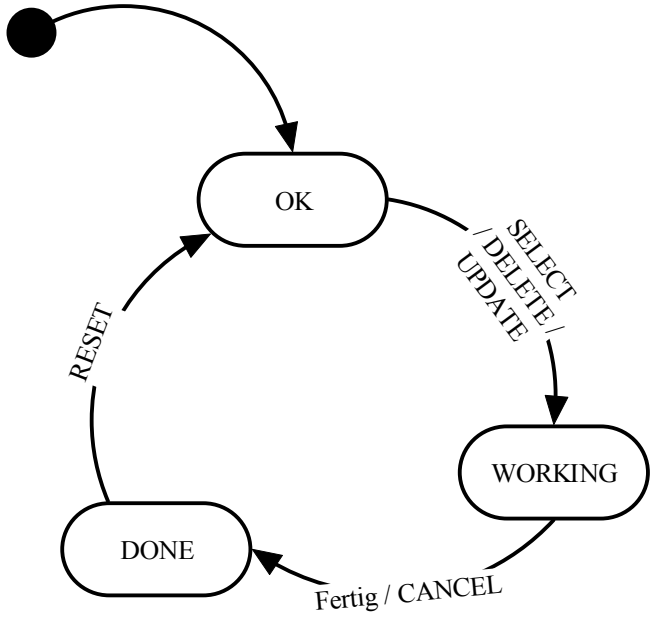
The **client** must wait for @status DONE . But it can also cancel the read operation by setting @command = CANCEL. When setting @command to CANCEL in the meantime, the **server** sends a CANCEL message to the instrument and sets @status to DONE.

As soon as the reply was received, the **server** stores the reply in an internal "reply" buffer, from which the <fieldname> fields are filled in the following. Subsequently, the **server** sets @count to the number of reply database lines, @status to OK and @command to „"
In case of an error message or timeout, @error and @error# are set to the relevant error value (e.g. ERROR_TIMEOUT).
Database fields <fieldname> are set to the contents of the first database line and @readrecord is set to 1. The status of the <fieldname> fields is set to OPC_GOOD.

Now, the **client** sets @command to RESET.
The **server** acknowledges this operation by setting @status to OK.

The **server** checks, if the value in @getrecord is permissible (>0 and <= @count) and fills the <fieldname> fields with the data of the reply line specified in @getrecord. Subsequently, the **server** sets @readrecord to the same value as @getrecord.

The resulting state diagram is:



5.5.4 Writing / updating a database (table)

The **client** has opened item <group>.DB.@status and found out that @status = OK, i.e. the client knows the databases and their fields (see Finding out database names).
The **client** opens the following items for the selected databases:

<group>.DB.<dbname>.@count
<group>.DB.<dbname>.@status
<group>.DB.<dbname>.@getrecord
<group>.DB.<dbname>.@readrecord
<group>.DB.<dbname>.@command
and fields
<group>.DB.<dbname>.<fieldname 1>
.....
<group>.DB.<dbname>.<fieldname n>
<group>.DB.<dbname>.$<fieldname 1>
.....
<group>.DB.<dbname>.$<fieldname n>

The **client** checks, if <group>.DB.<dbname>.@status = OK.
Unless this is the case, the client sets @command to RESET. The **server** sets @status to OK.

The **client** writes the values to be changed into fields $<fieldname> ... and writes value UPDATE into item @command. All fields must be written. Conditions are not possible.

The **server** composes a write message. If the instrument finds a data set which contains the same key values, this data set is replaced. If it doesn't, the new datas et is appended to the database.

The **server** checks, if all $<fieldname> fields are connected and filled in. Unless this is the case, the **server** sets @status to ERROR_NOT_ALL_ITEMS_SET. Moreover, a check, if any fields contain conditions (>,< ...) is made. If yes, @status is set to ERROR_IN_FIELDS.
If everything is o.k., the **server** sends the message to the instrument and sets @status to „3 - WORKING ...". Any reply buffer which might be provided from the last read operation (SELECT) is deleted and the <fieldname>,@getrecord etc. fields are set to OPC_BAD.
After the instrument has sent the confirmation, the **server** sets @status to DONE.

5.5.5 Deleting database (table) entries

The **client** has opened item <group>.DB.@status and found that @status = OK and knows the databases and their fields (see Finding out database names).

The **client** opens the following items for the selected databases:

<group>.DB.<dbname>.@count
<group>.DB.<dbname>.@status
<group>.DB.<dbname>.@getrecord
<group>.DB.<dbname>.@readrecord
<group>.DB.<dbname>.@command
and fields
<group>.DB.<dbname>.<fieldname 1>
.....
<group>.DB.<dbname>.<fieldname n>
<group>.DB.<dbname>.$<fieldname 1>
.....
<group>.DB.<dbname>.$<fieldname n>

The **client** checks, if <group>.DB.<dbname>.@status = OK. Unless this is the case, the client sets @command to RESET. The server sets @status to OK.

The **client** writes the conditions which must be deleted for the records into fields $<fieldname> ... and writes value DELETE into item @command. All fields must be written. Conditions are possible (e.g. higher than a datum).

The **server** composes a delete message. PR 1792 checks, if all $<fieldname> fields are connected and filled. Unless this is the case, PR 1792 sets @status to ERROR_NOT_ALL_ITEMS_SET.
If everything is o.k., PR 1792 sends the message to the instrument and sets @status to „3 - WORKING ...". Any reply buffer which might be existing from the last read operation (SELECT) is deleted and the <fieldname>,@getrecord etc. fields are set to OPC_BAD.
After the instrument has sent the confirmation, PR 1792 sets @status to DONE.

Caution:
Please note that, if a 2nd client is connected with PR 1792, PR 1792 cannot see who wrote which values. The task of the client programmer or user is to prevent 2 clients from generating contradicting values. PR 1792 sends the values existing at the time of @command changing to the instrument!

Subsequently, the **client** sets @command to RESET.
This is acknowledged by the **server** by setting @status to OK.

5.5.6 Disconnecting a database

The **client** disconnects all items pertaining to this database.
Only when all **clients** have disconnected all database items of a database, the **server** deletes the internal reply buffer of this database.

5.5.7 Access restrictions

To ensure that only one client at a time makes an access to the database, PR 1792 needs the cooperation of the clients.
PR 1792 can only check, if an item is already used by a client. PR 1792 cannot check, if it is the same client. Having two or more clients make an access to the database (e.g.: one client writes, another one reads) simultaneously has revealed to be purposeful. Not to make simultaneous write accesses to the database is in the clients' responsibility.

5.5.8 Data types

The client can find out the data type of the item as follows:

```
Dim SomeValue as Integer
SomeValue = AnOPCItem.CanonicalDataType
```

5.5.9 Data type conversion

IEC 61131	Internal presentation	Format	OPC
BOOL	1 bit		VT_BOOL
BYTE	8-bit, non-numeric		VT_UI1
WORD	16-bit, non-numeric[2]		VT_I2
DWORD	32-bit, non-numeric		VT_I4
LWORD	64-bit, non-numeric		VT_NULL
USINT	8-bit unsigned integer		VT_UI1
UINT	16-bit		VT_I4
UDINT	32-bit		VT_R8
ULINT	64-bit		VT_NULL
SINT	8-bit signed integer		VT_I2
INT	16-bit		VT_I2
DINT	32-bit		VT_I4
LINT	64-bit		VT_NULL
REAL	32-bit		VT_R4
LREAL	64-bit		VT_R8
TIME	Time difference in msec	(1.001 = 1 second, 1 millisecond)	VT_R8 (/1000)
DATE	4-byte BCD YYYYMMDD	2007-01-28	BSTR
TOD	4-byte BCD HHMMSSCC	14:45:10.12	BSTR
DT	8-byte BCD YYYYMMDDHHMMSSCC	2007-01-28-14:45:10.12	BSTR
STRING	255 p-string	BSTR	BSTR
WEIGHT	12 bytes	100,5kg	BSTR [kg|t|lbs...]
STR(N)	n p-string	BSTR	BSTR
BLOB (structures)	Byte array	01073f... or 01 07 3f	BSTR

[2] considered as a bit pattern in IEC1131. Therefore, conversion into data type is actually not purposeful and the value is converted into the corresponding signed integer value. I.e. VB clients have to work with the sign!

5.5.10 Particularities related to weight up and download

For obvious reasons, weights have a special importance for our applications. Unfortunately, the internal weight value type cannot be stored directly in Access. For this reason, every weight value in an instrument database table is offered in 3 'Items':

1. The weight value as a BSTR (123.5 kg)
2. The weight value as a VT_R8 (123.5)
3. The internal weight data type as a hex string (BSTR) of length 24.

With weight download, the following method is used:

- In case there is an error (Error xx) in the raw value (3.), the raw value is **always** written into the instrument.
- Otherwise, the weight value (1.) is compared to the raw value (3.). If both values are identical related to digit sequence (1235), exponents and unit, the raw value is also sent.
- Unless raw value and weight value (1.) are identical, a valid weight is generated from the weight value and identification "This is an entered weight value" is set and this weight is written into the instrument.

5.5.11 Marginal conditions

- In the controller, only database names allowed where the first character is not a special character. I.e. @TABELLE is not permitted e.g. in the controller. These names are used in the OPC server for marking special items.

- The databases which can be handled with PR 1792 DB must have at least one key field. All other databases are ignored (non-existent for PR 1792).

- The databases are enabled (not hidden).

- The database read operations may take quite a long time. Reading e.g. a database (a table) with 10.000 lines with „*" in the key fields takes approx. 1000 seconds. The speed is also dependent of whether the serial interface or Ethernet is used.

- **IMPORTANT:** As table reading can take a very long time and the controller memory is not sufficient to store a copy of the read results, it is impossible to ensure that the database is in a time-consistent status. Extreme example: PR 1792 starts a read operation and the controller starts transmitting the results. When a controller program deletes all lines of the table, all data sets which were already transmitted before deleting the table are indicated in PR 1792 nevertheless. I.e. the table is **not** disabled exclusively during data transmission. However, the controller disables the table whilst a table line is read.

- **The interface described in this manual is not suitable for saving all controller databases, because some databases are not transmitted and presentation of data types is not always possible without data loss.**

- The OPC server cannot detect any internal relationships between tables (databases). The individual tables are handled independently of each other. If e.g. the application is programmed so that entries in a table refer to other tables (order list makes reference to article table by an article ID), this relationship is unknown for the OPC server. The user can easily write values which do not make sense into the database (e.g. delete entries from the article table), i.e. he risks to disturb the application severely. It is the user's task to take suitable precautions.

- Another hint for the user of AccessIt: Tables should be written only, when the application in the instrument is stopped or doesn't make access to databases. Hint for the engineer who develops an application:

    - Always set databases which are only used for local purposes to 'hidden' (see DB_CREATE(): simply append ,:H' to the table name to make this database invisible for AccessIt.
    - Programmer protection of tables which can be written by AccessIt against simultaneous access must be possible (DB_OPEN with DB_LOCK#MODIFY).

- The OPC server cannot find out if data make sense for the application in the controller. Writing complete nonsense values into the table is easily possible. The situation of the controller application is comparable to a PC program working on an Access database. Writing 'nonsense' into the tables using Access isn't precluded either.

- Operating systems: Windows NT 4.0, Windows 2000, Windows XP and Windows 7.

5.6 Example

5.6.1 Examples with Visual Basic

1. Communication to the OPC server is built up.
2. ST_WGT_A is read and displayed cyclically in a timer event.
3. At program end, the communication with the OPC server is stopped.

```
Variable Declaration

Option Explicit

Private MyOPCServer As OPCServer
Private MyOPCGroup As OPCGroup
Private MyOPCItemGross As OPCItem
Private Const OPC_DS_DEVICE = 2



Private Sub Form_Load()

Dim Names(2) As String
Dim AccessPaths(2) As String
Dim ClientHandles(2) As Long
Dim DataTypes(2) As Integer
Dim ServerHandles() As Long
Dim Errors() As Long

' Connect to OPC-Server
Set MyOPCServer = New OPCServer
MyOPCServer.Connect "Sartorius OPCServer.1"
MyOPCServer.OPCGroups.DefaultGroupUpdateRate = 100
' Connect to Device PR1713
Set MyOPCGroup = MyOPCServer.OPCGroups.Add("PR1713")
' Connect to OPC-Item ST_WGT_A
Names(1) = "PR1713.ST_WGT_A"
ClientHandles(1) = 1
AccessPaths(1) = ""
DataTypes(1) = vbEmpty
MyOPCGroup.OPCItems.AddItems 1, Names, ClientHandles, ServerHandles,
Errors, DataTypes, AccessPaths
Set MyOPCItemGross = MyOPCGroup.OPCItems.GetOPCItem(ServerHandles(1))

End Sub



Private Sub Form_Unload(Cancel As Integer)

'Disconnect Items, Group and Server
Set MyOPCItemGross = Nothing
Set MyOPCGroup = Nothing
MyOPCServer.OPCGroups.RemoveAll
MyOPCServer.Disconnect
Set MyOPCServer = Nothing

End Sub



Private Sub TIM_Poll_Timer()

' Read ST_WGT_A from OPC-Server
MyOPCItemGross.Read OPC_DS_DEVICE
LAB_Gross.Caption = MyOPCItemGross.Value

End Sub
```

6 Errors

6.1 Error messages

During normal operation, the following PR 1792 messages can be reported in the logger window:

A

activatePoint: ... nActivations=...

activatePoint:: ...

activatePoint:: ... , id: ...

B

Branch::~Branch (): catch

The system cannot enter the 'item' into the active list. Do you use more than 1000 'items' per instrument?

Branch::timerTick: catched exception...

CB

CbitBlock::updateblock T:... B:... device reported error <...

You have accessed an item with an address that was rejected by the device. Check addresses of your mxxx or Mxxx items !.

CBitBlock::updateBlock: T:... B:... got unknown answer <...

You should never see that message. Please contact Technical Support.

CBitBlock::updateBlock: T:... B:... answer wrong length ...

You should never see that message. Please contact Technical Support.

CBitBlock::updateBlock: T:... B:... invalid ptType for I:...

CE

CEWPoint::setValueFromMotorolaBuffer: type VT_BSTR not implemented yet

An unimplemented data type telegram was received from the device. Please contact the Technical Support.

CEWPoint::setWriteValue (VARIANT) string truncated to ... P:...

CEWPoint::setWriteValue (VARIANT) catch e= ...

CEWPoint::setWriteValue (long lVal) catch e= ...

CEWPoint::setWriteValue (bool lVal) catch e= ...

CEWPoint::setWriteValue (float) catch e= ...

CEWPoint::setWriteValue (float) catch e= ...

CEWPoint::setWriteValue (sz) string truncated ...

CEWPoint::setWriteValue (sz) catch e= ...

CEWPoint::setValue (long lVal) catch e= ...

CEWPoint::setValue (bool) catch e= ...

CEWPoint::setValue (fltVal) catch e= ...

CEWPoint::setWriteValue (LPCTSTR) catch e= ...

CEWPoint::setWriteValue (string) catch e= ...

CEWPoint::deleteIfUnused() deleted ...
Ignore this message.

CEWPoint::printReadValue:PTT_DISCRETE ... = ...

CEWPoint::printReadValue:PTT_INTEGER ... = ...

CEWPoint::printReadValue:PTT_REAL ... = ...

CEWPoint::printReadValue:PTT_STRING ... = ...

CEWPoint::newValueFromDevice: new value for ...

Ch

checkMessageTimeout: timeout for block-msg Seq=... A:...
The server closes the *Topic* and waits for a message in vain. Please, check parameter *'message timeout'* in window *Topic configuration*. It should be higher than 5000 ms.

checkMessageTimeout: timeout for poke-msg Seq=... P:...
The server tried to send data to an instrument without receiving a reply. WARNING: The new value might be not set. Cable ok?

checkMessageTimeout: timeout for poll-msg Seq=... P:...
The server waits for a reply message in vain. Please, check parameter *'message timeout'* in window *Topic configuration*. The parameter should be higher than 5000 ms.

Co

CommMngr::send <...

CommMngr::read <...

CommMngr::checkCommStatus: Can't activate Group ...
The OPC server was unable to build up communication with the instrument. Check connections, Baudrate, slave address and Com port in window Group configuration.

CommMngr::checkCommStatus: No communication on ... . Waiting...
The server is trying to connect to the device (group). Please check cables, Group configuration and device.

CommMngr::checkCommStatus: Communication initialized on ...
This is not an error message. If you see that message the communication with this group (device) has started successfully.

CommMngr::checkCommStatus: No communication on ...
The communication was active but now it stopped. Please check the device and the cables.

CommMngr::checkCommStatus: initializing communication ... . Waiting...
The server is initializing a device. After a few seconds a success message should appear. Otherwise the device may not be connected.

CommMngr::checkCommStatus: PR 17xx must have software config:sequence number enabled
The device cannot handle the new protocol with sequence numbers. Please check if you have the right firmware version!

CommMngr::checkCommSTatus: <... device identifies as <PR... Check Group config
The connected device is not the same type as in the group configuration. Correct the group configuration to match the connected device.

CommMngr::checkCommStatus: Controller must have software config:sequence number enabled

The device cannot handle the new protocol with sequence numbers. Please check if you have the right firmware version!

CommMngr::checkCommStatus: wrong vmb answer

The controller did not send the expected answer to a version request. The firmware version of the device is wrong (probably too old). Please check and update. Or the connected device is not the specified one.

CommMngr::checkCommStatus: wrong controller HW or SW version

The firmware version of the device is wrong (probably too old). Please check and update.

CommMngr::checkCommStatus: could not activate spontaneous messages group: ...

OPC server tried to activate (with the 'pss') message the spontaneous messages and the device refused. Please check firmware version.

CommMngr::getLicenseInfo: could not get license info for group: ...

The OPC server tried to get the license key information from the device but didn't get an answer. Communication problems?

CommMngr::getLicenseInfo: no license for Server, group: ... Only DisplayIt possible!

You did not install the license for OPC Server communication in the device. Please check. You can only use a few items.

CommMngr::checkCommStatus: catched error

CommMngr::checkCommStatus: could not get phase config version info for group: ...

The OPC server tried to check the configuration info but the device didn't answer. Please check communication and device version.

CommMngr::checkCommStatus: phase configuration (version) ok, logicals enabled ...

CommMngr::checkCommStatus: phase configuration NOT ok (version), logicals disabled ...

This message is normal if you don't use the PR 1781 PhaseConfigurator program. The configuration info stored in the .ini file differs from the configuration info in the device. Did you make a cold start or changed the .ini file manually? Did you initialise the device with the PR 1781 program and did you get any error messages?

CTopic.sendMessage: m_messageList full

You are probably sending too much too fast.

CommMngr::writeDevice:must split: ...

CommMngr::sendMessage: error in ew_dev_write : ... SEQ=...

You should never see that message. Please check the ewdrvr_01.exe window for error messages.

CommMngr::sendMessage: catch..

CommMngr::sendMessageAndWait: could not send message:

The server tried to send a message to the device but failed. The next line in the logger is the message, that was not send.

CommMngr::sendMessageAndWait: got error msg: ... ...

The system tried to poke (set) a value in the device but got an error reply. Please note the first two (hexadecimal) values. In most cases there was an error while initialising (check PR1781 init. function).
This is a list of the error codes:
1 1 Command is unknown
1 2 Error in command format (too few or too many parameters)
1 3 Parameter was not in allowed range
1 4 No execution
5 7 Device is out of memory
5 a unknown FB (check PhaseConfiguration / Init)
5 b SPM address not valid
5 c Variable type not valid
5 d No instance
5 e Not initialized (PhaseConfig init?)
5 f Group already existing
5 10 Instance number not valid
5 11 Symbol number not valid
5 12 Write on VAR_OUTPUT not possible
5 13 Invalid unit number
5 14 No memory for internal mail (device) available
5 15 Group does not exist
5 16 No license for PR 1781/00
5 17 Invalid weighing point address
5 18 Negative float not allowed
5 19 Device busy with production: PhaseInit not allowed.

CommMngr::sendMessageAndWait: got wrong message

CommMngr::sendMessageAndWait: error: no reply or error

The server tried to send a message to the device but failed.

CommMngr::checkMessageTimeout: timeout for message Seq=... P:...

CommMngr::checkMessageTimeout: wrong message type

You should never see that message. Please contact Technical Support.

CommMngr::checkMessageTimeout: message timeout while group: ... shutdown pending

The server is closing the group and waiting for a reply message, but the message didn't come. Please check the 'message timeout' parameter in the group configuration window. It should be at least 5000 ms.

CommMngr::IOTopicOpenForPoll(): ew_dev_open return = ... ERROR

CommMngr::closeDeviceAndCleanup: communication with Group ... closed

CommMngr::getMessage: got message with wrong sequence number SEQ=... I=... <...

You should never see that message if the communication wasn't down.

CommMngr::detachClient: ignore message <...

Config file is : ...

For your information.

CS

CShellCallback::remove: invalid item
Someone (?) tried to remove an invalid item.

CShellCallback::ReadTag: invalid Item ....
You should never see this message. Please note the message and contact Technical Support

CShellCallback::ReadTag: catch ....
You should never see this message. Please note the message and contact Technical Support

CShellCallback::WriteTag: ... wrong data type
The data type was wrong and could not be transformed correctly.

CShellCallback::WriteTag: invalid item
You should never see this message. Please note the message and contact the technical support

Cr

createPoint: unrecognized Item ... rejected
The specified *'Item' name* is not known in the system. Valid names are *mx.. mr.. md.. mb...* oder *ST_...*

CTo

CTopic:doInits:: T: ... unknown ioState ...
If you see this error, please, contact Technical Support.

CTopic::AddTag: could not read device (cable?, right connection?, Licence?)
OPC server cannot communicate with the device.

CTopic::AddTag: could not create item ...
'Item' could not be generated.

CTopic::createPoint: catch error in DisplayIt Mode ...
internal error.

CTopic::createPoint: Item name is empty!
The specified *'Item' name* is zero (empty) "". Please, check the name.

CTopic::createPoint: Item name is too short or too long: ...
The *'Item' name* is too short or too long, mx1 is the shortest possible name.

CTopic::createPoint: no license for this item: ...
The *'Item' name* requires a licence.

CTopic::createPoint_SPM: address ... of Item ... too large, rejected
Invalid address for the variable. Is the address used in the SPM area (memory) of the device? For the following devices, the highest addresses are:

Controller	Transmitter / indicator
mx8191	mx1023
mr255	mr31
md255	md31
mb1022L...	mb126L... (minus string length)

CTopic::createPoint_SPM: invalid string length in Item name: ...
The maximum string length is 131 characters.

CTopic::createPoint_SPM: no address field in 'Item' name: ...
Tag name must contain an address, e.g.: mxx is wrong !

CTopic::createPoint: unrecognized Item name: ...
The variables (*Item names*) must be: mx *(bit value),* md *(integer),* mr *(real),* mb *(string)* or ST_...

CTopic:getReplies/case poll/DONE value = ...

CTopic:getReplies catched error ...

CTopic::RemoveTag:catched error
internal error.

CTopic::sendMessageAndWait: could not send message:
The server wanted to send a message to a device, which failed. The next line in the logger contains the message.

CTopic::setIniFileName: Couldn't set INI filename <...
Error during writing the INI file for the CFG file. Try to re-initialize.

CTopic::setIniFileName: Couldn't read INI filename <...
The file could not be read.

CTopic::timerTick: catched exeption...
Internal error

CTopic::updateBlockPoint: P:... found not activated

CTr

CTransLogic::getFbList: could not find ... in .ini file
The format of the INI file is not correct. Re-generate it with PR 1781.

CTransLogic::getFbList:could not find ... in .ini file ... path?
The INI file format is not correct. Re-generate it using PR 1781.

CTransLogic::getFbList:error in ini file (expected 2 values)
The INI file format is not correct. Re-generate it using PR 1781.

CTransLogic::getFbList:could not find ... in read .ini file ... path?
The INI file format is not correct. Re-generate it using PR 1781.

CTransLogic::getFbList:error in ini file ((MaxUnits) expected 1 values)
The INI file format is not correct. Re-generate it using PR 1781.

CTranslogic::getPhyName: using ini file: ...
Information only, not an error

CTransLogic::getPhyName: could not find ... in .ini file ...
The INI file format is not correct. Re-generate it using PR 1781.

Da

Data received on ew_dev_open() seq=...

Db

DbBranch::AddTag: WARNING: a second active connection to DB items was added
Please make sure that only one client is connected to this database item.

DbTable::getTable(): got error message: ...

DbPoint::setWriteValue (VARIANT)

DbTable::setState: The client tried to access two database tables of the same device at the same time
The client tried to access two database tables of the same device at the same time This is not allowed.

DbTable::pokePoint: SELECT, wrong state
The table @status must be IDLE.

DbTable::pokePoint: CANCEL, wrong state (idle)
It doesn't make sense to send CANCEL if state is IDLE.

DbTable::pokePoint: UPDATE, wrong state
The table @status must be IDLE.

DbTable::pokePoint: DELETE, wrong state
The table @status must be IDLE.

DbTable::pokePoint: RESET, wrong state
The table @status must be DONE.

DbTable::prepareData: error in data : ...
Please check the $xxx item values.

DbTable::prepareData: error in item ...
Please check the $xxx item values.

DbTable::sendUpdate: could not send update message
PR 1792 could not send the message. Too busy.

DbTable::sendUpdate: could not assemble update record
There was an an error in the update record data.

DbTable::prepareUpdateData: error in item ...
Please check the $xxx items for syntax errors.

DbTable::checkSelectAnswer(): got error message: ...
The device sent an error message. Please check the message.

DbTable::checkSelectAnswer(): got 0 records (no records found)
No records found for this SELECT.

DbTable::checkSelectAnswer(): got 0 records (table empty)
No records found for this SELECT.

DbTable::checkSelectAnswer(): crc error - repeating
Checksum error found. Please check communication line if this message is seen often.

DbTable::checkSelectAnswer(): could not send repeat
Checksum error found and repeat failed. Please check communication line if this message is seen often.

DbTable::checkSelectAnswer(): got message with wrong sequence number. abort
Got a wrong answer. This message should be very rare.

DbTable::checkSelectAnswer(): catch
You should never see this message. Please note the message and contact technical support.

DbTable::checkSelectAnswer(): timeout
The SELECT timed out. Device too busy?

DbTable::checkDeleteAnswer(): 0 records deleted (table empty)
The table is empty.

DbTable::checkDeleteAnswer(): got error message: ...
The device sent an error message.

DbTable::checkDeleteAnswer(): catch
You should never see this message. Please note the message and call technical support.

DbTable::checkDeleteAnswer(): timeout
The action timed out. Device too busy?

DbTable::checkUpdateAnswer(): got error message: ...
See error message.

DbTable::selectRecord: No valid data available. Please do a SELECT first
The previous SELECT must be finished without errors and the state must be IDLE. Do not forget the RESET.

DbTable::selectRecord: ...
If for example the SELECT produced 20 records the possible range is 1..20.

DbTable::selectRecord(): catch
You should never see this message. Please note the message and call Technical Support.

De

deactivatePoint: ... id=... ActivationCtr=...

deactivatePoint:: ... id=...

E

EW-DLL version: ...

For your information.

EW-DLL version: not available (driver not running)

The ewdrv_01.exe is not running or hanging. Please try to kill ewdrv_01 via task manager.

EW-driver version: ...

For your information.

ew_dev_close()

ew_dev_open()

F

Error code list

1 1	Unknown command
1 2	Command format error (number of parameters too low / high)
1 3	Parameter out of permitted range
1 4	Not executed
5 7	Memory in device full / Invalid data set number
5 a	Unknown FB (check phase configuration / Init)
5 b	Invalid SPM address
5 c	Invalid variable type
5 d	Data set missing
5 e	Not initialized (phase configuration Init?)
5 f	Group exists already
5 10	Invalid data set number
5 11	Invalid symbol number
5 12	Writing on *VAR_OUTPUT* not possible
5 13	Invalid unit number
5 14	No free memory space for internal mail (instrument)
5 15	Group does not exist
5 16	No licence for PR 1781/00
5 17	Invalid weighing point address
5 18	Negative deviation not permitted
5 19	Instrument busy: PhaseInit not permitted

G

getReplies: SPM poll answer, unrecognized point type

If you see this error, please contact the Technical Support.

I

Invalid EW-driver version format

Version conflict between ewdriver, ew30dll and PR 1792.

Invalid license number for connected transmitter

Enter the correct license number for the instrument. Instrument board number B:xxxxxxxxx can be used to order the license.

L

Lib Version: ...

For your information.

N

No BoardNr found in QV...

Instrument not detected as expected (V command). Please, check the firmware and update it, if necessary.

O

Obsolete EW-driver version, collaboration denied

Version conflict between ewdriver, ew30dll and PR 1792.

P

pokePoint: could not poke P: ... poke list is full!!!
You should never see this message. Perhaps you tried to write at a higher speed than permitted by communication.

pokePoint:: STATUS ... handle=... : poke denied
Setting of 'ST_...' 'Items' not possible.

pokePoint:: poke requested: ... ...

S

Server Product version: ...
For your information.

Server File version: ...
For your information.

U

updateBlockPoint: poll answer, point is not SPM
If you see this error, please contact the Technical Support.

updateBlockPoint: poll answer, unrecognized point type
If you see this error, please contact the Technical Support.

UpdateMngr::getReplies_poll: got poll reply for deactivated point

UpdateMngr::getReplies_poll: got poll reply for invalid point

UpdateMngr::getReplies_poll: got poll reply for I:...

UpdateMngr::getReplies_poll: got poll reply for I:... but state != pollSent

UpdateMngr::getReplies_poll:phase poll answer T: ... P: ... disc rlen=...
If you see this error, please contact the Technical Support.

UpdateMngr::getReplies_poll:phase poll answer T: ... P: ... int rlen=...
If you see this error, please contact the Technical Support.

UpdateMngr::getReplies_poll:phase poll answer T: ... P: ... real rlen=...
If you see this error, please contact the Technical Support.

UpdateMngr::getReplies_poll: Phase poll answer, unrecognized point type
If you see this error, please contact the Technical Support.

UpdateMngr::getReplies_poll: group ... Point ... poll answer >... ... ...
If you see this error, please contact the Technical Support.

UpdateMngr::updatePoints: G: ... I: ... logical points disabled, no poll....

UpdateMngr::updatePoints: found item in spontValueMap: ... ...

UpdateMngr::getReplies/simple:: unrecognized answer $... $... $... $...
If you see this error, please contact the Technical Support.

UpdateMngr::getReplies:: got poke reply for I:...

UpdateMngr::getReplies:: got poke reply for

UpdateMngr::getReplies:: got poke error-reply for I:... ... ...

The system tried to poke (set) a value in the device but got an error reply. Please note the first two (hexadecimal) values. In most cases there was an error while initialsing (check PR 1781 init. function).
This is a list of the error codes:
1 1 Command is unknown
1 2 Error in command format (too few or too many parameters)
1 3 Parameter was not in allowed range
1 4 No execution
5 7 Device is out of memory
5 a unknown FB (check PhaseConfiguration / Init)
5 b SPM address not valid
5 c Variable type not valid
5 d No instance
5 e Not initialized (PhaseConfig init?)
5 f Group already existing
5 10 Instance number not valid
5 11 Symbol number not valid
5 12 Write on VAR_OUTPUT not possible
5 13 Invalid unit number
5 14 No memory for internal mail (device) available
5 15 Group does not exist
5 16 No license for PR 1781/00
5 17 Invalid weighing point address.

UpdateMngr::getMessage wrong message type!

If you see this error, please contact the Technical Support.

UpdateMngr::updatePoints ignoring answer due to pending group shutdown

Information only, no error.

UpdateMngr::getReplies_spontan: got spontan 'sps' telegram

UpdateMngr::getReplies_spontan (sps): added to spontValueMap ...

Debug message: The new value was added to an internal map structure.

UpdateMngr::getReplies_spontan:: deny spont new value due to pending poke I:...

There is a poke pending. The server will wait until the delivery is confirmed before accepting new values.

UpdateMngr::getReplies_spontan: logical items disabled, skip

Server got a spontanous message from the device but the logical addresses are disabled. Check if the licenses are installed in the device and if the initialization with PR 1781 worked correctly (see logger).

UpdateMngr::getReplies_spontan:(sps) updating item >... ...

UpdateMngr::getReplies_spontan: could not find item in active List

Server got a spontaneous message from the device but the point was not activated. You should never see that message. Please contact the Technical Support.

UpdateMngr::getReplies_spontan: got spontan 'spa' telegram

UpdateMngr::getReplies_spontan (spa): found in spontActiveMap

UpdateMngr::getReplies_spontan:: deny spont new value due to pending poke I:...

UpdateMngr::getReplies_spontan: logical items disabled, skip

There is no license for logical addresses or PR 1781 init didn't work.

UpdateMngr::getReplies_spontan (spa): updating item >... ...

UpdateMngr::getReplies_spontan: got unrecognized spont message <...
This could be a spontaneous error message from the device.

UpdateMngr::getReplies_spontan: got wrong weighing point index

UpdateMngr::cleanup: catch error

UpdateMngr::getActualsForSpontanPoints: message list full?: ...
If you see this error, please contact the Technical Support.

UpdateMngr::processActivationSpontan: sending activation message

UpdateMngr::processOtherSends: sending message

UpdateMngr::processOtherSends: ERROR sending message
If you see this error, please contact the Technical Support.

UpdateMngr::processPokes: poking P: ...

UpdateMngr::processPokes: ERROR poking P: ...
If you see this error, please contact the Technical Support.

UpdateMngr:phaseInit(): Unknown command:...
This command is not valid.

UpdateMngr:phaseInit(): cmd:... Device NOT alive (open one of the ST_ items))
The communication with the device is not possible. Please check cables and the device.

UpdateMngr:phaseInit(): cmd:... MessageList not empty
An internal list is full. Please try again.

UpdateMngr:phaseInit(): cmd:... Can't init
The device was NOT!! initialised. You have to try again.

UpdateMngr::phaseInit: command is wrong (path?): ...
Something is wrong with the execute string. It should look like PhaseInit c:\....\yourinifile.ini.

UpdateMngr::phaseInit(): PhaseInit <as> BEGIN

UpdateMngr::phaseInit(): <as> ERROR. Device busy (phases active)? Phase configuration license not installed?
Phase configuration rejected by device, possible reasons: - The license for phase configuration is not installed in the device - The device is busy in a production. Wait until production is DONE. Emergency: execute device-coldstart.

UpdateMngr::phaseInit(): PhaseInit <as> OK

UpdateMngr::phaseInit(): PhaseInit <aw ... ... Begin

UpdateMngr::phaseInit(): PhaseInit <aw ... ... ERROR
If you see this error, please contact the Technical Support.

UpdateMngr::phaseInit(): PhaseInit <aw ... ... OK

UpdateMngr::phaseInit(): PhaseInit <au> BEGIN

UpdateMngr::phaseInit(): PhaseInit <au> ERROR
If you see this error, please contact the Technical Support.

UpdateMngr::phaseInit(): PhaseInit <au> OK
If you see this error, please contact the Technical Support.

UpdateMngr::phaseInit(): PhaseInit <ae> BEGIN

UpdateMngr::phaseInit(): PhaseInit <ae> ERROR
If you see this error, please contact the Technical Support.

UpdateMngr::phaseInit(): PhaseInit DONE OK

UpdateMngr::pokePoint: cannot poke. Shutdown pending...
If you see this error, please contact the Technical Support.

UpdateMngr::pokePoint: logical items disabled, not poking ...
Check if the licenses are installed in the device and if the initialisation with PR 1781 worked correctly (see WWLogger). Did you cold start the controller? If so, you have to reinitialize with PR 1781.

UpdateMngr::closeDeviceAndCleanup: closing communication

UpdateMngr::closeDeviceAndCleanup: waiting for ... pending pokes...

UpdateMngr::closeDeviceAndCleanup:maxRetryCount (pokes) reached .. abort
The server tried to send all pending poke messages but did not succeed. If the communication was down this is normal.

UpdateMngr::closeDeviceAndCleanup: waiting for ... pending messages...

UpdateMngr::closeDeviceAndCleanup:maxRetryCount (messages) reached .. abort
The server tried to send all pending messages but did not succeed. If the communication was down this is normal.

UpdateMngr::activateST_WGTs: otherSendFifo is full!!!
If you see this error, please contact the Technical Support.

6.2 Trouble shooting

Problem: communication does not work
- Check Baudrate
- Check slave address
- Check COM port
- Disconnect OPC server from client, terminate the OPC server and finish using task manager EWDRV1.exe. Subsequently, have the client re-connect the server.

Problem: When the client connects the server, Sartorius. *O*PCServer.1 is not displayed as available server.
- OPC server registration was not possible. E.g. during installation, no administrator privileges were provided.
Execute file "REGALL.BAT" in PR 1792 directory and repeat registration.

7 Annex

7.1 Further literature

Fundamental information on OPC and reference to recent literature is available from the OPC user organization:

http://www.opcfoundation.org/

7.2 Abbreviations and glossary

Application	The OPC server name. The OPC application name of PR 1792 is always "PR 1792".
OPC	Dynamic Data Exchange is a service of the Windows NT operating system, which enables applications to exchange data according to a standard method. Exchange is possible also between applications on computers in a network.
Database	The term 'database' is used in the controllers. A "database" is actually a table of the internal controller database.
Dec.	(as an index) Abbreviation of decimal number.
Item	Part of OPC addressing (name of variable, 'Item').
PLC	Programmable Logic Controller
SPM	Scratch Pad Memory in the instrument.
OPC Item	Part 'Item' of the OPC address contains the *Item name*. A data item (variable), which contains e.g. a value, status or a command. Data types: *Discrete*, *Integer*, *String*, *Real*
OPC Group	Part of the OPC address (name of connection). An OPC address is composed of 'Application', 'Group' and 'Item' . All OPC addresses of an instrument belong to the 'group' of the same 'application'. Each instrument must have an independent 'group'.
WP	The weighing point parameter (WP) indicates from which weighing module the function module has to fetch a weight value.

8 Index

Sartorius Mechatronics T&H GmbH
Meiendorfer Straße 205
22145 Hamburg, Germany
Tel +49.40.67960.303
Fax: +49.40.67960.383
www.sartorius-mechatronics.com