System description

netPROXY

V2

Hilscher Gesellschaft für Systemautomation mbH
www.hilscher.com

DOC160203SD02EN | Revision 2 | English | 2018-02 | Released | Public

Table of contents

1 Introduction

This manual describes the netPROXY application framework for designing communication devices in a protocol-stack-independent way.

1.1 What is netPROXY

netPROXY provides an object-based framework for device development with a uniform, protocol-independent host interface. This approach allows to design device variants with different communication systems but identical scope of functions in an extremely effective way. Consequently, development efforts and costs can be reduced significantly.

1.2 List of revisions

Revision	Date	Author	Changes
1	2017-11-23	RG	Created
2	2018-02-19	HH	Section *netPROXY limitations* [page 10] added.

Table 1: List of revisions

1.3 Terms, abbreviations and definitions

Term	Description
netPROXY	An application framework for efficient development of networked devices based on Hilscher netX. It provides an object model as a generic abstraction for services or data. netPROXY is also used as a general term related to netPROXY technology of Hilscher.
Group	Logical arrangement of related device functions in netPROXY. The functions can be grouped i.e. by roles: by physical role (IO-Ports, measurement channels, ...) or by functional role (controller, monitor, processing unit, drive axis ...)
netX Studio	Eclipse-based tool for netPROXY project design, management etc.
Object	A defined data structure consisting of data elements. Each data element contains the data of a defined data type.
Object instance	An instance with the data structure of the object. An object may provide one or more instances. Object Instances are used to provide copies with the same Data structure. Example: multiple analog measurement channels provide analog values and status as data elements of the data structure. The channels can be mapped in this case to object Instances.
Package	A set of netPROXY objects providing defined behavior (i.e. function). It will be installed to a specific group
Eclipse	GUI-based efficient integrated development environment into which netStudio is integrated. See http://www.eclipse.org.

Table 2: Terms and definitions

1.4 Documentations overview

The following figure shows the components and interfaces for the integration of a netIC IOT communication module into a host system. You need to use the for developing your device. The numbers refer to the relevant document.

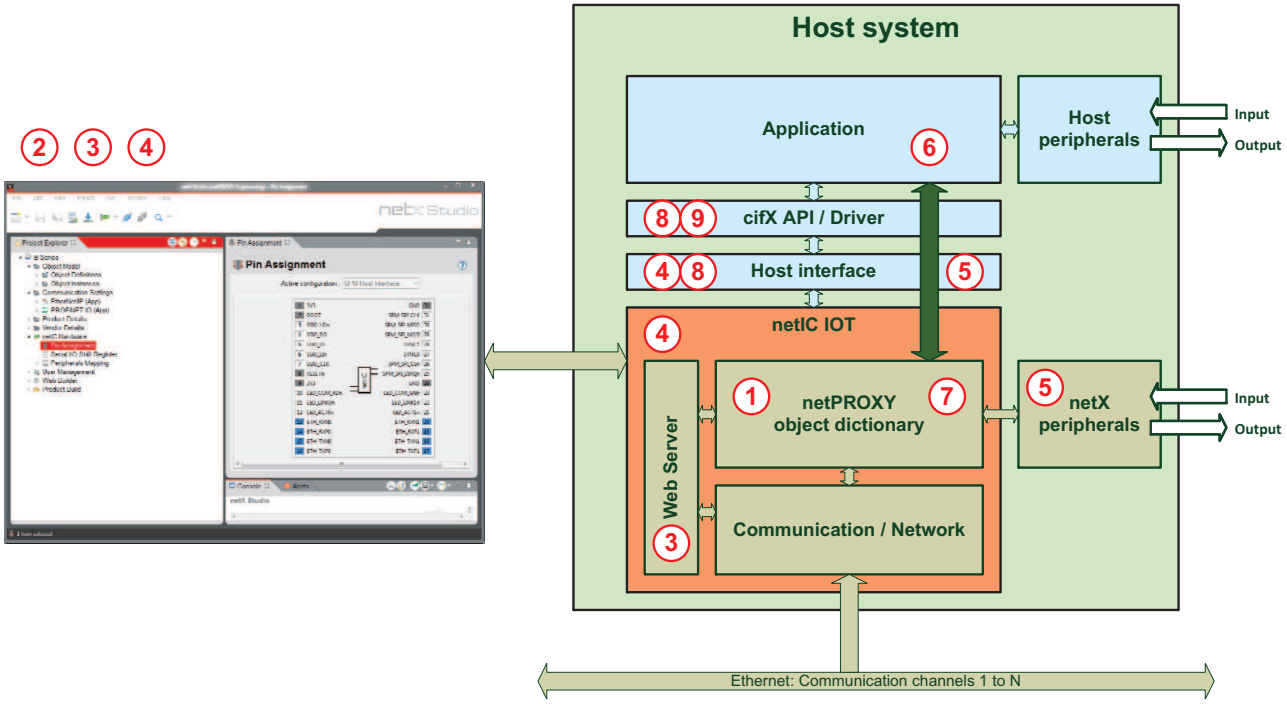

Figure 1: netIC IOT documentations overview

#	Target group	Document	Content
(1)	Product developer, Software developer	System description: netPROXY, DOC160203SDxxEN, English.	Description of the object-based framework for developing devices.
(2)	Product developer	Operating instruction manual: netX Studio Engineering Tool, Device development, DOC160103OIxxEN, English.	Device development using the netX Studio Engineering Tool. Description of the OPC UA, MQTT and Real-Time Ethernet configuration.
(3)	Product developer	Application note: netX Studio Engineering Tool, Web Builder, DOC160207ANxxEN, English	Functions and use of the Web Builders to create and manage HTML web pages.
(4)	Hardware developer, Product developer	Operating instruction manual: netIC IOT, Configuring hardware with netX Studio, DOC160501OIxxEN, English.	Configuring pin assignment and setting of interface parameters.
(5)	Hardware developer	Design guide: netIC IOT, NIOT-I-IC52-RE, DOC141002DGxxEN, English.	Integrating netIC IOT Hardware into the host system.
(6)	Software developer	Protocol API: netPROXY Host API, Function interface, netPROXY Host API.chm, English.	API for the host application to get access to objects in a device.
(7)	Software developer	Technical reference: netPROXY object reference, DOC160204TRxxEN, English.	Description of the netPROXY objects.
(8)	Software developer	Toolkit manual: cifX/netX Toolkit, DPM, DOC090203TKxxEN, English.	Description and usage of the cifX C-Toolkit.
(9)	Software developer	Programming reference guide: cifX API, DOC121201PRxxEN, English.	Description and usage of the standard cifX API.

Table 3: Documentation overview

1.5 References

This document refers to the following documents:

1. Hilscher Gesellschaft für Systemautomation mbH: Protocol API: Protocol API: netPROXY Host API, Function interface, netPROXY Host API.chm, English, 2018.
2. Hilscher Gesellschaft für Systemautomation mbH: Technical reference: netPROXY object reference, DOC160204TRxxEN, English.

2 netPROXY advantages

2.1 Uniform access to different protocol stacks

netPROXY provides uniform access to devices running different protocol stacks for different Fieldbus or Real-time Ethernet systems. It helps to reduce development costs in projects dealing with multiple protocol stacks significantly.

You can consider netPROXY as an intermediate layer between the user application and the protocol stack. The following figure shows how netPROXY establishes an application abstraction layer on top of the protocol stack with a transparent object model.

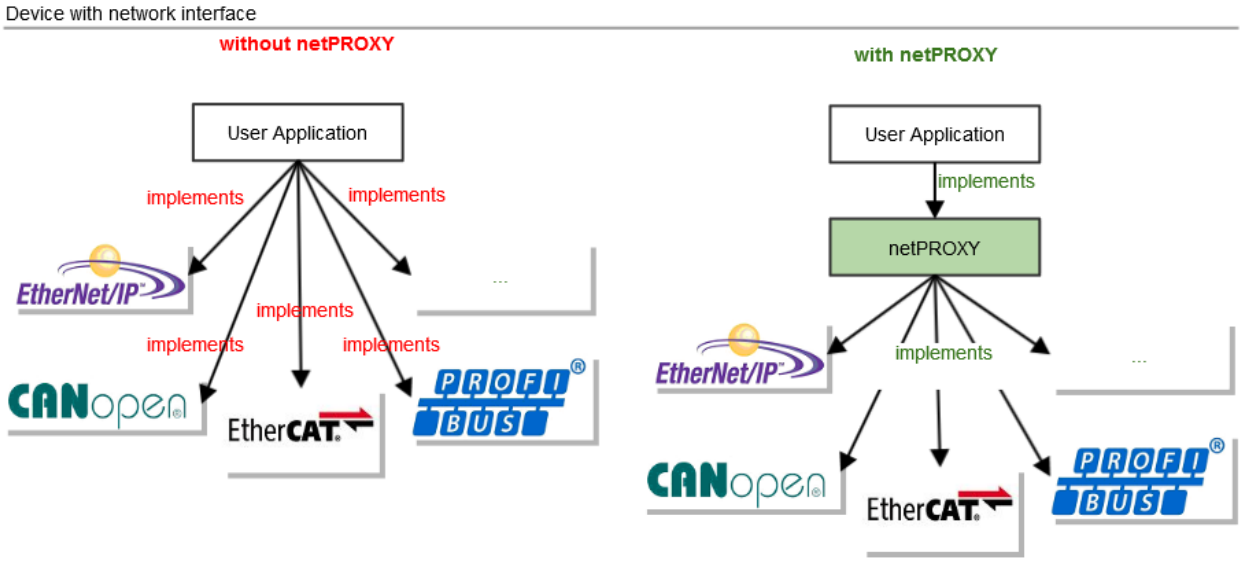

Figure 2: Comparison of uniform netPROXY approach with standard approach in multi-protocol stack projects

2.2 Advantages of using netPROXY for your device

Each network system normally defines some specific services, which should be handled by the user application in specific manner. This requires almost always a deep understanding of the function and flow control in the specific network system. Therefore considerable effort needs to be invested each time making the step towards a new network technology.

The netPROXY technology aims to achieve the synergy during development of the user application on the top of different protocol stacks. In this way the major advantage of the netX automation platform can be taken not only on the hardware side using the netX processor, but also in reuse of the application software with different communication systems.

The abstraction layer on the top of the protocol stack hides the complexity of the individual communication system and provides basic generic services for cyclic and acyclic data exchange, configuration, etc. The user can focus on the own application instead of protocol specific details.

The intelligent tool netX Studio simplifies the application development. It helps to create the network specific mapping and even reuses the user input to generate the following:

- the application source code
- network specific device description file(s)
- webpages for the integrated web server
- documentation..

2.3 Impact on implementation effort

Conventional approach

The industrial communication systems and protocols were created with a focus on a specific automation application field and provide a well-defined object model and services to cover different application profiles.

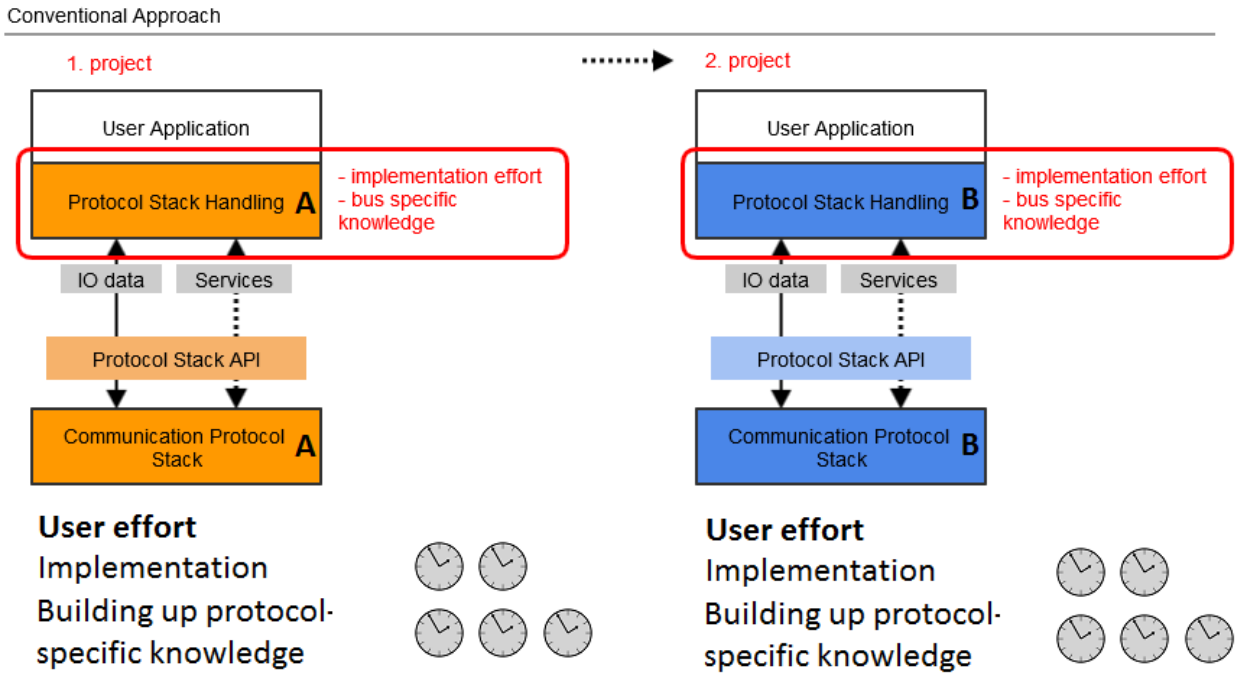

Figure 3: Conventional approach for multi-protocol stack projects

The conventional approach using the dedicated communication protocol stack services results in the considerable implementation effort. This effort is mainly caused by required knowledge about functioning of the specific bus/network system as far as understanding of the related communication stack protocol services provided by the API implementation.

netPROXY approach

netPROXY was developed by Hilscher to simplify the realization of networked devices based on netX controllers.

The advantage of the netPROXY approach is that netPROXY provides the possibility to describe the device application data in a generic way using a simple object model. This can be done just once “by design” covering the specific device application functions and then reused to transform these data to different network protocols.

netPROXY uses the object model not just to manage and access the data, but performs the automatic transformation between netPROXY objects network services. Some of the transformations can be realized in a generic way. Others require configurable transformation rules to maintain high flexibility.

The next advantage of netPROXY is that services are provided in the same way as a data object access. This avoids introduction of the new service API functions with each new extension resulting in a very lean API. This is easy to learn and to use.

Once netPROXY is used to describe the device application data and access the network protocol, the integration of all other different networks is achieved automatically using default mapping rules for each network system.

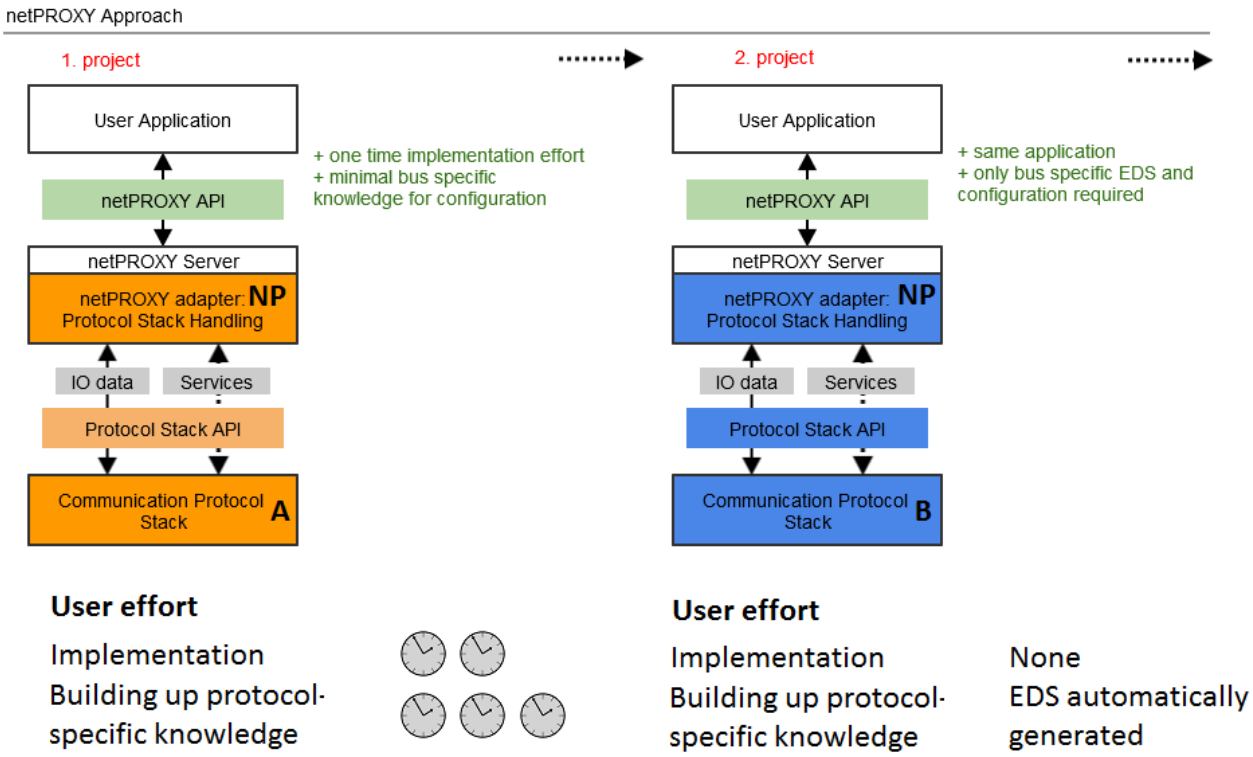

Figure 4: netPROXY approach for multi-protocol stack projects

2.4 netPROXY limitations

netPROXY hides the complexity of the individual communication system and provides basic generic services for cyclic and acyclic data exchange. As a result of this abstraction, the following limitations for applications apply:

- a profile application is not possible,
- product replacement is not possible due to special device behavior,
- an application with high requirements to real-time communication is not possible.

For these requirements, the application has to use the protocol-specific services.

In case, special device behavior is required, verify your requirements with the netPROXY object reference description.

2.5 netX Studio Engineering Tool

netPROXY comes along with an intelligent engineering tool simplifying application development which is called netX Studio.

netX Studio is based on Eclipse and offers XML-based data handling. It simplifies the creation of the device data model and helps with proper configuration of netPROXY. It also includes integrated generators automatically providing:

- a C-header file with the customer-specific objects for application
- the device description file
- webpages for the integrated web server
- documentation
- user management

3 netPROXY object model

netPROXY models devices based on objects. The object is the central part of netPROXY, every data or service which is used to build a device based on netPROXY technology is abstracted with the help of an object. Usually, an object reflects a specific property of the device being modelled.

This chapter explains the relation between objects and their instances and elements.

It also explains how groups and packages of multiple objects can be used for modelling a device in a structured way.

3.1 Objects, instances and elements

This section focuses on objects, their instances and elements. Objects are the central data items within the netPROXY architecture. The following basic facts are valid for all objects:

Definition and identification

- An object is uniquely identified by a number, the Object ID. For rules concerning the Object ID, see *Global address space for Object IDs* [▷ page 12].
- An object has a name.

Description

- The description of an object contains information about the internal structure of the object.

Relation between object and instance

- An object has at least one instance (instance 0). If more than one instance exists, the additional instances are numbered in ascending order (instance 1, 2, …).
- In each instance, you can assign a different set of values to the elements of the object.

Relation between object and element

- Each object has at least one element (element 0). If more than one element exists, the additional elements are numbered in ascending order (instance 1, 2, …).
- Each element has a defined data type The following datatypes are available for:

Data type	Available for	
	Customer-defined elements	System elements
Boolean	Available	Available
Binary	Not available	Available
Integer	Available	Available
Unsigned (integer)	Available	Available
Real	Available	Available
String	Available	Available
Object reference	Not available	Available
Variable binary	Not available	Available

Table 4: Data types and their availability

For more information see the following section *Object element data types* [page 13].

Program access

Program code can access netPROXY objects via the netPROXY Core API. For more information on the functions of the netPROXY Core API, see separate document netPROXY Function interface, Protocol API (Document ID: DOC160205APIXXEN) (reference [1]).

3.2 Global address space for Object IDs

The Object ID is a unique 32bit number uniquely identifying the object. It is related to a special object definition. The following table shows the global address space for Object IDs.

Object ID	Category
0x00000000	NULL
0x00000001 - 0x0FFFFFFF	netPROXY
0x10000000 - 0x1FFFFFFF	System (for instance: Device ID)
0x20000000 - 0x2FFFFFFF	Communication protocol (for instance: Parameters)
0x30000000 - 0x3FFFFFFF	Hilscher objects
0x40000000 - 0x4FFFFFFF	Customer specific objects
0x50000000 - 0xFFFFFFFF	Reserved

Table 5: Global address space for Object IDs

3.3 Object element data types

The following data types are available for elements:

3.3.1 Boolean

This data type can only represent the values "True" and "False".

- The value is "False" if all Bits are zero
- If at least one bit is set the value is "True".
- The size can be 1, 2, 4 or 8 byte.

3.3.2 Integer

A signed integer is represented as a two's complement binary number. The size of the number which can be held by this element depends on the element size.

Currently only the following sizes can be used:

Size (in byte)	Min Value	Max Value	Data type representation
1	-128	127	`int8_t`
2	-32768	32767	`int16_t`
4	- 2147483648	2147483647	`int32_t`
8	-2°63	2^63-1	`int64_t`

Table 6: Integer types

3.3.3 Unsigned

An unsigned integer is a binary number. The size of the number which can be held by this element, depends on the element size.

Currently only the following sizes can be used:

Size (in byte)	Min Value	Max Value	Data type representation
1	0	255	`uint8_t`
2	0	65.535	`uint16_t`
4	0	4.294.967.295	`uint32_t`
8	0	2^64-1	`uint64_t`

Table 7: Unsigned integer types

3.3.4 Real

Floating point number, the representation is according to IEEE 754.

Currently only the following sizes can be used:

Size (in byte)	Data type representation C/C++
4	`float`
8	`double`

Table 8: Real types

3.3.5 String

All strings are zero- terminated. The maximum string length, which the element can hold must be specified in the range between 1 and 65535 including the termination byte. The following string formats are known:

Printable ASCII string

Only the following chars are allowed

- `32` (`20h`) – `126` (`7Eh`)

ASCII Decimal Number

Only the following chars are allowed

- `48` (`30h`) – `57` (`39h`)

ASCII Time

This format uses the standardized format ISO 8601:2006-09 / EN 28601 (extended format)

`YYYY`= four-digit year

`MM`= two-digit month (01=January, etc.)

`DD`= two-digit day of month (01 through 31)

`hh`= two digits of hour (00 through 23) (am/pm NOT allowed)

`mm`= two digits of minute (00 through 59)

`ss`= two digits of second (00 through 59)

TZD= time zone designator (Z or +hh:mm or -hh:mm)

The hh, mm and `ss` can also have an optional fraction separated by a comma.

Example for a complete date plus hours, minutes, seconds and a decimal fraction of a second

- `YYYY-MM-DDThh:mm:ss,sTZD` (e.g. `1997-07-16T19:20:30,45+01:00`)

ASCII Version

Prefix= Version identifier ('V' = Released, 'T' = Test, or any upper case letter)

Major= Major version number 1-5 digits (0 to 2^16-1)

Minor= Minor version number 1-5 digits (0 to 2^16-1)

Revision= Revision version number 1-5 digits (0 to 2^16-1)

Build= Build number 1-10 digits (0 to 2^32-1)

All parts, without the prefix, will split with a point (2Eh). The build part is optional and can be omitted, for example by hardware relevant version information.

Any text can be added to the version information and must be separated by a single space (20h) character.

Examples for complete version information:

- PrefixMajor.Minor.Revision ("V1.0.0", "T1.0.0 debug code")
- PrefixMajor.Minor.Revision.Build ("V1.2.3.4", "V1.2.3.4 null series")

3.4 Addressing an element of an object within a device

Four numbers address a specific element of an object within a device:

- Group ID
- Object ID
- Instance number
- Element number

Range of values

ID or number	Description
Group ID	netPROXY group ID to address the group of objects (collection of objects) within a device. Often, a device contains only one group. In this case, the group ID is always 0.
Object ID	netPROXY object ID to address the object. The netPROXY object 0x00001000 has a list of all objects within a device including the group ID the objects belongs to.
Instance number	netPROXY instance number addresses instance of an object, because an object can exist multiple times. Value 0 addresses the first instance of an object. Value 1 addresses the second instance (if available) of an object. The N instances of an object are addressed from 0, 1, …, N-1.
Element number	netPROXY element number addresses a part of an object Value 0 addresses the first element of an object. Value 1 addresses the second element (if available) of an object. The N elements of an object are addressed from 0, 1, …, N-1.

Table 9: Range of values to address an object element

3.5 Access to an object from the application

The application can access the object via two API functions, namely `read()` and `write()`.

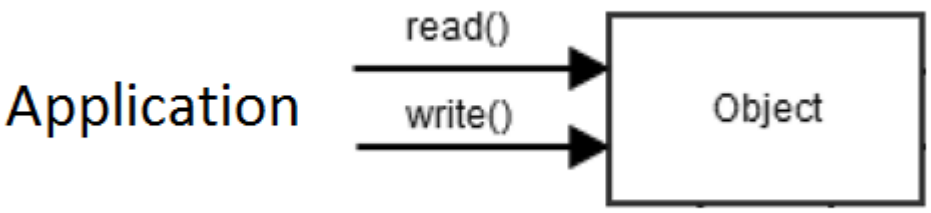

Figure 5: Access to an object from the application

When accessing an object via `read()` or `write()`, it has to be referenced by the following four specifications:

Field Name	Elementary Data Type	Description
GroupID	UINT32	netPROXY Group
ObjectID	UINT32	netPROXY Object ID
Instance	UINT32	netPROXY Instance
Element	UINT32	netPROXY Element

Table 10: Referencing an Object

4 Device structure

There are two types of structures possible to build a netPROXY based device.

4.1 netPROXY device with separate processors and dedicated host application

These devices use embedded modules (e.g. NIC52-RE) with a netPROXY firmware. Typically, such a device has two CPUs which are connected with each other via the host interface. All communication between the host CPU and the netX CPU of the netPROXY device runs over the host interface.

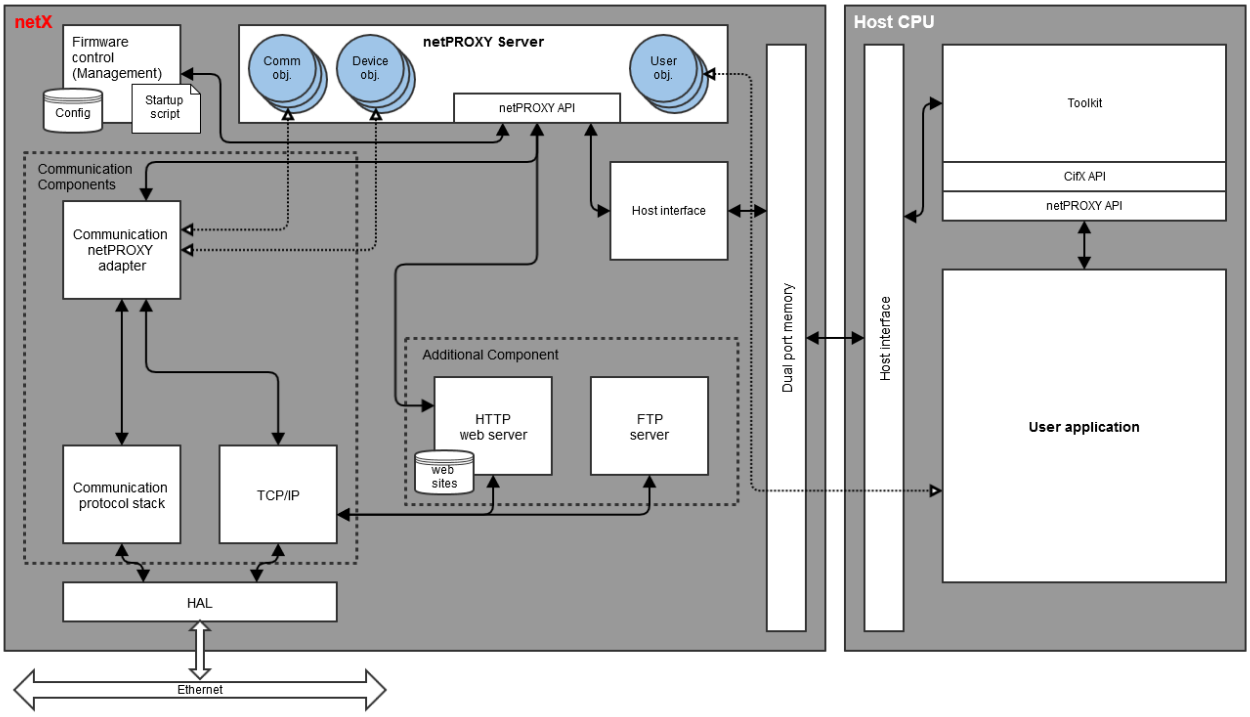

Figure 6: netPROXY device with dedicated host application

5 Appendix

5.1 Legal notes

Copyright

Important notes

Utmost care was/is given in the preparation of the documentation at hand consisting of a user's manual, operating manual and any other document type and accompanying texts. However, errors cannot be ruled out. Therefore, we cannot assume any guarantee or legal responsibility for erroneous information or liability of any kind. You are hereby made aware that descriptions found in the user's manual, the accompanying texts and the documentation neither represent a guarantee nor any indication on proper use as stipulated in the agreement or a promised attribute. It cannot be ruled out that the user's manual, the accompanying texts and the documentation do not completely match the described attributes, standards or any other data for the delivered product. A warranty or guarantee with respect to the correctness or accuracy of the information is not assumed.

We reserve the right to modify our products and the specifications for such as well as the corresponding documentation in the form of a user's manual, operating manual and/or any other document types and accompanying texts at any time and without notice without being required to notify of said modification. Changes shall be taken into account in future manuals and do not represent an obligation of any kind, in particular there shall be no right to have delivered documents revised. The manual delivered with the product shall apply.

Under no circumstances shall Hilscher Gesellschaft für Systemautomation mbH be liable for direct, indirect, ancillary or subsequent damage, or for any loss of income, which may arise after use of the information contained herein.

Liability disclaimer

The hardware and/or software was created and tested by Hilscher Gesellschaft für Systemautomation mbH with utmost care and is made available as is. No warranty can be assumed for the performance or flawlessness of the hardware and/or software under all application conditions and scenarios and the work results achieved by the user when using the hardware and/or software. Liability for any damage that may have occurred as a result of using the hardware and/or software or the corresponding documents shall be limited to an event involving willful intent or a grossly negligent violation of a fundamental contractual obligation. However, the right to assert damages due to a violation of a fundamental contractual obligation shall be limited to contract-typical foreseeable damage.

It is hereby expressly agreed upon in particular that any use or utilization of the hardware and/or software in connection with

- Flight control systems in aviation and aerospace;
- Nuclear fusion processes in nuclear power plants;
- Medical devices used for life support and
- Vehicle control systems used in passenger transport

shall be excluded. Use of the hardware and/or software in any of the following areas is strictly prohibited:

- For military purposes or in weaponry;
- For designing, engineering, maintaining or operating nuclear systems;
- In flight safety systems, aviation and flight telecommunications systems;
- In life-support systems;
- In systems in which any malfunction in the hardware and/or software may result in physical injuries or fatalities.

You are hereby made aware that the hardware and/or software was not created for use in hazardous environments, which require fail-safe control mechanisms. Use of the hardware and/or software in this kind of environment shall be at your own risk; any liability for damage or loss due to impermissible use shall be excluded.

Warranty

Hilscher Gesellschaft für Systemautomation mbH hereby guarantees that the software shall run without errors in accordance with the requirements listed in the specifications and that there were no defects on the date of acceptance. The warranty period shall be 12 months commencing as of the date of acceptance or purchase (with express declaration or implied, by customer's conclusive behavior, e.g. putting into operation permanently).

The warranty obligation for equipment (hardware) we produce is 36 months, calculated as of the date of delivery ex works. The aforementioned provisions shall not apply if longer warranty periods are mandatory by law pursuant to Section 438 (1.2) BGB, Section 479 (1) BGB and Section 634a (1) BGB [Bürgerliches Gesetzbuch; German Civil Code] If, despite of all due care taken, the delivered product should have a defect, which already

existed at the time of the transfer of risk, it shall be at our discretion to either repair the product or to deliver a replacement product, subject to timely notification of defect.

The warranty obligation shall not apply if the notification of defect is not asserted promptly, if the purchaser or third party has tampered with the products, if the defect is the result of natural wear, was caused by unfavorable operating conditions or is due to violations against our operating regulations or against rules of good electrical engineering practice, or if our request to return the defective object is not promptly complied with.

Costs of support, maintenance, customization and product care

Please be advised that any subsequent improvement shall only be free of charge if a defect is found. Any form of technical support, maintenance and customization is not a warranty service, but instead shall be charged extra.

Additional guarantees

Although the hardware and software was developed and tested in-depth with greatest care, Hilscher Gesellschaft für Systemautomation mbH shall not assume any guarantee for the suitability thereof for any purpose that was not confirmed in writing. No guarantee can be granted whereby the hardware and software satisfies your requirements, or the use of the hardware and/or software is uninterruptable or the hardware and/or software is fault-free.

It cannot be guaranteed that patents and/or ownership privileges have not been infringed upon or violated or that the products are free from third-party influence. No additional guarantees or promises shall be made as to whether the product is market current, free from deficiency in title, or can be integrated or is usable for specific purposes, unless such guarantees or promises are required under existing law and cannot be restricted.

Confidentiality

The customer hereby expressly acknowledges that this document contains trade secrets, information protected by copyright and other patent and ownership privileges as well as any related rights of Hilscher Gesellschaft für Systemautomation mbH. The customer agrees to treat as confidential all of the information made available to customer by Hilscher Gesellschaft für Systemautomation mbH and rights, which were disclosed by Hilscher Gesellschaft für Systemautomation mbH and that were made accessible as well as the terms and conditions of this agreement itself.

The parties hereby agree to one another that the information that each party receives from the other party respectively is and shall remain the intellectual property of said other party, unless provided for otherwise in a contractual agreement.

The customer must not allow any third party to become knowledgeable of this expertise and shall only provide knowledge thereof to authorized users as appropriate and necessary. Companies associated with the customer shall not be deemed third parties. The customer must obligate authorized

users to confidentiality. The customer should only use the confidential information in connection with the performances specified in this agreement.

The customer must not use this confidential information to his own advantage or for his own purposes or rather to the advantage or for the purpose of a third party, nor must it be used for commercial purposes and this confidential information must only be used to the extent provided for in this agreement or otherwise to the extent as expressly authorized by the disclosing party in written form. The customer has the right, subject to the obligation to confidentiality, to disclose the terms and conditions of this agreement directly to his legal and financial consultants as would be required for the customer's normal business operation.

Export provisions

The delivered product (including technical data) is subject to the legal export and/or import laws as well as any associated regulations of various countries, especially such laws applicable in Germany and in the United States. The products / hardware / software must not be exported into such countries for which export is prohibited under US American export control laws and its supplementary provisions. You hereby agree to strictly follow the regulations and to yourself be responsible for observing them. You are hereby made aware that you may be required to obtain governmental approval to export, reexport or import the product.

Terms and conditions

Please read the notes about additional legal aspects on our netIOT web site under http://www.netiot.com/netiot/netiot-edge/terms-and-conditions/.

List of figures

List of tables

Contacts

HEADQUARTERS

Germany

Hilscher Gesellschaft für
Systemautomation mbH
Rheinstrasse 15
65795 Hattersheim
Phone: +49 (0) 6190 9907-0
Fax: +49 (0) 6190 9907-50
E-mail: info@hilscher.com

Support

Phone: +49 (0) 6190 9907-99
E-mail: de.support@hilscher.com

SUBSIDIARIES

China

Hilscher Systemautomation (Shanghai) Co. Ltd.
200010 Shanghai
Phone: +86 (0) 21-6355-5161
E-mail: info@hilscher.cn

Support

Phone: +86 (0) 21-6355-5161
E-mail: cn.support@hilscher.com

France

Hilscher France S.a.r.l.
69500 Bron
Phone: +33 (0) 4 72 37 98 40
E-mail: info@hilscher.fr

Support

Phone: +33 (0) 4 72 37 98 40
E-mail: fr.support@hilscher.com

India

Hilscher India Pvt. Ltd.
Pune, Delhi, Mumbai
Phone: +91 8888 750 777
E-mail: info@hilscher.in

Italy

Hilscher Italia S.r.l.
20090 Vimodrone (MI)
Phone: +39 02 25007068
E-mail: info@hilscher.it

Support

Phone: +39 02 25007068
E-mail: it.support@hilscher.com

Japan

Hilscher Japan KK
Tokyo, 160-0022
Phone: +81 (0) 3-5362-0521
E-mail: info@hilscher.jp

Support

Phone: +81 (0) 3-5362-0521
E-mail: jp.support@hilscher.com

Korea

Hilscher Korea Inc.
Seongnam, Gyeonggi, 463-400
Phone: +82 (0) 31-789-3715
E-mail: info@hilscher.kr

Switzerland

Hilscher Swiss GmbH
4500 Solothurn
Phone: +41 (0) 32 623 6633
E-mail: info@hilscher.ch

Support

Phone: +49 (0) 6190 9907-99
E-mail: ch.support@hilscher.com

USA

Hilscher North America, Inc.
Lisle, IL 60532
Phone: +1 630-505-5301
E-mail: info@hilscher.us

Support

Phone: +1 630-505-5301
E-mail: us.support@hilscher.com