SIMATIC Rail

Integration of Subsystems on PROFINET

First edition: March 1, 2007
Date: Mai 06, 2009
Document version: 1.1
No. of pages: 76
Document code: =!H.6611.KZ17&EEC025
Certification

The products and systems listed in this documentation are manufactured and sold on the basis of use of a quality management system to DIN ISO 9001 that is certified by DQS. The DQS certificate is recognized in all EQ Net countries.

Rights

SIMATIC® and SIBAS® are registered trademark of Siemens AG. It is possible that other designations in this documentation/information are registered trademarks whose use by third parties for their own purposes may violate the owners’ rights.
### Typographical conventions

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Courier New</strong></td>
<td>The <em>Courier New</em> font identifies program text, user inputs and text that appears on the monitor.</td>
</tr>
<tr>
<td><strong>UPPER CASE</strong></td>
<td>Directory names, file names, names of applications and acronyms are printed in upper case. Exception: like in the source text, names of header files are specified in lower case.</td>
</tr>
<tr>
<td><strong>Italics</strong></td>
<td>Words highlighted in <em>italics</em> designate names of variables, constants, functions, commands, classes and structures.</td>
</tr>
<tr>
<td><strong>Bold</strong></td>
<td>Words highlighted in <strong>bold</strong> designate buttons in the software.</td>
</tr>
<tr>
<td>File</td>
<td>Open</td>
</tr>
<tr>
<td><strong>[Key designation]</strong></td>
<td>Italic in square brackets are used for keys on the keyboard. Example: &quot;Press [Ctrl], [Esc] or [F1].&quot;</td>
</tr>
<tr>
<td><strong>[Key1]+[Key2]</strong></td>
<td>Key designations linked with a &quot;+&quot; stand for pressing both keys at the same time.</td>
</tr>
<tr>
<td><strong>[]</strong></td>
<td>In text or syntax examples, square brackets refer to optional elements. Square brackets are omitted on input. In the text, numbers in square brackets refer to literature references.</td>
</tr>
<tr>
<td><strong>&lt;&gt;</strong></td>
<td>In text or syntax examples, pointed brackets refer to a variable string. This is where a string must be entered. The pointed brackets are then omitted.</td>
</tr>
<tr>
<td><strong>...</strong></td>
<td>In text or syntax examples, three dots refer to code that has been omitted.</td>
</tr>
</tbody>
</table>
Contents

1 Introduction ................................................................................................................. 8
  1.1 Content of the document .................................................................................. 8
  1.2 Requirements for the reader ....................................................................... 8
  1.3 Starting situation ............................................................................................ 8
  1.4 Goals and general conditions ..................................................................... 8

2 System Overview ........................................................................................................... 9
  2.1 Sibas PN .................................................................................................................. 9
    2.1.1 Device concept .............................................................................................. 9
  2.2 PROFINET .............................................................................................................. 11
    2.2.1 PROFIBUS user organization .................................................................. 11
    2.2.2 PROFINET overview ................................................................................. 13
    2.2.3 PROFINET characteristics ....................................................................... 14
      2.2.3.1 Protocols on the vehicle bus ............................................................ 15
    2.2.4 PROFINET IO ............................................................................................... 18
      2.2.4.1 Overview ............................................................................................. 18
      2.2.4.2 Device model ....................................................................................... 21
      2.2.4.3 Device description (GSD) .................................................................. 22
      2.2.4.4 Project planning .................................................................................. 23
      2.2.4.5 System start-up .................................................................................... 24
      2.2.4.6 Cyclic communication ....................................................................... 25
      2.2.4.7 Noncyclic communication .................................................................. 25
    2.2.5 Network structures ...................................................................................... 26
    2.2.6 Switch ............................................................................................................. 26
    2.2.7 External power supply ................................................................................ 27
    2.2.8 PROFINET interfaces for subsystems ...................................................... 28
      2.2.8.1 ERTEC ASIC ......................................................................................... 29
      2.2.8.2 Development Kit ERTEC ................................................................. 30
      2.2.8.3 Development Kit for PROFINET IO ................................................. 30
      2.2.8.4 PROFINET PC/104 .............................................................................. 31
      2.2.8.5 Ordering information / Technical support .......................................... 32
    2.2.9 Standard Ethernet integration .................................................................... 32

2.3 Additional requirements for subsystems ............................................................. 33
  2.3.1 Functional interface ...................................................................................... 33
2.3.2 Device behavior ................................................................. 33
2.3.3 Supervision times ............................................................... 33
2.4 Lines and connectors .................................................................. 35
2.5 Local services ....................................................................................................... 35
  2.5.1 Definition ............................................................................. 35
  2.5.2 Local download service ......................................................... 37
  2.5.3 Local configuration service .................................................... 44
  2.5.4 Local identification service .................................................... 45
  2.5.5 Time synchronization service .................................................. 45
  2.5.6 Self-test ................................................................................ 46
  2.5.7 Web server for device-specific diagnostics and service .......... 46
2.6 Diagnostics ........................................................................................................ 50
  2.6.1 Operational signaling ............................................................. 51
  2.6.2 Workshop diagnostics ............................................................. 51
  2.6.3 Operational and functional status ............................................. 51
    2.6.3.1 Definition of the functional status ....................................... 52
    2.6.3.2 Definition of operational status ........................................... 53
    2.6.3.2.1 General System Mode .................................................... 53
    2.6.3.2.2 Redundancy Mode ........................................................ 55
    2.6.3.2.3 Test Service Mode ........................................................ 55
    2.6.3.2.4 State machine ................................................................. 56
  2.6.4 Diagnostics example ............................................................... 58
    2.6.4.1 Scenario 1 – Operational and functional status for an open door..... 59
    2.6.4.2 Scenario 2 – Operational and functional status for a defective door 59
    2.6.4.3 Scenario 3 – Operational and functional status for an isolated door 59
  2.6.5 Diagnostic truth ............................................................................. 60
  2.6.6 Diagnostic generation in the subsystem ........................................ 62
    2.6.6.1 Functional status ............................................................... 62
    2.6.6.2 Operational status ............................................................... 62
    2.6.6.3 Operational signaling .......................................................... 62
    2.6.6.4 Diagnostic buffer for workshop diagnostic messages ............... 63
    2.6.6.5 Diagnostic buffer for log data ............................................... 63
    2.6.6.6 Initialization data for diagnostic messages ............................. 63
    2.6.6.7 Layout of workshop diagnostic messages (record) .................... 65
    2.6.6.8 Description file .................................................................. 68
  2.6.7 CSV file ........................................................................................ 68
    2.6.7.1 Introduction ...................................................................... 68
    2.6.7.2 CSV format ...................................................................... 68
List of figures

Figure 2-1: Schematic diagram of Sibas PN with PROFINET as vehicle bus (as ring topology) ................................................................. 10
Figure 2-2: Integration of switches into the Ethernet interfaces of the stations ................................................................. 14
Figure 2-3: Partitioning of the bandwidth ........................................................................................................................................... 15
Figure 2-4: Protocols and applications for PROFINET ........................................................................................................................... 17
Figure 2-5: Device architecture of PROFINET IO ................................................................................................................................. 19
Figure 2-6: Communication relationships with PROFINET IO ............................................................................................................. 20
Figure 2-7: Device model with PROFINET IO ....................................................................................................................................... 21
Figure 2-8: Example of a Device Ident Number .................................................................................................................................. 23
Figure 2-9: From configuration to data exchange ................................................................................................................................ 24
Figure 2-10: Cyclic transmission of process data ................................................................................................................................. 25
Figure 2-11: Simple ring ............................................................................................................................................................................. 26
Figure 2-12: Principle of external power supply for the switches (communication interface) ........................................................................... 27
Figure 2-13: View for consistent (local) services.................................................................................................................................. 36
Figure 2-14: Device update with status validity check ........................................................................................................................... 38
Figure 2-15: Minimum FTP server functionality according to RFC 959 .................................................................................................. 38
Figure 2-16: Download Object ................................................................................................................................................................. 40
Figure 2-17: Archive .................................................................................................................................................................................... 40
Figure 18 Operational status and visualization ......................................................................................................................................... 51
Figure 19: Relationship between function and operational status ........................................................................................................ 52
Figure 20: Operational status of functional unit ....................................................................................................................................... 53
Figure 21: State diagram for operational status ....................................................................................................................................... 56
Figure 22 Functions, functional units, operational status and functional status for the example of door control ................................................................................................................................. 58
Figure 24: Diagnostic generation ................................................................................................................................................................. 62
Figure 25: Layout of a CSV file (lines, columns, cells) .................................................................................................................................. 68
Figure 3-1: PNO certification for a PROFINET IO device ......................................................................................................................... 71
Figure 4-1: Test setup with PC module and IO device using IM 104 RT TS ................................................................................................. 72
1 Introduction

1.1 Content of the document

This document describes the integration of subsystems into the SIMATIC Rail project. At the time of creation of this document, the components of SIMATIC Rail were still under development so that changes and additions cannot be excluded. The document contains a summary of the currently available knowledge. Additional versions may follow.

The control system structures shown are meant as an example for illustration. Specific definitions will be made in the form of specifications for the vehicle projects.

1.2 Requirements for the reader

Basically, the reader does not require any special knowledge to understand this document, but knowledge of SIMATIC and PROFINET is helpful.

1.3 Starting situation

The integration platform for components of the control system is the PROFINET communication system of the PROFIBUS user organization (PNO) [1].

1.4 Goals and general conditions

SIMATIC Rail is the project name for the development of products for Sibas PN (Siemens Railway Automation System, PROFINET) on the basis of the SIMATIC industrial automation system from SIEMENS.

There is explicitly no compatibility with previous systems, such as Sibas 32.

The connectivity to PROFINET is implemented by means of industrial automation products from SIEMENS A&D.

This is done to achieve the following goals:

- Integration of information technologies (Internet, http, ftp...)
- Use of widely used hardware and software as well as open standards
- Participation in the high level of innovation of industrial automation
- Support by the user organization (PNO)
2 System Overview

2.1 Sibas PN

2.1.1 Device concept

Sibas PN comprises the control and communication systems that are required for the automation of rail vehicles, including the integration of subsystems. This also includes the necessary configuration tools.

The integration platform is PROFINET, which is used as a vehicle bus to ensure essentially the integration of the following control systems:

- Vehicle control system (Control System, CS)
- Traction control system (SRR)
- Distributed peripheral units (DP)
- WTB train bus node (PN-WTB Link)
- HMI system (Human Machine Interface)
- Control computer for various subsystems (e.g. door, brake, auxiliaries, passenger information, ...)
- PROFINET Switch (as transition from ring to line, star or tree topology)
There are additional components that are used outside the vehicle or used only temporarily. These include, for example, test and commissioning tools.

Figure 2-1: Schematic diagram of Sibas PN with PROFINET as vehicle bus (as ring topology)
2.2 PROFINET

2.2.1 PROFIBUS user organization

An open technology requires a vendor-independent institution as a working platform to ensure its maintenance, further development and distribution in the market. To support these goals, the PROFIBUS Nutzerorganisation e.V. (PNO) was founded in 1989 as a non-profit user organization that represents the interests of vendors, users and institutes.

The PNO is a member of the international umbrella organization PROFIBUS/PROFINET international (PI) that was founded in 1995. Comprising 24 regional user organizations (Regional PROFIBUS/PROFINET Associations, RPA) and about 1,300 members, including in the USA, China and Japan, PI is the world's largest community of interests in the field of industrial communication. The RPAs organize the participation in exhibitions and information events and also ensure that new requirements of the markets are taken into account for further developments.

Tasks

Major tasks of PI include:

- Maintenance and further development of the PROFIBUS and PROFINET technologies.
- Support of the worldwide distribution of the technologies.
- Investment protection for users and vendors through influence on the standardization process.
- Representation of the interests of members toward standardization bodies and associations.
- Worldwide technical support through Competence and Training Centers http://www.profibus.com/pn/support/pccs/
- Quality assurance through device certification

Membership

The membership is organized by regions.

It is open to all companies, associations, institutes and individuals that/who are interested in participating actively in the development and distribution of the PROFIBUS and PROFINET technologies. The coordinated activities of the members that can be very different and come from different industries generates considerable synergies and a broad-based sharing of information. This results in innovative solutions, an effective exploitation of resources and ultimately in competitive advantages in the market.
Organization of technological development

The activities of technological development are guided by the Advisory Board. The developer teams are organized in Technical Committees (TCs) with more than 50 fixed Working Groups (WGs). This is complemented by a varying number of ad hoc WGs that cover specific subjects for a limited period of time. The WGs develop new specifications and profiles, work on quality assurance and standardization, participate in standardization bodies and carry out effective marketing activities (exhibitions, presentations) to distribute the technologies. The head office (PI Support Center) coordinates all occurring activities. More than 500 experts in the Working Group are actively involved in the development and distribution of the technology.

The subdivision into more than 50 WGs enables a highly efficient development work with a focus on specific subjects and industries.

Every member is entitled to participate in the Working Groups and thus to exert influence on the further development. All new work results are presented to the members for commenting before being released by the Advisory Board.

Device profile

The Working Group WG19 "Train Applications" was established to create profiles of components for railway automation. This WG defines general and subsystem-specific characteristics for device profiles. As a first step, a profile for the connection of the train bus according to UIC556 specification to PROFINET is being created. Additional device profiles will follow.

A profile enables the proper interaction of devices from different vendors on PROFINET for a specific application. Profiles are defined by specialists from (different) companies with the goal of committing all device vendors to use this profile definition.

PROFINET profiles are:

- vendor-independent definitions for consistent device performance
- documented in PROFINET guidelines
- optionally usable

PROFINET profiles describe:

- device classes, e.g. drives
- operating modes, e.g. redundancy
- application-specific requirements
- communication processes and minimum requirements for devices
- the data content of the files to be exchanged (meaning and format)

Profiles enable:
interoperability at the application level (consistent access to the exchanged data without changing the user software)

- simplification of tenders (vendor-independent)
- interchangeability of the devices used

Technical support

PI maintains over 30 Competence and Training Centers worldwide and has accredited 7 test labs for certifications. These facilities provide consulting, training and support in different ways for users and vendors and perform tests for device certification. As a PI facility, they offer their services in a vendor-independent manner within the framework of an agreed set of rules. They are regularly reviewed for their suitability by an accreditation process that is tailed for each specific group. Current addresses can be found on the Web site of the organization [1] (http://www.profibus.com/pn/support/pccs/).

2.2.2 PROFINET overview

PROFINET is the innovative automation standard of the PROFIBUS user organization for the implementation of an integrated and consistent automation solution based on Industrial Ethernet.

PROFINET enables the component-based integration of distributed field devices as well as time-critical applications into the Ethernet communication in the same manner as distributed automation systems.

The possibility to leverage IT standards in addition to the transmission of field data permits the implementation of an integrated infrastructure that offers IT openness and realtime capabilities. PROFINET is the implementation of a realtime Ethernet (RT Ethernet) for automation systems that can also be used in railway applications.

PROFINET is used to implement the vehicle bus for Sibas PN that connects control computers within a railway vehicle.

PROFINET is a Switched Ethernet with an extension for the realtime transmission of field data. These RT capabilities are a feature of PROFINET and are not provided by the common Ethernet. These capabilities are achieved by the PROFINET protocol stack and a purpose-designed Ethernet Controller ERTEC (Enhanced Realtime Ethernet Controller), which can be used to implement both the switch functionality and the communication connectivity of control computers.

The network components for Switched Ethernet consist of the active switches, PROFINET interface modules and the associated lines and connectors.

The components are taken from the product range of Siemens A&D.
PROFINET is standardized by IEC 61158 and IEC 61784. For railway applications, standardization in the IEC61375 is in preparation.

An overview of PROFINET is provided in the PNO document “PROFINET Technology and Application – System Description”:


and in the documentation for SIMATIC NET from SIEMENS A&D:


The key characteristics of PROFINET are described below.

2.2.3 PROFINET characteristics

PROFINET is based on Fast Ethernet with the characteristics 100Mbps, full-duplex and switching. Within the context of Sibas PN, signal transmission according to 100Base-TX occurs via symmetrical copper wire (Twisted Pair).

All controllers are connected via active network components. Network components with PROFINET are switches. Communications between the switches is collision-free.

In order to ensure determinism and clock-synchronism to support the realtime capabilities, the realtime network is made up of highly synchronous switches. They ensure the input and output as well as the transmission of data over the network at exactly the right times.

In order to minimize the number of components and the costs, the switch is integrated into the Ethernet interface and thus into the station (“Teilnehmer”).

Figure 2-2: Integration of switches into the Ethernet interfaces of the stations

For the transmission of realtime and non-realtime data, the Ethernet bandwidth, which is available for spontaneous communication with the standard Ethernet, is partitioned into separate time zones for these data types.

Note: The PROFINET mechanisms or characteristics described below represent the current state of the PROFINET implementation. Due to functional changes or enhancements, these PROFINET mechanisms or characteristics may vary from the present documentation in the future.
2.2.3.1 Protocols

In order to subdivide the available Ethernet transmission bandwidth into ranges for realtime and non-realtime data, the bandwidth is organized in time cycles. All connected switches are synchronized to these cycles. This is achieved by an automated mechanism that detects all time parameters of the line and thus permits the highly accurate synchronization of all switches to the start of the cycle. Cycle synchronization occurs on the basis of synchronization telegrams and the purpose-designed switch hardware (ERTEC).

The subdivision into cycles makes it possible to reserve a certain portion of the bandwidth for the transmission of realtime data. The remaining bandwidth is available for non-realtime data until the start of the next cycle.

For a better understanding, the various terms used in this context will be explained below.

Realtime data are also called RT data. These are field or process data that are transmitted cyclically at fixed times. The transmission is independent of whether the values of these data have changed or not. This means that a current image of the data is always transmitted. PROFINET provides the following protocols for this deterministic data transmission:

- IRT (Isochronous Realtime)
- RT (Realtime)

Non-realtime data (NRT data) are handled by standard protocols like they are used for Ethernet. The transmission of NRT data is usually spontaneous. Protocols used are TCP/UDP/IP.

![Partitioning of the bandwidth](image)

Figure 2-3: Partitioning of the bandwidth

A PROFINET communication cycle contains one area each for cycle synchronization, deterministic communication and open communication. The deterministic channel transports only the cyclic IRT telegrams, with the RT and TCP/IP telegrams being transported in the open channel.
To avoid any restriction of openness for NRT telegrams, TCP/IP telegrams with maximum length must not be constrained. This results in a limitation of the minimum cycle time (a telegram with a length of 1440 bytes takes about 125 µs).

Then again, TCP/IP telegrams of maximum length must not delay the start of the cycle. Therefore, the PROFINET switches delay TCP/IP frames that are pending shortly before the start of the cycle and during the IRT phase until the IRT phase has ended.

The basic cycle time can be set for the entire system and will be 1 ms or 2 ms for TS applications. The transmit and receive cycles of the individual controllers can be configured as multiples of the basic cycle time (e.g. 4, 8, 16 ms).

The time reserved for IRT is determined during the configuration of a system and is based on the number of controllers that exchange IRT data and their cyclic data volume (data volume and cycle time). This bandwidth reservation is strictly observed at runtime. This is possible since only scheduled communication takes place during this phase. The use of the ERTEC switch excludes any impact of the communication volume during the open communication phase.

Realtime communication uses the VLANTag according to IEEE802.1Q for the preferred forwarding of the RT frames. This permits the setting of priority levels. PROFINET RT runs priority level six.

TCP/IP and UDP/IP and RT communication can be implemented with commercially available Ethernet controllers.

The length of RT and IRT telegrams can be between 64 and 1440 bytes.

All IP-based protocols can be used for NRT communication. The bandwidth of Fast Ethernet is limited by the bandwidth reservation for synchronization and deterministic communication.
Figure 2-4: Protocols and applications for PROFINET

Figure 2-4 shows a summary of the protocols or protocol layers that are used for PROFINET and the associated application cases. IT and PROFINET applications can communicate via TCP/UDP/IP.

Figure 2-5: use cases for PROFINET

PROFINET RT allows high-performance realtime-communication as well as unlimited TCP/IP communication.
2.2.3.2 Protocols on the vehicle bus

In course of developing SIMATIC Rail several use cases were discussed in order to gain the requirements of different components (e.g. door units, brakes …) according to the communication system.

Following criterias were observed:

- minimum cycletime
  - cyclic data transmission each 32ms or more
- jitter
  - Requirements according to jitter, which could evolve from TCP/IP-next to RT-communication and processing time of the switches in worst case
- determinism
  - Adequate characteristics according to determinism will be guaranteed by PROFINET RT communication proposal in the SIMATIC RAIL engineering system and by system affected prioritisation of RT-communication.

Note: In case of a non PROFINET-communication component being used, a frame priority tag (according to IEEE802.1q) with a value of 6 or grater must not be used.

Depending on these use cases PROFINET RT achieves the needs of a vehicle bus in SIBAS PN.

The “PN PC/104” (see Fehler! Verweisquelle konnte nicht gefunden werden.) is thereby recommended as the default communication subassembly for Subsystems with PROFINET communication.

2.2.4 PROFINET IO

The application model for the interfacing of distributed field devices, which is also used for Sibas PN, is "PROFINET IO". This uses the IO view of PROFIBUS DP, in which the useful data of the field devices are cyclically read into the controller, processed and sent back to the field devices [1].

2.2.4.1 Overview

The master-slave method that is known from PROFIBUS DP has been converted into a provider-consumer model for PROFINET. A provider generates and sends data that the consumer receives and processes.

All devices on the Ethernet have equal rights from a communication perspective. However, the assignment of field devices to a central controller is defined via the configuration. PROFINET IO consequently partitions the controllers into IO controllers and IO devices. IO controllers are typically controllers (central vehicle control unit). The interfaces of such controllers are not standardized by the PNO in a vendor-independent manner, and in particular the loading protocols are different.
The interface for IO devices has been standardized via the PNO PROFINET IO and GSD). This means that controllers from different vendors can communicate with IO devices. IO devices are logically assigned to an IO controller during the configuration.

In addition, PNO profiles can be used to define application profiles for devices.

In TS projects, a SIMATIC-based controller is used as IO controller, whereas all other devices act as IO devices (this includes e.g. external switches that are not assigned to any device).

In addition to IO controllers and IO devices, there are devices with SIMATIC commissioning and diagnostic functionality that are only temporarily connected to PROFINET. These devices are summarized under the term of IO Supervisor.

Figure 2-6: Device architecture of PROFINET IO

The diagnostics with IO Supervisor mentioned in the illustration is limited to the standard SIMATIC diagnostic mechanisms. Additional diagnostic mechanisms that are defined on the SIMATIC Rail project may require additional tools.

The exchange of data between the devices uses the realtime and non-realtime channels, depending on the data type:

- cyclic application data over the realtime channel
- event-driven alarms over the realtime channel
- parameterization and configuration as well as reading of diagnostic information over the open channel based on UDP/IP.
CR: Communication Relation
AR: Application Relation

Figure 2-7: Communication relationships with PROFINET IO

The communication between the devices is described by different application and communication relationships. During start-up, an application relationship (IO-AR) between the IO controller and the IO device is established by means of the so-called Context Management (CM). An application relationship includes several communication relationships (CRs) on which configuration data, application data and alarms can be transmitted. The IO controller transmits the parameterization and configuration data to the IO device via the "Record Data CR". The cyclic transmission of the application data takes place via the "IO CR", whereas the noncyclic events are transmitted to the IO controller via the "Alarm CR" and are acknowledged. Alarm types used for PROFINET are pulling, plugging, diagnostics, status, update alarm. Vendor-specific alarms are also possible. Alarms can have a high or low priority.
2.2.4.2  Device model

For the PROFINET IO device, a consistent device model has been specified that permits the modeling of modular and compact field devices. It is oriented to the basic principles of PROFIBUS DP and consists, for a modular field device, of slots that accommodate modules. The modules carry IO channels over which the process signals can be read in or put out. The slots or modules can be further subdivided into subslots/submodules.

The IO device itself is addressed via Slot0, which acts as the "station proxy".

Subsystems can be mapped to this model in which the different device functions are assigned to modules or submodules.

![Device model with PROFINET IO](image)
2.2.4.3 Device description (GSD)

The characteristics of a PROFINET IO device are described by the device vendor in a device description, the Generic Station Description (GSD). The GSD is created in the Generic Station Description Markup Language (GSDML).

The GSDML is described in the PNO guideline "GSDML Specification for PROFINET IO":


Access requires a login with the PNO (PNO membership).

The GSDML contains all necessary information for the field device:

- characteristics of the IO device (e.g. communication parameters)
- plug-in modules (number and type)
- configuration data of each module (e.g. analog input module)
- parameters of the modules (e.g. 4...20 mA)
- error texts for diagnostics (e.g. wire breakage, short-circuit)

The description base for GSDML is XML. With XML being an open, widely used and accepted data description standard, powerful tools and derived characteristics are automatically available, such as:

- creation and validation through the use of standard tools
- foreign language integration
- hierarchical structure

The structure of the GSDML complies with ISO 15745, comprising a device-specific part with the configuration data and module parameters and a communication-specific part.

Parts of the GSD can already be predefined by means of standardized device interfaces or PNO profiles, respectively, so that "only" the device-specific characteristics need to be added.

Each device is assigned a globally unique device identification within the context of PROFINET. The 32-bit Device Ident Number is to be requested from the PNO.

This number consists of the vendor ID and the device ID.
Figure 2-9: Example of a Device Ident Number

The 16-bit vendor ID is assigned by the PNO. The 16-bit device ID can be defined by the vendor during its product development.

The device vendor assigns the modules a module identification that is unique among all devices, the Module Ident Number. This 32-bit number is to be managed by the device vendor and to be stored in the device description.

The device vendor assigns the submodules a submodule identification that is unique among all modules, the Submodule Ident Number. This 32-bit number is defined by the device vendor and to be stored in the device description.

2.2.4.4 Project planning

The GSD file of an IO device is imported into an engineering system where it forms the basis for the planning of the configuration of a PROFINET IO system.

In the engineering system, the IO devices are assigned a NameOfStation (station name). The configuration tool ensures that the NameOfStation is unique within the project planning area and can thus be used as a unique key for identification in the network.

The NameOfStation must be loaded into the IO device where it must be permanently stored. It is used to identify the IO device in the IO controller as well as the installation location (instantiation of IO devices).

With the NameOfStation, the IP address can also be loaded into the IO device and permanently stored (alternative option: IP address is assigned by the IO controller during start-up based on the NameOfStation).

Each IO channel of the field devices is assigned a peripheral address in the IO controller. The peripheral input addresses contain the received process values. These are evaluated and processed by the user program. The user program computes the peripheral output values and outputs them to the process. In addition, the individual peripheral modules or channels are parameterized in the engineering system, e.g. 4...20 mA current range of an analog channel. Once the configuration is complete, the project and configuration data are downloaded into the IO controller. The IO devices are automatically parameterized and configured during operation by the IO controller and will then start the cyclic exchange of data.
2.2.4.5 System start-up

Each IO device must know its NameOfStation for identification. In the configuration phase, each IO device is assigned a parameter block that can be uniquely identified via the NameOfStation. This parameter block is available in the IO controller.

After power-up and successful start-up of the devices, the IO devices will initially determine their neighborhood relationships. For this purpose, each IO device will tell its "neighbor" such parameters as NameOfStation or MAC address. Neighbors are the next devices that are directly connected to the switch of an IO device via Ethernet ports. The neighborhood relationships are determined and cyclically updated via the LLDP protocol.

For start-up, the IO device logs into the IO controller using its station name. The controller transfers the parameterization data to the IO device via the Context Management protocols. If the IO device did not have its IP address remanently stored, it will also receive its IP address.

In turn, the IO device transmits its determined neighborhood relationships to the IO controller.

The first part of Context Management also includes the synchronization of PROFINET in order to meet the requirements for the IRT data exchange. In the second part of the CM, the application relationships are established. This division into two parts is necessary to permit the start-up of PROFINET with its switches without the necessity of running the applications in the IO devices. This means that the IRT data exchange can take place even if devices are switched off or have a longer start-up time. The application relationships can then be set up and removed without disturbing the network.

According to the device model, device applications are viewed as slots that can be pulled or plugged in.
2.2.4.6 **Cyclic communication**

Cyclic data (RT and IRT data) are transferred not secured and without acknowledgement from the provider to the consumer in a fixed pattern. At the consumer end, the non-arrival of cyclic data is detected by means of a configurable time monitoring that issues a corresponding error message to the application.

The set-up and removal of the IO CR (e.g. link relationship between IO controller and IO device) is handled by a higher-level protocol. IO controller and IO device may be a provider or a consumer, depending on the direction of the communication relationship. A provider does not receive an explicit feedback from the consumer whether the data have arrived. The feedback channel between the two devices requires another links with reversed roles.

![Cyclic data transmission](image)

*Figure 2-11: Cyclic transmission of process data*

The cyclic protocols does not support the segmentation and reassembly of the records. This means that the total length of a data package, including all protocol headers, must not exceed the length of an Ethernet package. The individual storage blocks in the package are assigned specific functionalities that are maintained for each IO CR. The user interface in the provider and the consumer operates in buffered mode. If data are written to the interface faster than they are fetched, previous data will be overwritten. Each provider is assigned an update interval that must not be underrun. Each consumer monitors its provider by checking for regular receipts.

For each submodule, the cyclic data are provided with information about the validity of the data (IOPS, Producer Status of IO data and IOCS, Consumer Status of IO data).

2.2.4.7 **Noncyclic communication**

Noncyclic data (NRT data) are transmitted event-driven at specific times from the provider to the consumer. The user interfaces in the provider and the consumer operate in queued mode. If data are written to the interface faster than they are fetched, this may result in an overflow (with corresponding error response) and thus in a loss of data.

The data are transmitted with or without reference to a specific link, depending on the protocol. The remaining characteristics are defined by the respective protocols and will not be further explained here.
2.2.5 Network structures

PROFINET networks can be configured as line, tree, ring structures or combinations thereof. Ring structures offer the advantage that they enable disjunctive paths between the devices (media redundancy) to control single point of failure.

The switch function of the ERTEC-based interface modules can be used to build ring structures. Benefits include:

- cost-effective solution since no additional external switches are required.
- single point of failure (failure of a single device or line interruption) can be tolerated.

![Simple ring](image)

The PROFINET network structure is individually defined for each system. The definition of the structure determines whether the corresponding subsystems operate on the ring or outside the ring over a stub line. In this case, the connection is established via a free port of a switch that operates on the ring.

2.2.6 Switch

With Sibas PN, the switches that are required for Fast Ethernet are integrated into the controllers as much as possible and also serve as the interface of the controller to PROFINET. This interface can also be used to implement the service/debug interface.

The key component is the ERTEC Ethernet controller. A switch has the following characteristics:

- 2/4-port switch
- all ports optionally 10 or 100 Mbit (only 100 Mbit full-duplex is used for PROFINET and Sibas PN)
- layer-2 routing
- IRT protocol routing
- self-diagnosis according to SNMP (Simple Network Monitoring Protocol)

The use of external switches is avoided for cost reasons but may become necessary to connect individual controllers or to implement special functions.
Like IO devices, switches are assigned to an IO controller that provides them with the necessary parameterization information.

## 2.2.7 External power supply

In order to maintain the full functionality of the PROFINET network, power must be supplied to the switches that are integrated in the communication interface even if the connected controller is powered off (e.g. operational shutdown or shutdown caused by a failure). This means the switches that are integrated in the devices continue to run when the controller fails or is shut down. The communication paths are not affected.

For this purpose, the PROFINET switches must be powered by a separate power supply. The communication interfaces can also be powered via the controllers.

![Figure 2-13: Principle of external power supply for the switches (communication interface)](image)

Figure 2-13 illustrates the principle of the external power supply (floating) for the controllers with the switch integrated in the communication interface.

The external power supply must be implemented for all subsystems having their integrated switch located in the communication path of other subsystems (e.g. a switch in a ring structure) and capable of being individually shut down during operation.

The requirement of an external power supply shall be defined for the individual subsystems by SIEMENS TS in cooperation with the subsystem vendor.
2.2.8 PROFINET interfaces for subsystems

The following possibilities exist for the connection of subsystems as IO devices to PROFINET:

- PROFINET IRT communication (synchronized communication) or RT communication with communication ASIC.
- PROFINET RT communication (non-synchronized communication) using a standard Ethernet controller and PROFINET IO firmware.

Synchronized communication (IRT, isochronous realtime) requires the use of the Enhanced Realtime Ethernet Controller ASIC (ERTEC). The ERTEC is also required if the PROFINET switch function is to be integrated and the device is to be operated on the PROFINET ring.

**Note:** In order to be able to integrate control computers of subsystems (e.g. door, A/C, passenger information, ...) in different network topologies (ring, line, star or tree topology) without additional switch devices, it is necessary to implement an integrated PROFINET switch in the control computer of the subsystem. When the SIEMENS PROFINET PC/104 (see chapter 2.2.8.4) or an ERTEC integration (see chapter Fehler! Verweisquelle konnte nicht gefunden werden. ) is used, this is already supported by the hardware of these devices.

The definition whether the RT or the IRT protocol is used for communication has to be made for each IO device by SIEMENS TS in cooperation with the subsystem vendor.

The SIEMENS PROFINET PC/104 module is available from the following vendor:

Siemens AG
I&S EDM
Mail: friedrich.pook@siemens.com
Frauenauracher Str. 98
91056 Erlangen (Germany)

The ERTEC ASIC is available from the following vendors:

**SIEMENS:**
Development Kit for PN IO for Ethernet processor and for ERTEC 200/400

**NEC:**
ERTEC200/400
http://www.profibus.com/member/nec_electronics__europe__gmbh/train/partner/

In addition to the ERTEC ASIC, communication modules and development kits from different vendors are available for communication integration on PROFINET.

An overview is presented at the PNO Web site:

http://www.profibus.com/pn/applications/
2.2.8.1 ERTEC ASIC

The Enhanced Realtime Ethernet Controller ASIC (ERTEC) from Siemens AG or NEC is available as a PROFINET switch and for the connection of devices to PROFINET. The ASIC is available in the versions ERTEC 200 (2-port) and ERTEC 400 (4-port).

Information about the SIEMENS products is available at http://www.automation.siemens.com/microsite/ertec/html_76/ertec_e.swf

The Industrial Ethernet ASIC ERTEC is a high-performance 2-port switch (ERTEC200) or 4-port switch (ERTEC400) with integrated 32-bit microprocessor that has been designed for industrial applications. It can be used to implement PROFINET interfaces and switches that provide openness to the IT world as well as realtime capabilities. An easy, space-saving connection of devices to Switched Ethernet is enabled by the implementation of a highly integrated switch. The high processing power of the integrated high-performance ARM 946 RISC permits optimum communication interfacing with advanced internet applications. The integrated PCI interface can be used to easily connect PC products to PROFINET.

Due to the integrated ARM 946 processor, the ERTEC can be used as a system-on-chip implementation for simple devices. Since the processor-intensive cyclic communication for PROFINET IO with realtime and isochronous realtime is completely handled by the ERTEC hardware, the remaining processor resources are sufficient to cover also the applications of simple field devices. This eliminates the need for external processors. On-chip peripherals enable the direct connection of IO signals, serial interfaces and timers.

The ASIC ERTEC is available in compliance with RoHS. The lead-containing variants are no longer available as of October 1, 2007.

The peripherals that are typically required by the ERTEC are 4 MB ROM and 32 MB flash memory.

The interfacing of the ERTEC to the CPU of the subsystem can be implemented – depending on the ERTEC type – via the Local Bus Unit (LBU) or a PCI interface.

The table below indicates the key differences between the ERTEC 200 and the ERTEC 400:

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>ERTEC 200</th>
<th>ERTEC 400</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of ports</td>
<td>2-port switch</td>
<td>4-port switch</td>
</tr>
<tr>
<td>Ethernet PHY</td>
<td>Integrated</td>
<td>External</td>
</tr>
<tr>
<td>Host interface</td>
<td>Local Bus Unit (LBU)</td>
<td>PCI and Local Bus Unit (LBU)</td>
</tr>
<tr>
<td>Quantity framework for RT and IRT data</td>
<td>Up to 200 bytes each for input and output</td>
<td>Up to 1440 bytes each for input and output</td>
</tr>
</tbody>
</table>
The connection of the Ethernet controllers to the physical communication network is implemented by so-called PHYs, which are modules for the transition from the internal to the external world.

These are already integrated in the ERTEC 200. When the recommended PHYs for the ERTEC 400 are used and with the ERTEC 200, the auto-negotiation and auto-crossing functions are supported.

2.2.8.2 Development Kit ERTEC

The PROFINET IO Development Kits ERTEC 400 and ERTEC 200 can be used to develop and test PROFINET hardware and software application by using the ERTEC ASICs.

The corresponding Development Kit makes the PROFINET technology accessible to device vendors and users.

The Development Kit ERTEC contains all components for the development of a PROFINET IO device with Realtime (RT) and Isochronous Realtime (IRT):

- Evaluation Board with ERTEC 200 or 400 as PROFINET IO device
- PROFINET IO stack for PROFINET IO device (availability as object code for VxWorks and planned availability as source code)
- CP 1616 (PCI board) for PROFINET IO controller
- NCM PC Software package for configuration of the CP 1616 as engineering system for PN IO

2.2.8.3 Development Kit for PROFINET IO

The Development Kit for PROFINET IO based on standard Ethernet controllers (with application example for NetARM processors) is available for non-ERTEC Ethernet connections. It permits the implementation of communication interfaces without switch functionality for the PROFINET IO RT protocol stack.

The Development Kit comprises:

- firmware stack in source code for PROFINET IO, customized to the example of the NetARM processor Net+50 from NetSilicon. Of course, this FW stack can also be ported to any other processor from NetSilicon or any other vendor.
- user description for the PN IO stack
- comprehensive documentation and program examples for an IO device
- comprehensive documentation and program examples for a CPU317-2 PN/DP that can be used as a test partner (IO controller).

The Development Kit is designed for 32-bit systems. The recommended hardware for its use includes:

- 32-bit processor, 32-bit operating system
- >= 4MB RAM, 2MB flash
2.2.8.4 PROFINET PC/104

The PROFINET PC/104, referred to as PN PC/104 below, is an interface module that permits the interfacing of electronic controls to PROFINET. The PN PC/104 module acts as a PROFINET IO device. This cannot be used to assume the function of a PROFINET IO controller.

Users of the PN PC/104 modules are provided with an API software as a source code for their design projects, which enables them to run the module as an interface for their controllers as a PROFINET IO device. This provides a simple way to integrate the PROFINET modules into any operating system environment using a standard PC/104 interface. The PN PC/104 hardware handles the high-performance data exchange via PROFINET independently. Host interfacing is implemented via a DPRAM by means of the application programming interface. This ensures the fastest possible access to the data.

The ERTEC 200, as a high-performance Ethernet controller with integrated real-time switch and 32-bit microprocessor, is the perfect solution for connecting electronic controllers to PROFINET or Ethernet.

The benefits of the PROFINET PC/104 module include:

- Easy installation and commissioning
- High-speed, consistent access to process data
- Short response time
- Software in source code as application programming interface for host interfacing
- Low-cost migration from MVB to PROFINET (if MVB PC/104 was used)
- No license fees for subsystem vendors
- Low host usage since the PN stack runs on the PC/104
- Consistent subsystem interface to control technology through consistent service interface
- Subsystems can be connected in a line and ring structure with one type of the PN PC/104 module through the integrated switch.
- Hiding of the Profinet protocol and the SIMATIC-Rail specific services.
- Communications between the PN PC/104 module and the host system is implemented via a DPRAM. For ease of use, the original protocols are abstracted, i.e. they are made available to the user via a clearly-structured interface.
2.2.8.5 Ordering information / Technical support

The following components are available from SIEMENS A&D:

- Development Kit DK ERTEC 400 PN IO
  Order number: 6GK1953-0CA00

- Development Kit ERTEC 200
  Order number: 6GK1953-0BA00

- Development Kit for PN IO for Ethernet Processor (NS9360)
  Order number: 6ES7 195-3BC00-0YA0

Detailed information on the PROFINET technology components are available at:

http://support.automation.siemens.com/WW/llisapi.dll?aktprim=0&lang=en&referer=%2fWW%2f&func=cslib.csinfo&siteid=csius&caller=view&treeLang=de&DataKey=18881571&extranet=standard&nodeId0=18881571&objaction=csopen

The Integration Centers in Europe and USA provide toll-free telephone support for developers and implementers. The comprehensive services of the Integration Centers also include developments of software and hardware to order as well as on-site support. One-day crash seminars are also offered for developers and implementers. This service addresses primarily PROFINET users from industrial automation.

The use of a so-called Development Kit and a device description by means of GSD provides interfaces for the PNO-defined standard information. These interfaces are to be supplied with information by the subsystem vendor according to their specification.

2.2.9 Standard Ethernet integration

Subsystems that are already equipped with an Ethernet interface (i.e. not with a PROFINET interface) and that are not compliant with the PROFINET IO model can be integrated into PROFINET via a switch. This type of integration is only permitted in exceptional cases in the SIMATIC Rail project (see chapter 2.2.8). Communication is exclusively handled via non-realtime data (TCP/IP). The PROFINET IO device model and the PROFINET IO protocols (RT and IRT) cannot be used in this configuration.
2.3 Additional requirements for subsystems

2.3.1 Functional interface

The control interface between the vehicle control system and the subsystem (functional interface of a subsystem) is defined separately and is not described in this document.

The definitions are to be made between SIEMENS TS and the system vendor.

The functional interface is used, for example, to make specific definitions for the following functions:

- cyclic data
- noncyclic data
- operating status of vehicle and subsystem
- functional structure of the subsystem interface
- default behavior
- operator control and visualization
- functional and component diagnostics

2.3.2 Device behavior

The subsystem must not exhibit an undefined behavior in any operating state or provide undefined data at outputs. If no current data are available, defined default values shall be provided or the data shall be marked as invalid.

A defined start-up shall be carried out after power on.

The operating behavior of a control unit must not affect the communication on the PROFINET via the switch that is integrated in the communication interface.

2.3.3 Supervision times

Values for supervision times are to be set on the IO device for different mechanisms. The settings are made during system configuration. The parameters are loaded by the IO controller into the IO device during start-up or are to be set on the IO device.

2.3.4 Functional safety

The subsystem contractor is fully responsibly for the safety integrity of the functions provided by his subsystem. This regulation does apply even if the contractor uses Sibas PN / PROFINET integration according to this specification.
This regulation does also apply for all operational modes of the vehicle, such as regular operation, maintenance, but also parameterization, test runs of or with subsystems or SW-updates of subsystems.

If safety requirements collide with other requirements, such as required use of SIBAS PN functions, safety requirements are higher priority. In this case further commitments are to be made between SIEMENS I MO and the subsystem contractor.

Sibas PN delivers functions according to SIL 1. This safety integrity was certified by TÜV Süd as an independent test centre. The subsystem contractor can use these functions (e.g. communication, local services) in one’s sole discretion, if they are sufficient to match the safety integrity of the functions provided by his subsystem.
2.4 Lines and connectors

Device connections for PROFINET are implemented as D-coded M12 plug-in connection. Railway-proof lines are to be used for the communication line.

Specifications and vendor information are to be obtained via TS GT CoC VC.

2.5 Local services

2.5.1 Definition

Local services are implemented on individual devices, such as a download service for software and parameters in a controller or subsystem. These local services carry out the function (e.g. software download) for the respective device as a locally implemented server on the device. The function of the server basically may be used by all applications that can access the local server with respect to the communication connection and security environment. Such clients may include, for example: Applications for download and for parameterization of a subsystem, which can address the device directly via the vehicle bus.
Lokale Dienste laufen auf den Kommunikationsteilnehmern ab. Die gezeigten Schnittstellen sind für Teilnehmer am Fahrzeugbus zugänglich.

![Diagram of local services](image)

**Figure 2-14: View for consistent (local) services.**

Ideally, all local services of the same type (download, identification, ...) should have the same interfaces and use the same communication protocols.

Initially, the following local services are defined:
- download,
- identification,
- time synchronization,
- diagnostic service for maintenance (workshop)

Other services may follow at subsequent release levels.

The "ISubsystemStatus" interface defines a consistent status interface through which each subsystem communicates its functional status, such as "ok", "fault", "Test running", "Test request" in service mode, operation etc. The definition of this interface is indicated in more detail by the specification of the functional interface of the subsystems.
This interface is not part of the services, but results of service calls (e.g. for diagnostics or self test) are indicated (in a consistent manner) at this interface.

### 2.5.2 Local download service

The local download service carries out the loading of software and parameters into devices on the vehicle bus. The loading operation occurs via the vehicle bus. This ensures efficient loading even into devices that are installed at locations that are difficult to access\(^1\).

The local download service is mandatory for all devices that support loadable software and parameters.

For PROFINET IO devices it is possible to assign the parameters during start-up when the application relationship between the PROFINET IO controller and the IO device (subsystem) is initialized: Data specified in the GSD for the subsystem can be assigned values in the engineering system. These values will then be sent by the PROFINET IO controller to the subsystem during initialization.

For all other loadable software or firmware components applies: The loading of the firmware occurs over the PROFINET vehicle bus with FTP. This requires that the device that supports the download is locally equipped with an FTP server with password-protected access. This FTP server can be used to load the firmware. Once loaded, the firmware is checked for validity, and the data are written into the active area if the check is successful. After a reset (or power off/on), the new SW/parameter version will become effective in the device.

The "active area" designates the storage area where the image to be started resides when the device is booted.

---

\(^1\) With respect to the correct IT term, the "download" is actually an upload to the FTP server in the device. However, since the designation "Download into the controller/subsystem/..." has been established at TS Rolling Stock, the term "download" is also retained here.
Download with feedback of validity

Here, the software update is carried out as a step-by-step process. After the successful FTP download, a "status file" is retrieved from the subsystem. The requirements for the user are higher here; he/she is required to evaluate the status feedback in order to obtain more detailed error messages.

According to RFC1350, the FTP server must [2] at least master the commands that are indicated in RFC 959 [3], paragraph 5.1, see Figure 2-16:

5.1 MINIMUM IMPLEMENTATION

In order to make FTP workable without needless error messages, the following minimum implementation is required for all servers:

- TYPE - ASCII Non-print
- MODE - Stream
- STRUCTURE - File, Record
- COMMANDS - USER, QUIT, PORT,
  - TYPE, MODE, STRU,
    - for the default values
  - RETR, STOR,
  - NOOP.

The default values for transfer parameters are:

- TYPE - ASCII Non-print
- MODE - Stream
- STRU - File

All hosts must accept the above as the standard defaults.

Figure 2-16: Minimum FTP server functionality according to RFC 959.
As TYPE, in addition to Default, the type IMAGE (RFC 959, paragraph 3.1.1.3. IMAGE TYPE) must also be supported, for this serves the efficient transfer of binary data.

In order to be able to use standard FTP server software without the need for intervention, the following is defined:
To let the subsystem know that it has received an update, a control file is polled, and the actions are executed if it has changed. After the execution, or the cancellation of the download, a status file is written with pass/fail messages and can retrieved by the user.

For devices whose software consists of multiple images, there are two possibilities to ensure the consistency of all images.

1. Download all images and indicate all files in the control file. This enables the subsystem to check all images first before starting the activation.

2. An archive (ZIP) is also supported for the download. This archive is used to pack all images and send them to the subsystem as a package.

In devices with sufficient RAM, the FTP download is written into the RAM and is only written into the active area when the validity check was successful ("Flash burned"). If no RAM is available, a temporary area must be created in the flash memory. The FTP download is stored in this area. If the image is valid, it is written into the active area. For this purpose, the actual software update is preceded by a header, forming Download Object (DO). The Download Object always consists of a header and the actual update data. The header contains the following information:

- magic Number ( 4 bytes firmly defined 0x08075352 )
- length of header ( 2 bytes unsigned int )
- length of DO ( 4 bytes unsigned int )
- version of DO ( String: 2 bytes (unsigned short) Length of string + bytes (ASCII) )
- type of DO ( String: 2 bytes (unsigned short) Length of string + bytes (ASCII) )
- checksum MD5 ( Bytes: 2 bytes (unsigned short) Number of bytes + Byte array)

The length of the DO includes the entire length including the header.

The version of the DO has the format of the Numeric Software Versioning Scheme. It consists of Major, Minor and optional revision and build number each separated by a point. The numbers may also contain characters, e.g.: "1.3rc1".

For the format: major.minor[.revision[.build]] see [4].
When a subsystem supports ZIP decompression, it is also possible to download an archive with multiple download files. It must be ensured that each download file always has this header before the software update.

The subsystem recognizes by means of the Magic Number whether the downloaded object is a DO or an archive. ZIP Magic Number is "PK" (Phil Katz) in Hex: 0x504b
The validity is determined by means of the following checks:

- integrity of the DO (correct checksum, Magic Number, length of DO).
- valid DO type.
- valid DO version if applicable (subsystem can then decide itself if version dependencies exist and not perform a download).

The subsystem polls in intervals of 5 seconds maximum (the time to 5 seconds max. is determined by the device vendor and also described accordingly in a document) whether the file "control.dfs" exists. If the operating system/file system of the subsystem has an event-driven change code, this should be used instead of polling.

All download files (DO, archive, control file) are stored in the "root" directory of the FTP server. The "root" directory of the FTP server need not necessarily be the "root" directory of the file system. This directory is determined by the subsystem vendor.

When the control file is changed, the status file (if any) is deleted first and the control file is parsed and the actions carried out.

If the download file has passed the validity check, the software update is written into the active area. Then the status file is written. The temporarily stored items including the control file are deleted again.

During the "burning operation", the firmware is set to an invalid status and then reset to valid when the burning has been completed successfully.

The validity check must also take place in the bootloader so that a check is possible even in the "unloaded state".

Activation of the DO may only be permitted if the state of the device excludes any risk for operation. This is achieved by means of PROFINET IO:

If an AR (Application Relation) exists between the controller and the device, the information of a "Service Mode" variable is referenced in order to determine whether an activation is permitted for the subsystem or not. This variable is defined via the functional interface of the subsystem.

If no AR is available, it is assumed that no controller is available and a download is possible. In case of a short communication failure, it is not assumed that this is an unauthorized software update (because the vehicle bus is a "trusted environment").

When the CPU goes into Stop mode, it sets the PNIO controller to Clear Mode. The devices become aware of this status and must permit the software download.

If the subsystem supports a software reset, this can be triggered with the control file. If not, a corresponding message is written into the status file for this action: "not supported". These devices can then only be reset by switching them off and on again.

The software reset can be performed in 2 places:

1. In the control file, update requests are sent with reset in an action. This is the best way for the scenario where no status file is polled. This means that a reset is carried out exactly when the software update was checked and accepted.

2. Only the update requests are indicated in the control file. The user waits until the status file has been written and sends another control file with a reset command. This way is meant for the second scenario where the user wants to read the status file first and performs the reset only after evaluation.

The control file (filename: "control.dfs" dfs = Download File Script) is a simple ASCII file that consists of simple commands.
Each command starts in a new line with ":," followed by the command. The following characters all belong to the parameters of the command and end with a new command or End-Of-File. All commands are case-sensitive. Supported file formats are Unix and DOS.

The commands provided for the control file are:

<table>
<thead>
<tr>
<th>Command</th>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>:update</td>
<td>filename</td>
<td>Filename of the DO or archive. The file can be found in the &quot;root&quot; directory of the FTP server. A path must not be specified.</td>
</tr>
<tr>
<td>:reset</td>
<td>sec (optional default=0)</td>
<td>Performs a software reset. sec is an optional parameter and specifies the time between the writing of the status file and the execution of the reset. This gives the user the opportunity to obtain the status of the validity check even before the reset.</td>
</tr>
<tr>
<td>:spezial (optional)</td>
<td>args (optional)</td>
<td>Special command for subsystems. This permits the inclusion of specific information for subsystems. These special commands are to be described in the subsystem specification.</td>
</tr>
</tbody>
</table>

The implementation of ":reset" and ":spezial" is not necessarily required, whereas the ":update" command is mandatory. All update requests that are referenced in the control file are checked for validity first and will only be accepted if all of them are valid and the "special" command (if any) was successful. The sequence of acceptance corresponds to the sequence in the control file ( if multiple updates are present ).

The reset command can be used at any point and will only be executed after successful validity check, writing of the status file and any waiting time. If the validity check or a special command fails, the reset is not executed. This lies within the responsibility of the device vendors.
If a control file only contains a reset (no update), this will immediately be executed and the "sec parameter" ignored.
If the special command invalid, the updates will not be accepted.

Example control file:

```
:update image1
:update image2
:reset
```

The status file format (filename: "status.dfs") corresponds to the control file format with an extension.
Every action from the control file is assigned a status.
The status starts with "=" at the beginning of a line, followed by a status.

Status values:

<table>
<thead>
<tr>
<th>Command</th>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>=ok</td>
<td>description (optional)</td>
<td>Action successful. Optionally it is possible to add information. Texts must be written in English.</td>
</tr>
<tr>
<td>=fail</td>
<td>description (optional)</td>
<td>Action failed. Optionally it is possible to add information. Texts must be written in English.</td>
</tr>
</tbody>
</table>

Only if all :update commands return =ok will all updates be written into the active area. Otherwise, =ok only signifies that the validity check was successful during this update.

A :reset fail gives no indication about the acceptance of the update. This could also be a subsystem that does not support this command.

Example status file:

```
:update image1
=ok
:update image2
=fail wrong checksum
:reset
=fail update fail
```

The FTP server is assigned a login and password.
These are defined by the device vendor and described in the specification.

The user can perform the software update by means of an FTP client.
The operation can also be automated by means of scripts. For example, for the device update without status feedback.
Example of a download with an FTP client of Windows:

C:\>ftp 192.168.2.2
Connection to 192.168.2.2 was established.
220 ProFTPD 1.3.0a Server (ProFTPD Default Installation) [192.168.2.2]
User (192.168.2.2:(none)): ftp_download
331 Password required for ftp_download.
Password:
230 User ftp_download logged in.
ftp> put d:\image1
200 PORT command successful
150 Opening ASCII mode data connection for image1
226 Transfer complete.
FTP: 64d bytes sent in 0.00seconds 124416000.00Kbps
ftp> put d:\image2
200 PORT command successful
150 Opening ASCII mode data connection for image2
226 Transfer complete.
FTP: 64d bytes sent in 0.02seconds 7776.00Kbps
ftp> put d:\control.dfs
200 PORT command successful
150 Opening ASCII mode data connection for control.dfs
226 Transfer complete.
FTP: 64d bytes sent in 0.00seconds 124416000.00Kbps
ftp> get status.dfs
200 PORT command successful
150 Opening ASCII mode data connection for status.dfs (124414 bytes)
226 Transfer complete.
FTP: 64d bytes received in 0.02seconds 7799.63Kbps
ftp> quit
221 Goodbye.

2.5.3 Local configuration service

The local configuration service will not be implemented in the first step. This chapter is to be considered an outlook to future extensions.

The local configuration service handles the change of parameters in the device during operation if this is required for the device. The parameter changes can become immediately effective for operation. The configuration protocols are still to be defined, they must basically provide the following possibilities:

1. The device requests a parameter set from a known communication partner.
2. The device receives the request to update one or more parameter sets, with indication of the location of the data sets. The device can then send a request (after 1.) and update its parameters if it decides to permit this. This request may basically come from any client.
2.5.4 Local identification service

The local identification service is mandatory for all devices on the vehicle bus. It permits to request all type identifiers and versions of all replaceable hardware and software modules of the device.

For subsystems on PROFINET IO (PROFINET IO devices), the following information about the subsystem is stored for each module in the parameterization of the subsystem interface by means of the GSDML file:

- name,
- vendor,
- order number,
- hardware version,
- software version.

This information can be read out via the engineering tools.

The version information is stored for all devices in a file that can be read out via FTP.

This makes the version information of all devices available in a consistent format so that it can also be read out without a PROFINET IO controller. The file that contains the version information is named "description.xml" and is available in readable form in the "root" directory of the FTP server.

The file format for the file is specified by the GSDML format definition. As a minimum requirement, the values for the above data are to be entered in the sections for the ApplicationProcessType (with ModuleList and SubModuleList). For PROFINET IO devices that need a GSD anyway for configuration purposes, this should be stored unchanged as "description.xml". In this manner, the device brings along its matching GSD, which can be read out without a PROFINET IO controller.

2.5.5 Time synchronization service

The time synchronization ensures that the different local clocks in the different devices across the train that are accessible accessed by communication are synchronized over time as continuously as possible. The time synchronization is mandatory for all devices that use time information, e.g. respond to times/differences, generate or evaluate time stamps.

The protocols used for time synchronization will be NTP (RFC 1305) and SNTP (RFC 2030). An NTP server on the PROFINET vehicle bus is available as a time source.

Sources for NTP and SNTP are fully available on the Internet and are also supplied as part of the operating system (with the VxWorks versions used and LINUX).

The SNTP/NTP client can choose whether it will respond to a broadcast from the NTP server (passive client) or actively attempt to retrieve the time from the NTP server (client must know IP of NTP server).

It should be noted that with the passive client the NTP server determines the time during which the client attempts to synchronize. For an active client, the time of the next synchronization must be explicitly selected. In this case, the time is determined by the device vendor and is documented accordingly.

2 Here again, a real file and file system must not be available, provided that the RECV call of the FTP server returns the data in the defined format.
Caution: For applications with the VxWorks operating system, there can be the limitation that the "passive client" method cannot be used.

Within the system, the time is always indicated in UTC. For the purpose of input and output, UTC is to be converted according to the current time zone setting. The time is transmitted unencrypted.

The NTP server ensures that the time does not make any major jumps and does not count backwards (adjust). This also has an impact on all NTP/SNTP clients. When an NTP/SNTP client is booted, it must synchronize with the server and set the time (no adjust if supported).

2.5.6 Self-test

The self-test service will not be implemented in the first step. This chapter is to be considered an outlook to future extensions.

The self-test service permits to initiate self-tests in subsystems over the vehicle bus and it can be implemented for the subsystems for which such test capabilities are required. The self-test service receives the request to carry out test runs. The request may basically come from any client on the vehicle bus.

The self-test service performs the tests (or initiates them) if the execution of the tests is permitted. The results of the local self-tests are made available via the diagnostics service. If the execution or the results of the tests lead to functional limitations, these will be published at the subsystem status interface. (This is „SubsystemStatus“ in Figure 2-14.)

2.5.7 web server for device-specific diagnostics and service

All stations on the vehicle bus have to provide a local web server, such as for facilitating device-specific service, supporting a diagnostics for maintenance, monitoring and parameterization.

The presence of such a web server does not replace the services that have been described as mandatory above.

The “PN PC/104” communication subassembly provides support for these functions. There is a web server with java-applets mounted on the subassembly which allows a direct communication between a webapplication and the Systemhost (of the subassembly) via TCP/IP. Alternatively to java-applets other webobjects can be loaded and viewed on a remote browser.

3 If the time difference between server and client is very large, the time is set in the client and no adjust takes place. To avoid this, it is planned for NTP/SNTP to reset the time during a restart. If the time varies heavily during operation on the NTP server, it remains an open point as to whether adjust will be able to control this in realtime Linux.
2.5.7.1 Webcontents

The following functions have to be provided by the web-application of the subsystem (e.g. Door unit, brake ...) contractor:

- display and evaluation of all diagnostic memories of the subsystem (see Fehler! Verweisquelle konnte nicht gefunden werden.) incl. operational- and protocol-memories
- display of current maintenance-messages and process-values (e.g. inputs, outputs, voltage, currency, temperatures, speed ...)
- input of maintenance- or service- parameters
- start of test runs
- selective reset of current maintenance-messages
- display data of the identification-service (see 2.5.4)
- change of displayed language (support for project-specific languages, English and German are minimum requirements)
- Storage and access to data files (such as device-documentation ...)
- The java-applets should be signed to guarantee the data-integrity.

2.5.7.2 Requirements according to functional safety of web application

The subsystem contractor is fully responsible for the safety integrity of the functions provided by his subsystem. This regulation does also apply if the contractor uses web applications (see Fehler! Verweisquelle konnte nicht gefunden werden.).
2.5.7.3 HTTP-authentication

There must be no access to information/functions of the web server without user authentication (see 2.5.7.1).

Therefore a Basic-http-authentication, which requests a username and a password, is required.

The authentication has to begin with a standard http-request (as shown in figure 2-19: basic authentication) made by the http-client (e.g. a web browser on HMI or a maintenance-pc). If the access is restricted by http-authentication, the http-response has to contain a WWW-authentication-header.

The http-client is requested to send a username and a password. A web form is opened in the client-window to ask the user to enter name and password, which will be sent to the server. If the authentication is valid, access will be granted.

The authentication will be transmitted on each following client-request.

After 3 unsuccessfully authentications an error message will be generated, which informs the user about the unauthorised access of the password protected area.

A "basic authentication" is sufficient since there is no public network between the maintenance-PC or the HMI and the subsystem. Access protection and encryption via WAN is not part of this consideration and has to be provided by other mechanisms.
2.5.7.4 Use cases for web application

**direct access to subsystem web server**

The user starts a web browser on the maintenance-laptop and calls the URL of the subsystem web server (1.)

The subsystem web server delivers the corresponding HTML-page. (2.) The user performs the related actions on the webpage.

The required action will be directed to the subsystem host. (3.)

The subsystem performs the actions. (4.)

The subsystems send the results back to the web browser.

SP SR and SP CS are not part of this communication.

User authentication must be performed on the subsystem web server or the HTML-page. The subsystem has to ensure that all conditions to perform the action are proven.

**direct access to subsystem web server within navigation to SPSR web-portal**

In comparison with the left case the user does not call the subsystem URL, but the web-portal URL of the SP SR

Access to the web portal of the SPSR is also restricted by basic authentication.

By pressing a button in the web-portal the user will be redirected to the HTML-page of the subsystem.

Another user authentication must be performed by the subsystem web server or the HTML-page.

SPCS is not part of this communication.

*figure 2-20: use cases web application*
2.6 Diagnostics

Diagnostics is a method or activity that takes symptoms (indications) to derive the nature of a failure.

Diagnostics covers the necessary activities that are specific to inspection, recording, storage and transmission (if applicable) or reporting of diagnostic data. This is designed to support a wide variety of activities and users, such as:

- The train personnel who is informed of any service restrictions (e.g. reduced tractive effort etc.) and provided with suggestions for corrective measures if necessary.
- Maintenance staff who are provided with fault information that can e.g. be used to analyze a problem.
- Maintenance scheduling staff who are provided with fault information in advance (i.e. before the railway vehicle reaches the depot for maintenance). This enables or supports the planning of maintenance work (support of corrective maintenance).
- Maintenance scheduling staff who are provided with data that indicate the wear of components (hardware) (including so-called operating data).

From a subsystem diagnostics perspective, the following procedures are relevant:

- Operational signaling
- Workshop diagnostics
- Functional status
- Operational status
2.6.1 Operational signaling
Operational signaling (also referred to as: diagnostics for the operating personnel) designates the detection of fault conditions or functional or service restrictions that are relevant for the operating personnel.
Operational signaling supports the train personnel (e.g. the train driver or train conductor) during the operation of the vehicle. Operational signaling has a functional design and is oriented at functional restrictions. The train personnel is informed of functions restrictions of the vehicle and is provided with assistance (corrective action) in order to minimize any operational restrictions (restrictions for passenger service).

2.6.2 Workshop diagnostics
Workshop diagnostics (also referred to as: diagnostics for the maintenance personnel) designates the detection of conditions that are relevant for the maintenance or repair by the maintenance personnel. In this context, diagnostics covers the necessary activities for inspection (also remote if applicable), for recording, storage and transfer of the data if necessary.
Workshop diagnostics supports the maintenance personnel with the elimination of technical defects (repair) that have occurred during operation. In this case, workshop diagnostics focuses on the smallest replaceable unit (KTE). Workshop diagnostics supports the maintenance personnel with troubleshooting in case of restriction cannot be clearly located.
Workshop diagnostics also provides operational data that can be used to plan preventive maintenance activities.

2.6.3 Operational and functional status
The operational status of a function / subfunction indicates the general capability of a functional unit to perform the respective task. The status model is standardized in the Sibas PN system for all (sub)functions (status examples include NORMAL, FAILURE etc.). On a display (HMI), display objects (pictograms) are used to inform about the general capability of individual functions to perform a task. The display objects are assigned a specific color as a function of the availability of the function (project-specific definition). This mechanism is illustrated in the figure below in an informal and exemplary manner:

![Figure 21 Operational status and visualization](image)

Each function/subfunction of a subsystem must provide the following status:
- **Functional status of a (sub)function** that indicates the execution status of a task. The range of values of the functional status is (sub)function-specific, i.e. the status models will usually differ from (sub)function to (sub)function.
• **Operational status of a (sub)function** that indicates the general capability of a functional unit to perform the respective task. The range of values of the operational status is not (sub)function-specific. The underlying status model is defined in chapter 2.6.3.1.

A functional unit is defined as the technical implementation of one or multiple functions. The figure below illustrates the relationships in the following UML class diagram:

![UML class diagram](image)

*Figure 22: Relationship between function and operational status*

A function is defined as the implementation of the capability of a system or subsystem that solves the task or a set of tasks. A function may in turn be subdivided into subfunctions. Typically, a system / subsystem implements its functionality as a number of different subfunctions.

Each implemented function / subfunction must inform about the degree of its availability.

**2.6.3.1 Definition of the functional status**

The functional status depends on the viewed (sub)function and is defined in the context of the system architecture descriptions for the individual functions, such as the system architecture for external lighting. The functional status is part of the interfaces that are defined there.
2.6.3.2 Definition of operational status

The implementations of the operational status are standardized in the Sibas PN system and defined in the figure below. These operational status are transmitted as Unsigned8 values via the interface that is described in chapter 2.6.6.

![Figure 23: Operational status of functional unit](image)

2.6.3.2.1 General System Mode

The table below describes the individual states of the "General System Mode".

<table>
<thead>
<tr>
<th>Name of operational status</th>
<th>Description</th>
<th>Current implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFF</td>
<td>Functional unit is switched off.</td>
<td>Yes</td>
</tr>
<tr>
<td>NORMAL</td>
<td>Functional unit is ready.</td>
<td>Yes</td>
</tr>
<tr>
<td>DEGRADED</td>
<td>The functional unit is degraded, i.e. one or multiple functional restrictions are present.</td>
<td>Yes</td>
</tr>
<tr>
<td>OUT_OF_ORDER</td>
<td>The functional unit has failed but communications with it is still possible.</td>
<td>Yes</td>
</tr>
<tr>
<td>INITIALIZATION</td>
<td>The functional unit is starting up.</td>
<td>Yes</td>
</tr>
<tr>
<td>READY_FOR_SHUTDOWN</td>
<td>The functional unit can be switched off.</td>
<td>Yes</td>
</tr>
<tr>
<td>SHUTDOWN</td>
<td>The functional unit is switched off.</td>
<td>Yes</td>
</tr>
<tr>
<td>State</td>
<td>Description</td>
<td>Status</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>ISOLATED</td>
<td>The functional unit has been isolated and is waiting for integration. This can usually only be done by maintenance personnel.</td>
<td>Yes</td>
</tr>
<tr>
<td>STANDBY</td>
<td>Status for a non-active, redundant functional unit.</td>
<td>Yes</td>
</tr>
<tr>
<td>SW_DOWNLOAD</td>
<td>This status is assumed when a SW download is performed on the functional unit.</td>
<td>No</td>
</tr>
<tr>
<td>RESET</td>
<td>This status is assumed when a system reset is performed on the functional unit.</td>
<td>No</td>
</tr>
<tr>
<td>MISSING_PRECONDITION</td>
<td>A precondition for proper operation of the functional unit is missing.</td>
<td>Yes</td>
</tr>
<tr>
<td>UNKNOWN</td>
<td>This status is assumed when it cannot be determined from the outside which operational status a functional unit has assumed. This status exists e.g. in case of a communication failure.</td>
<td>Yes</td>
</tr>
<tr>
<td>KNOWN</td>
<td>This status is assumed when it can be determined from the outside which operational status a functional unit has assumed.</td>
<td>Yes</td>
</tr>
</tbody>
</table>

*Table 1 States of the General System Mode*
2.6.3.2.2 Redundancy Mode
The table below describes the individual states of the "General System Mode".

<table>
<thead>
<tr>
<th>Name of operational status</th>
<th>Description</th>
<th>Current implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTIVE</td>
<td>Status for an active, redundant functional unit.</td>
<td>Yes</td>
</tr>
<tr>
<td>PASSIVE</td>
<td>Status for a non active, redundant functional unit.</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table: 2 States of the Redundancy Mode

2.6.3.2.3 Test Service Mode
The table below describes the individual states of the "Test Service Mode".

<table>
<thead>
<tr>
<th>Name of operational status</th>
<th>Description</th>
<th>Current implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>NORMAL OPERATION</td>
<td>This status is assumed when the functional unit is operating in normal mode.</td>
<td>Yes</td>
</tr>
<tr>
<td>TEST / SERVICE TEST</td>
<td>This status is assumed when a test run is performed on the functional unit.</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table: 3 States of the Test Service Mode
2.6.3.2.4 State machine

Changes of the functional states are implemented by means of transitions that cause a switchover when events or conditions occur. The associated state machine is illustrated in the following state diagram:

![State diagram for operational status](image)

**Note:** In the implementation of the state machine and in the design of the associated interface, it must be considered that more states may be added in subsequent configuration stages. In the operational status state machine, a distinction is made between an external and an internal view across the two regions of the AND-enhanced state **Operational status**. The external view is used to indicate whether the operational status of the functional unit can be determined from the outside (e.g. an operational status of a functional unit via the SP CS), i.e.:

- If the operational status of the functional unit can be determined, the resulting state will be **KNOWN**.
- If the operational status of the functional unit cannot be determined, the resulting state will be **UNKNOWN** (e.g. if communications with the functional unit is disturbed).

Within the context of diagnostics, the substates of the internal view are particularly interesting, provided they are known (state **KNOWN**). The AND-enhanced state **ON** is subdivided into three orthogonal regions:

- The first orthogonal region shows the general status of task execution.
- The states in the second orthogonal region indicate whether a test is running or whether the function unit is operating in normal mode.
• The states in the third orthogonal region indicate whether the functional unit is active or passive if this unit has a redundant design.

The states in this region indicate for a functional unit whether it is functional without restrictions (NORMAL), whether a precondition for proper operation is missing (MISSING_PRECONDITION), etc. The status MISSING_PRECONDITION is used for the prevention of consequential faults, refer to additional information in the chapters about operational signaling and workshop diagnostics (see chapter 2.6.6.4).

Note: A functional unit must not necessarily assume all of the mentioned operational states, e.g. a non-redundant functional unit will never enter the PASSIVE state. It is not defined within the context of this description under which conditions a functional unit will assume a certain state. This is the task of the subsystem-specific system architecture descriptions of the individual functions. Therefore, the events and conditions that result in the switching of transitions are defined within the context of the system architecture descriptions.
2.6.4 Diagnostics example

This section informally illustrates the use of the functional status and operational status for the example of the door control function.

**Note:** The scenarios described here are used to illustrate the concept; the system architecture description for the door control will always and exclusively be binding for the door control function.

For the scenarios described below, the following partition into functional units and functions is assumed (extract):

- The functional unit of the door control function is the door control unit. This is assigned an operational status.
- A function that is responsible for the opening and closing of the passenger door.
- A function that is responsible for the extending and retracting of the extendible door step.

The following functional states are defined for the functions:

- the functional status of the passenger door (*CLOSED AND LOCKED*, *MOVING*, *OPEN*, *LOCKED*, etc).
- the functional status of the extendible door step (*RETRACTED*, *NOT USED*, *EXTENDED*, *LOCKED*, etc.).

The figure below illustrates the relationships again at the instance level:

![Diagram](https://example.com/diagram.png)

*Figure 25 Functions, functional units, operational status and functional status for the example of door control*
2.6.4.1 Scenario 1 – Operational and functional status for an open door

The train has entered a station. For one door of a car that has already been opened, the states of the two subfunctions (after opening of the door by the passengers) are as follows (it is assumed that the extendible door steps need to be extended in this station):

- Functional status of passenger door: OPEN
- Functional status of extendible door step: EXTENDED
- Operational status of door control function: NORMAL

2.6.4.2 Scenario 2 – Operational and functional status for a defective door

Scenario 2: Operational and functional status for a defective door

The train has entered a station and the door opens but cannot close again due to a technical defect. The functional and operational status of the door is as follows:

- Functional status of passenger door: OPEN
- Functional status of extendible door step: EXTENDED
- Operational status of door control function: OUT_OF_ORDER

2.6.4.3 Scenario 3 – Operational and functional status for an isolated door

A door was manually locked and blocked after a technical defect had occurred. The operational and functional status for the door is as follows:

- Functional status of passenger door: CLOSED
- Functional status of extendible door step: RETRACTED
- Operational status of door control function: ISOLATED
2.6.5 Diagnostic truth

The system must apply the quality features according to UIC 557 (equivalent to IEC 60706-5) to the design and demonstrate them for the supplied products (system). The focus is on a fault detection, also referred to as the detectable fault rate, of 98% for non safety-related faults.

\[
\text{Detectable fault rate } EF = \frac{\text{Number of detectable faults of system}}{\text{Number of all possible faults of system}}
\]

According to the above standard, safety-related faults must all be detected! Detectable faults include both faults that are automatically diagnosed via the system diagnosis and faults that are detected by staff during service or maintenance, e.g. in visual inspections, functional tests, trials.

In order to prove that the required fault detection rate is achieved, a failure analysis shall be performed for the system to be supplied and to be submitted to the contracting party for DF. For the failure analysis, the supplier shall perform the following steps and present them in a clearly structured tabular format:

1. In the sense of an FMEA, list the components (LRUs on the vehicle) and individual functions of the system and present the corresponding possible faults and their consequences.

2. Analyze the kind of fault detection and give an accurate description:
   - Fault detection through diagnosis: These diagnostic events form the basis for the configuration of the diagnosis.
   - Fault detection in the context of operation or maintenance by means of defined inspections and tests. The supplier shall include these inspections and tests in the operating and maintenance documentation.
   - No fault detection; fault is not detected.

3. Allocation of component failures to diagnostic messages.

Another indicator for the diagnostic quality is the diagnostic truth WD according to VDV 166-2.
Diagnostic truth $WD = \frac{\text{Number of true faults}}{\text{Number of reported faults}}$ in %

A value of 75% is defined, which shall also be demonstrated by the supplier.
2.6.6 Diagnostic generation in the subsystem

![Diagram of Diagnostic Generation](image_url)

**Figure 26: Diagnostic generation**

**Note:** Similar to workshop diagnostic signaling, the log data of the subsystem are forwarded by means of the same mechanism (see chapter 2.6.6) or the same record. This process uses dedicated CSV files for log data, presignaling (log data) and the corresponding parameter file "Log presignaling".

### 2.6.6.1 Functional status

The functional status is transmitted by means of the bit interface via the cyclic data according to the respective functional interface.

### 2.6.6.2 Operational status

The states of the operational status (General System Mode, Redundancy Mode, Test Service Mode) are transmitted via the cyclic data according to the definition in chapter 2.6.3.2 as Unsigned8 values each.

### 2.6.6.3 Operational signaling

Events for operational signaling are transmitted by means of the bit interface via the cyclic data.
For each subsystem, up to 32 individual messages are defined per (sub)function of a functional unit (= including subsystem) according to their functional interface.

### 2.6.6.4 Diagnostic buffer for workshop diagnostic messages

Workshop diagnostic messages are stored on the subsystem in a remanent circular buffer. When the end of the data buffer is reached, the process is restarted and the contents are overwritten (circular buffer), with a record that has a "Records overwritten" code being inserted. The data buffer should be designed that a minimum of 200 workshop diagnostic messages can be stored.

If the workshop diagnostic message is "presignaling-relevant", it will also be entered in the data buffer for presignaling. The layout of this diagnostic buffer and of the records is the same as for the workshop diagnostic buffer. It must be possible to buffer 50 presignaling-relevant workshop diagnostic messages.

The filename of the workshop diagnostic file is selected so that it is assignment can be subsequently determined. With each reading operation of the workshop diagnostic file, a description file (schema.ini, see chapter 2.6.7) is also transmitted. This ensures that the contents of the CSV files can be properly evaluated.

### 2.6.6.5 Diagnostic buffer for log data

Log data are transmitted using the same mechanism that is described in chapter 2.6.6.4 for workshop diagnostic messages.

### 2.6.6.6 Initialization data for diagnostic messages

In order to be able to provide the records of workshop diagnostic messages with the necessary information, it is required to provide such information to the subsystem.

This information is provided to the subsystem by means of different mechanisms, at different times (via cyclic data, or during the initialization of the PROFINET IO device) and from different sources.

The table below shows which information is provided to the subsystem for this purpose (definition see chapter 2.6.6.7).

<table>
<thead>
<tr>
<th>Data Source</th>
<th>Data Transfer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Vehicle Control System (CS)</td>
</tr>
<tr>
<td>MissingPrecondition</td>
<td>X</td>
</tr>
<tr>
<td>TrainOperationID</td>
<td>X</td>
</tr>
<tr>
<td>CarNo</td>
<td>X</td>
</tr>
<tr>
<td>GeneralTrainMode</td>
<td>X</td>
</tr>
<tr>
<td>TrainMode</td>
<td>X</td>
</tr>
<tr>
<td>TrainSubMode</td>
<td>X</td>
</tr>
<tr>
<td>OtherTrainModes</td>
<td>X</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---</td>
</tr>
<tr>
<td>NTP Client Parameter</td>
<td>X</td>
</tr>
</tbody>
</table>

*Table 4 Initialization data for diagnostic messages*
### Layout of workshop diagnostic messages (record)

The table below shows the layout of a workshop diagnostic record. The data types contained herein comply with the Profile Guidelines for PROFINET [7].

Similar to workshop diagnostic signaling, the log data of the subsystem are forwarded by means of the same mechanism (see chapter 2.6.6) or the same record.

<table>
<thead>
<tr>
<th>Data field</th>
<th>Description</th>
<th>Format</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>VersionID</td>
<td>Version of the header data structure 1st byte / 2nd byte Major / Minor</td>
<td>Unsigned16</td>
<td>2 bytes</td>
</tr>
<tr>
<td>DiagnosticClass</td>
<td>0 = Corrective 1 = Preventive</td>
<td>Unsigned8</td>
<td>1 byte</td>
</tr>
<tr>
<td>DateTime</td>
<td>Time in UTC with 1 ms resolution.</td>
<td>NetworkTime</td>
<td>8 bytes</td>
</tr>
<tr>
<td>SequenceNo</td>
<td>Cons. no. of diagnostic record</td>
<td>Unsigned16</td>
<td>2 bytes</td>
</tr>
<tr>
<td>HeaderLength</td>
<td>Length of diagnostic header in bytes</td>
<td>Unsigned8</td>
<td>1 byte</td>
</tr>
<tr>
<td>DataLength</td>
<td>Length of environmental data in bytes</td>
<td>Unsigned8</td>
<td>1 byte</td>
</tr>
<tr>
<td>VendorID</td>
<td>16-bit ID that forms a unique reference to the vendor company. This is</td>
<td>Unsigned16</td>
<td>2 bytes</td>
</tr>
<tr>
<td></td>
<td>assigned by the PNO for each vendor. This means that a vendor needs this</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ID only once. (information derived from GSD)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DeviceID</td>
<td>The Device_ID permits the detailed distinction of the IO field device types.</td>
<td>Unsigned16</td>
<td>2 bytes</td>
</tr>
<tr>
<td></td>
<td>It can be defined for each supplier (information derived from GSD)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DeviceSerialNo</td>
<td>Serial number of device</td>
<td>VisibleString</td>
<td>25 bytes</td>
</tr>
<tr>
<td>NameOfStation</td>
<td>Unique device name with reference to equipment code</td>
<td>VisibleString</td>
<td>25 bytes</td>
</tr>
<tr>
<td>ModuleName</td>
<td>Name of faulty module. (information derived from GSD)</td>
<td>VisibleString</td>
<td>25 bytes</td>
</tr>
<tr>
<td>ModulSerialNo</td>
<td>Serial number of module (if available)</td>
<td>VisibleString</td>
<td>25 bytes</td>
</tr>
<tr>
<td>HardwareRelease</td>
<td>Hardware release of module according to GSD</td>
<td>VisibleString</td>
<td>4 bytes</td>
</tr>
<tr>
<td>Data field</td>
<td>Description</td>
<td>Format</td>
<td>Length</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------</td>
<td>--------</td>
</tr>
<tr>
<td>SoftwareRelease</td>
<td>Software release of module according to GSD information</td>
<td>VisibleString</td>
<td>4 bytes</td>
</tr>
</tbody>
</table>
| ApplID               | Version of the subsystem application (if available in addition to Software_Release)  
|                      | 1st byte / 2nd byte / 3rd byte Major / Minor / Patch                          | Unsigned32  | 4 bytes |
| DiagnosticCode       | Diagnostic event no. of subsystem                                            | Unsigned16  | 2 bytes |
| EventState           | Message incoming / outgoing  
|                      | 0 = Message outgoing  
|                      | 1 = Message incoming                                                         | Unsigned8   | 1 byte  |
| DiagnosticType       | Record type  
|                      | 0003h workshop message  
|                      | 0004h log data                                                              | Unsigned8   | 1 byte  |
| MissingPrecondition  | 0 = Precondition exists  
|                      | 1 = Precondition does not exist                                               | Unsigned8   | 1 byte  |
| TrainOperationID      | Vehicle no., e.g. train number (ICE745) or operator-specific definition.     | Unsigned16  | 2 bytes |
| CarNo                | UIC ID of car according to UIC 438, operator-specific definition if required.  
|                      | When UIC is used:  
|                      | - major tech. features (4 digits)  
|                      | - Number within class (3 digits)  
<p>|                      | - Check digit (2 digits)                                                      | VisibleString | 9 bytes |
| GeneralTrainMode     | General train status (normal operation, service, maintenance). This status affects authorizations to make changes to the system (download of SW permitted/prohibited, forcing of signals permitted/prohibited). It applies to the entire train. | Unsigned8   | 1 byte  |
| TrainMode            | Operational status of train. This indicates the main states of the train from an operator perspective. | Unsigned8   | 1 byte  |
| TrainSubMode         | Operational substates of the train from an operator                           | Unsigned8   | 1 byte  |</p>
<table>
<thead>
<tr>
<th>Data field</th>
<th>Description</th>
<th>Format</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>OtherTrainModes</td>
<td>Other states that may be active independently of the operational status of the train.</td>
<td>Unsigned8</td>
<td>1 byte</td>
</tr>
<tr>
<td>OccVersionID</td>
<td>Version of environmental data structure 1st byte / 2nd byte Major / Minor</td>
<td>Unsigned16</td>
<td>2 bytes</td>
</tr>
<tr>
<td>OccDataType</td>
<td>Type of environmental data structure 0 = No transfer of environmental data &gt; 0 = Type of environmental structure</td>
<td>Unsigned8</td>
<td>1 byte</td>
</tr>
<tr>
<td>OccDataLen</td>
<td>Length of environmental data structure in bytes (max. 100)</td>
<td>Unsigned8</td>
<td>1 byte</td>
</tr>
<tr>
<td>OccData</td>
<td>Device-specific environmental data in Base64 format</td>
<td>Environment-specific</td>
<td>Environment-specific</td>
</tr>
</tbody>
</table>

*Table 5 Layout of workshop diagnostic messages*
2.6.6.8 Description file

For each transfer of the workshop diagnostic buffer (CSV file), a description file shall always be provided that contains information about the data types for each column as well as a CSV file with the global information. Depending on the contents, it is then possible to transfer additional files.

To permit the evaluation of a CSV file, each file must be assigned an entry in the associated description file.

Each section in the description file consists of section names with general keys and the associated values. This is followed by the keys for the existing columns in the CSV file.

2.6.7 CSV file

The workshop diagnostic messages are stored in a CSV file according to the data layout (chapter 2.6.6.6).

2.6.7.1 Introduction

In this chapter, the file format for the export of diagnostic data is defined. The format conventions must be observed in order to enable further processing (e.g. conversion of the exported files into other formats) within the context of subsequent processing steps.

The files are structured by using a CSV (Comma Separated Value) format that is defined in detail in the chapters below. The idea of using XML files was abandoned due to the resulting file size (specification).

2.6.7.2 CSV format

A CSV file is subdivided – similar to a table – into lines and columns. A cell is an entry that consists of exactly one line and one column, i.e. each cell is assigned exactly one line and one column. The figure below illustrates the relationships.

![Figure 27: Layout of a CSV file (lines, columns, cells)]
In the Sibas PN system, a CSV file (that is designed to permit the export of diagnostic data) is primarily a text file that consists of Unicode characters. These characters are encoded with UTF-8, following these two rules:

- To describe the structure consisting of rows, columns and cells, only a subset of the UTF-8 encoded character set (Unicode) is used that is compatible with the ASCII character set (this comprises the Unicode characters \texttt{U+0000} – \texttt{U+007F}).
- The contents of a cell is a UTF-8 encoded string. This may use the full Unicode character set.

The structure of a CSV file (lines, columns and cells) is defined according to the following rules:

- Lines are terminated by a line feed (abbreviated as \texttt{LF}).
- However, the last line of a file must not necessarily be terminated by a line feed.
- Each line can contain any number of cells whose total number, however, is fixed for each file. This directly results in all lines of a file having the same number of cells.
- Cells within a line are separated from each other by the comma separator.
- Each cell can comprise any number of UTF-8 encoded characters and is terminated by the comma separator. \texttt{No} separator is used at the start (before the first cell) and the end of a line (after the last cell).

- Example: \texttt{aaa, bbb, cccLF}
- The contents of individual cells can be written in double quotation marks.
  - Example: \texttt{"aaa", "bbb", "ccc"LF}
- Each cell must be written in double quotation marks \texttt{""} if the contents comprises a separator.
  - Example: \texttt{aaa, "bb, b", cccLF"d", "e", "f}
- Each cell whose contents comprises a double quotation mark, shall be written in double quotation marks. Double quotation marks which are part of the contents of a cell must be doubled (Escape sequence). This indicates that this is not the end of a string in a cell.
  - Example: \texttt{aaa, "b""bb", cccLF"d", "e", "f}
- Each cell whose contents comprises a line feed shall be written in double quotation marks.
  - Example: \texttt{aaa, "bLFbb", cccLF"d", "e", "f}

The structure of the CSV files (formal implementation of plain text rules, see above) is defined by the EBNF syntax, using the standardized notation from \texttt{<VERWEIS3>}. Valid terminal symbols are UTF-8 encoded character in the standard notation (e.g. \texttt{U+0022} for the double quotation mark \\texttt{"}):

- \texttt{FILE = RECORD, [LF RECORD], [LF]}
- \texttt{RECORD = FIELD, [SEPARATOR, FIELD]}
- \texttt{FIELD = ESCAPED | NONESCAPED}
- \texttt{ESCAPED = DQUOTE, [UTFCHARACTER_NO_DQUOTE | 2*DQUOTE], DQUOTE}
• \textit{NONESCAPED} = \{\texttt{UTFCHARACTER\_NO\_DQUOTE}\}
• \textit{DQUOTE} = \texttt{U+0022}
• \textit{LF} = \texttt{U+000A}
• \textit{SEPARATOR} = \texttt{U+002C}

\texttt{UTFCHARACTER\_NO\_DQUOTE} designates a UTF-8 encoded Unicode character, with the exception of the double quotation mark (this is to be doubled as required, see also distinction of rules into \textit{ESCAPED} and \textit{UNESCAPED}).

\textbf{Note:} Spaces (also called white spaces) are always part of a cell (\textit{FIELD} in the syntax).

\textbf{Note:} This format is mandatory, whereas other CSV formats (derivates) are not supported.
3 PNO Certification / Standardization

3.1 PROFINET compliance

For IO devices, the PROFINET compliance is to be determined in a PNO test lab and a Device ID number is to be requested. If successful, a test certificate can be requested that is required for use in vehicle projects of SIEMENS TS.

Figure 3-1: PNO certification for a PROFINET IO device

For more information about PNO and test labs, please visit

http://www.profibus.com/pn/applications/certification/
4 Test and Commissioning

4.1 PC test platform

For commissioning and testing of a controller with PROFINET interface (e.g. IM 104 RT), a PROFINET PC module can be used as the partner station. The CP1616 module from Siemens A&D SC can be used to implement an IO controller that communicates with an IO device.

This permits to establish a PROFINET link to the test PC and to test the GSDML of the controller (IO device).

Figure 4-1: Test setup with PC module and IO device using IM 104 RT TS

The required software and documentation is available along with the PC module CP1616.
5 Further information

5.1 Useful links

The following table contains a summary of links to further information:

<table>
<thead>
<tr>
<th>Subject</th>
<th>Organization</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNO</td>
<td>PNO</td>
<td><a href="http://www.profibus.com/pn/">http://www.profibus.com/pn/</a></td>
</tr>
<tr>
<td>PROFINET Technology</td>
<td>PNO</td>
<td><a href="http://www.profibus.com/pn/technology/">http://www.profibus.com/pn/technology/</a></td>
</tr>
<tr>
<td>PROFINET newsletter</td>
<td>PNO</td>
<td><a href="http://www.profibus.com/pi/events/newsletter/">http://www.profibus.com/pi/events/newsletter/</a></td>
</tr>
<tr>
<td>PROFINET certification</td>
<td>PNO</td>
<td><a href="http://www.profibus.com/pn/applications/certification/">http://www.profibus.com/pn/applications/certification/</a></td>
</tr>
<tr>
<td>PNO Regional Support</td>
<td>PNO</td>
<td><a href="http://www.profibus.com/pn/rpa/">http://www.profibus.com/pn/rpa/</a></td>
</tr>
<tr>
<td>PROFINET Competence centers</td>
<td>PNO</td>
<td><a href="http://www.profibus.com/pn/support/pccs/">http://www.profibus.com/pn/support/pccs/</a></td>
</tr>
<tr>
<td>PROFINET Competence center ComDec</td>
<td>ComDec</td>
<td><a href="https://www.automation.siemens.com/_de/we-get-it-done/comdec.htm">https://www.automation.siemens.com/_de/we-get-it-done/comdec.htm</a></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;&lt;Login required&gt;&gt;</td>
</tr>
<tr>
<td>PROFINET product guide</td>
<td>PNO</td>
<td><a href="http://www.profibus.com/pn/applications/">http://www.profibus.com/pn/applications/</a></td>
</tr>
<tr>
<td>ERTEC</td>
<td>Siemens A&amp;D</td>
<td><a href="http://www.automation.siemens.com/microsite/ertec/html_76/ertec_e.swf">http://www.automation.siemens.com/microsite/ertec/html_76/ertec_e.swf</a></td>
</tr>
</tbody>
</table>
## 5.2 Further references

- The Rapid Way to PROFINET, Manfred Popp, Karl Weber, PNO Karlsruhe, PNO Order No.: 4182 (english)
- Das PROFINET IO-Buch, Manfred Popp, Hüthig-Verlag Heidelberg, ISBN 3-7785-2966-8 (german)
## List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASIC</td>
<td>Application Specific Integrated Circuit</td>
</tr>
<tr>
<td>CS</td>
<td>Control System</td>
</tr>
<tr>
<td>CSV</td>
<td>Character Separated Values, Comma Separated Values or Colon Separated Values</td>
</tr>
<tr>
<td>DP</td>
<td>Decentralized Periphery</td>
</tr>
<tr>
<td>ERTEC</td>
<td>Enhanced Realtime Ethernet Controller, Ethernet switch Communication ASIC of A&amp;D for PROFINET</td>
</tr>
<tr>
<td>GSD</td>
<td>Generic Station Description</td>
</tr>
<tr>
<td>GSDML</td>
<td>Generic Station Description Markup Language</td>
</tr>
<tr>
<td>HMI</td>
<td>Human-Machine Interface</td>
</tr>
<tr>
<td>http</td>
<td>Hypertext Transport Protocol</td>
</tr>
<tr>
<td>IEC</td>
<td>International Electrotechnical Commission</td>
</tr>
<tr>
<td>IO</td>
<td>Input/Output Eingangs-/Ausgangs- ...</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>IRT</td>
<td>Isochronous Realtime, Protocol of PROFINET IO</td>
</tr>
<tr>
<td>IT</td>
<td>Information Technology</td>
</tr>
<tr>
<td>LAN</td>
<td>Local Area Network</td>
</tr>
<tr>
<td>LLDP</td>
<td>Link-Layer Discovery Protocol, IEEE802.1ab</td>
</tr>
<tr>
<td>NCM PC</td>
<td>NCM PC is a version of the SIMATIC engineering system STEP 7 that is tailored to PC configuration.</td>
</tr>
<tr>
<td>NRT</td>
<td>Non-Realtime</td>
</tr>
<tr>
<td>PNO</td>
<td>Profibus Nutzer Organisation [1]</td>
</tr>
<tr>
<td>RT</td>
<td>Realtime, Protocol of PROFINET IO</td>
</tr>
<tr>
<td>Sibas 32</td>
<td>Siemens Railway Automation System 32-bit technology</td>
</tr>
<tr>
<td>Sibas PN</td>
<td>Siemens Railway Automation System PROFINET</td>
</tr>
<tr>
<td>SIMATIC RAIL</td>
<td>Development project for a railway-proof automation system based on SIMATIC</td>
</tr>
<tr>
<td>SNMP</td>
<td>Simple Network Monitoring Protocol</td>
</tr>
<tr>
<td>RT</td>
<td>Realtime, Protocol of PROFINET IO</td>
</tr>
<tr>
<td>WTB</td>
<td>Wire Train Bus</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
</tbody>
</table>
7 References