Facility components

The following illustration shows the major components of an automated welding system with an integrated optical seam tracking system.

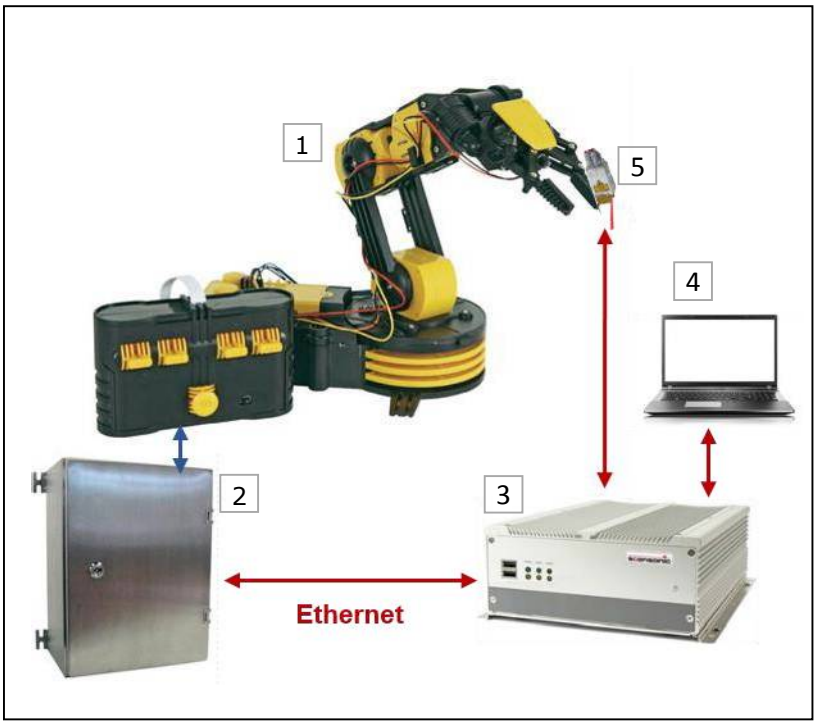

Figure 2

1 Facility

2 Facility controller

3 TH6D embedded PC (EPC)

4 Operator PC for the graphical user interface (GUI): THxDView (PC is not included in the delivery.)

5 TH6D sensor (Tracking Head 6 Dimensions)

The other components of an automated welding system are not relevant to the interactions between the facility controller and the sensor. The facility controller performs all functions associated with the automated manufacturing process. Thus, the specific welding parameters are retrieved or modified at the facility controller. This is also where the connection to the facility controller is controlled.

Connection overview

IMPORTANT

The TH6D sensor must always be properly earthed (grounded)!

Be sure to follow the *Earthing provisions* as described in the Installation chapter (Chapter 4) of the TH6D manual.

Ethernet is used for the data exchange between the facility controller and the TH6D sensor system. A real-time peer-to-peer (P2P) connection is required for both communication nodes (the facility controller and the TH6D system). This is why they use static IP addresses which only differ in the last number (octet).

A programmable switch must be used for the network connection. You must ensure that this data connection is capable of real-time communications.

For this reason, only the RTETH1 port on the TH6D embedded PC (3) should be used for the connection to the facility controller.

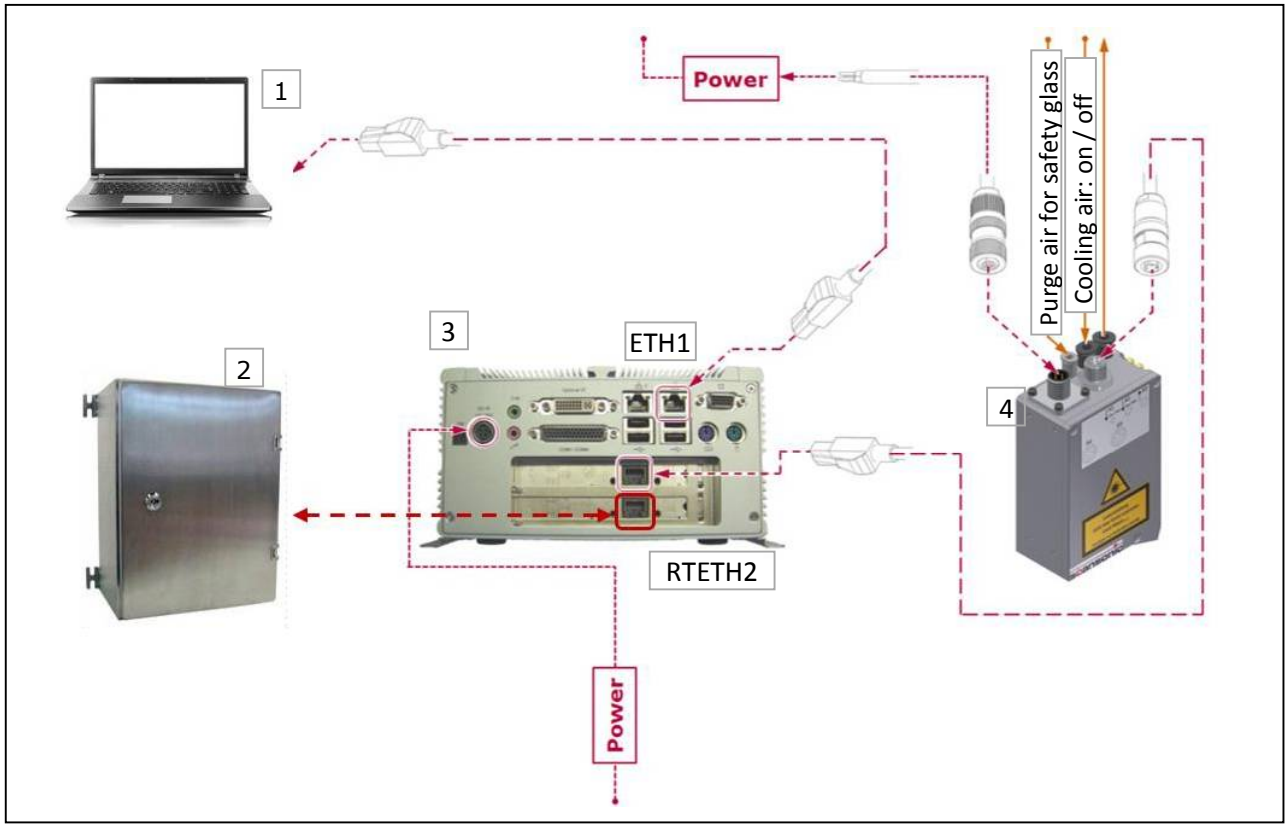

Figure 3

1 Operator PC with THxDview software

2 Facility controller (PLC)

3 TH6D process computer (EPC)

4 TH6D sensor (Tracking Head 6 Dimensions)

4. Data UDP protocol

4.1 Protocol between the TH6D software and the facility controller

The DATA UDP communication protocol uses DATA UDP notation. It uses only ASCII characters from 32 to 126 (0x20 – 0x7E) (the UTF-8 standard).

ISO reference model

The DATA UDP communication protocol is based on the communication principles described in the ISO-OSI reference model. It is based on the lowest layer, on Ethernet. The DATA UDP communication protocol is implemented in the ISO layers 5 - 7.

Number	Layer	Protocol
7	Application layer	DATA UDP (default ASCII 0-127)
6	Presentation Layer	
5	Communication layer	
4	Transport layer	TCP
3	Network layer	IP
2	Data link layer	Ethernet
1	Bit transfer layer	

System configuration

The facility controller is connected via Ethernet and TCP/IP with the TH6D embedded PC. The communication uses TCP/IP sockets with a corresponding IP address and port.

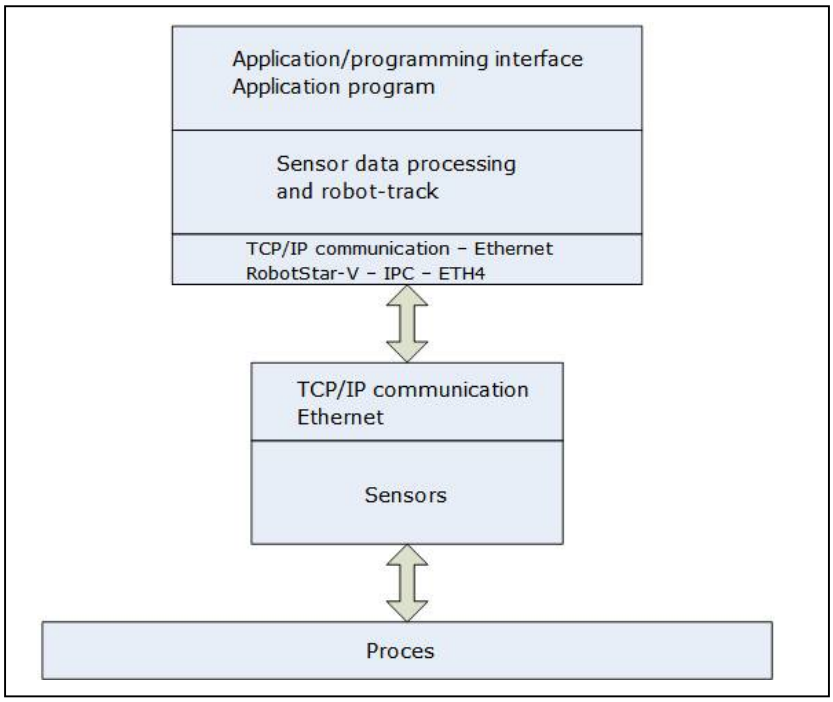

Figure 6:

Synchronization All facility controller commands and corresponding sensor responses have timestamps. The timestamps are in microseconds, based on a signed 32-bit data type (0 to 2147483647). The first controller command contains the "reception timestamp" 0.

Example In this example, the command contains the timestamp of the controller and the reception timestamp of the last response. The response contains the sensor's timestamp, which also contains the computation time of the data and is listed on the transmitted timestamp of the sender (in this example, 1 ms).

<cmd tsp="14000" rtsp="13000" send="Robot1" recv="Sensor">
…
<rep tsp="15000" send="Sensor" recv="Robot1"> …

4.1.1 Structure of the communication protocol

Protocol structure Commands may only be sent by the facility controller. Each command receives a corresponding response. A command message from the facility controller contains only one command. Only the two commands to set and read parameters (*getPar, setPar*) can be combined in one command message to save time. Sensor information is only requested by the facility controller (the sensor does not independently send message data).

Each message contains:

Command-Tag When necessary, Command-Tag can also contain attributes.

Timestamp tsp: This attribute contains the current system time of the sender when sending the message.

rtsp: This attribute contains the last response received from the sensor (for synchronization purposes).

Sender Contains the symbolic identifier of the sender.

Receiver Contains the symbolic identifier of the recipient.

Example Request of the facility controller (for the first command):

```
<cmd tsp="12345" rtsp="0" send="Robot1" recv="Sensor" >
    <setPar  p1="10" />
    <getPar  p3="Par1" />
</cmd>
```

Response from the sensor

```
<rep tsp="13456" send="Sensor" recv="Robot1" />
    <setPar  res="1" error="ErrorCode" />
    <getPar  res= "1" p3="P3Content" error="ErrorCode" />
</rep>
```

The sequence of tags and attributes within a message may vary. Missing tags or attributes (within a message) will not cause a parser error.

Commands in use

Commands	Message of the facility controller	Response from the sensor	Data
setPar	Set Parameter (p1 is the job number)	setPar res="1"	Ack
getPar	Get Parameter	getPar res="1"	Parameter Value
camOn	Camera On	camOn res="1"	Ack
camOff	Camera Off	camOff res="1"	Ack
camEn	Camera Enable	camEn res="1"	Ack
camDis	Camera Disable	camDis res="1"	Ack
getVal	Get Position Values	getVal res="1"	Camera Data

res = "1 " : okay

res = "-1": error

res = " 2":Tack weld is present (getVal – command)
All of the values delivered in the response message are valid.

res = "3": No valid data available (getVal – command)

res = " 4":Invalid range (getVal – command)
A tack weld may be present here. "Area" and "TrackingPoint" may be invalid; only "gap" is a reliable value.

res = "3": Invalid gap (getVal – command) "Gap" is invalid; "Area" and "TrackingPoint" may also be valid. A tack weld may be present.

res = "6": Invalid tracking point (getVal – command) "Area" is invalid. "Gap" and "Mismatch" are the only permitted values. "TrackingPoint" may be invalid; a tack weld may be present.

res = "7": "TrackingPoint" is invalid; "Gap" and "Mismatch" are permitted values; a tack weld may be present.