

INTERNATIONAL STANDARD

NORME INTERNATIONALE

**OPC Unified Architecture –
Part 15: Safety**

**Architecture unifiée OPC –
Partie 15: Sécurité**



THIS PUBLICATION IS COPYRIGHT PROTECTED

Copyright © 2025 IEC, Geneva, Switzerland

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either IEC or IEC's member National Committee in the country of the requester. If you have any questions about IEC copyright or have an enquiry about obtaining additional rights to this publication, please contact the address below or your local IEC member National Committee for further information.

Droits de reproduction réservés. Sauf indication contraire, aucune partie de cette publication ne peut être reproduite ni utilisée sous quelque forme que ce soit et par aucun procédé, électronique ou mécanique, y compris la photocopie et les microfilms, sans l'accord écrit de l'IEC ou du Comité national de l'IEC du pays du demandeur. Si vous avez des questions sur le copyright de l'IEC ou si vous désirez obtenir des droits supplémentaires sur cette publication, utilisez les coordonnées ci-après ou contactez le Comité national de l'IEC de votre pays de résidence.

IEC Secretariat
3, rue de Varembe
CH-1211 Geneva 20
Switzerland

Tel.: +41 22 919 02 11
info@iec.ch
www.iec.ch

About the IEC

The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications

The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigendum or an amendment might have been published.

IEC publications search - webstore.iec.ch/advsearchform

The advanced search enables to find IEC publications by a variety of criteria (reference number, text, technical committee, ...). It also gives information on projects, replaced and withdrawn publications.

IEC Just Published - webstore.iec.ch/justpublished

Stay up to date on all new IEC publications. Just Published details all new publications released. Available online and once a month by email.

IEC Customer Service Centre - webstore.iec.ch/csc

If you wish to give us your feedback on this publication or need further assistance, please contact the Customer Service Centre: sales@iec.ch.

IEC Products & Services Portal - products.iec.ch

Discover our powerful search engine and read freely all the publications previews, graphical symbols and the glossary. With a subscription you will always have access to up to date content tailored to your needs.

Electropedia - www.electropedia.org

The world's leading online dictionary on electrotechnology, containing more than 22 500 terminological entries in English and French, with equivalent terms in 25 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

A propos de l'IEC

La Commission Electrotechnique Internationale (IEC) est la première organisation mondiale qui élabore et publie des Normes internationales pour tout ce qui a trait à l'électricité, à l'électronique et aux technologies apparentées.

A propos des publications IEC

Le contenu technique des publications IEC est constamment revu. Veuillez vous assurer que vous possédez l'édition la plus récente, un corrigendum ou amendement peut avoir été publié.

Recherche de publications IEC -

webstore.iec.ch/advsearchform

La recherche avancée permet de trouver des publications IEC en utilisant différents critères (numéro de référence, texte, comité d'études, ...). Elle donne aussi des informations sur les projets et les publications remplacées ou retirées.

IEC Just Published - webstore.iec.ch/justpublished

Restez informé sur les nouvelles publications IEC. Just Published détaille les nouvelles publications parues. Disponible en ligne et une fois par mois par email.

Service Clients - webstore.iec.ch/csc

Si vous désirez nous donner des commentaires sur cette publication ou si vous avez des questions contactez-nous: sales@iec.ch.

IEC Products & Services Portal - products.iec.ch

Découvrez notre puissant moteur de recherche et consultez gratuitement tous les aperçus des publications, symboles graphiques et le glossaire. Avec un abonnement, vous aurez toujours accès à un contenu à jour adapté à vos besoins.

Electropedia - www.electropedia.org

Le premier dictionnaire d'électrotechnologie en ligne au monde, avec plus de 22 500 articles terminologiques en anglais et en français, ainsi que les termes équivalents dans 25 langues additionnelles. Egalement appelé Vocabulaire Electrotechnique International (IEV) en ligne.

INTERNATIONAL STANDARD

NORME INTERNATIONALE

**OPC Unified Architecture –
Part 15: Safety**

**Architecture unifiée OPC –
Partie 15: Sécurité**

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

COMMISSION
ELECTROTECHNIQUE
INTERNATIONALE

ICS 25.040.40

ISBN 978-2-8327-0212-3

Warning! Make sure that you obtained this publication from an authorized distributor.
Attention! Veuillez vous assurer que vous avez obtenu cette publication via un distributeur agréé.

CONTENTS

FOREWORD.....	6
INTRODUCTION.....	8
1 Scope.....	9
2 Normative references.....	9
3 Terms, definitions, symbols, abbreviated terms and conventions.....	10
3.1 Terms and definitions	10
3.1.1 Common terms and definitions	10
3.1.2 Additional terms and definitions.....	12
3.2 Symbols and abbreviated terms.....	14
3.2.1 Abbreviated terms from IEC 61784-3	14
3.2.2 Additional symbols and abbreviated terms	15
3.3 Conventions.....	15
3.3.1 General conventions	15
3.3.2 Conventions for requirements numbering.....	15
3.3.3 Conventions in state machines	16
4 Overview of OPC UA Safety.....	16
4.1 General.....	16
4.2 Implementation aspects.....	16
4.3 Features	17
4.4 Security policy	17
5 General	18
5.1 External documents providing specifications for the profile	18
5.2 Safety functional requirements	18
5.3 Safety measures	18
5.4 Safety communication layer structure	19
5.5 Requirements for CRC calculation	21
6 Safety communication layer services.....	21
6.1 General.....	21
6.2 Information models.....	22
6.2.1 General	22
6.2.2 Object and ObjectType Definitions.....	22
6.2.3 DataType definition	34
6.2.4 SafetyProvider version	38
6.2.5 DataTypes and length of SafetyData.....	38
6.2.6 Connection establishment	38
6.3 Service interfaces	38
6.3.1 Overview	38
6.3.2 OPC UA Platform interface (OPC UA PI)	39
6.3.3 SafetyProvider interfaces	39
6.3.4 SafetyConsumer interfaces	46
6.3.5 Cyclic and acyclic safety communication	53
6.3.6 Principle for "application variables with qualifier"	53
6.4 Diagnostics	53
6.4.1 General	53
6.4.2 Diagnostics messages of the SafetyConsumer.....	54
6.4.3 Method ReadSafetyDiagnostics of the SafetyProvider.....	56

7	Safety communication layer protocol	56
7.1	General.....	56
7.2	SafetyProvider and SafetyConsumer	56
7.2.1	SPDU formats.....	56
7.2.2	Behaviour	58
7.2.3	Subroutines	76
8	Safety communication layer management.....	82
8.1	General.....	82
8.2	Safety function response time part of communication	82
9	System requirements (SafetyProvider and SafetyConsumer)	84
9.1	Constraints on the SPDU parameters	84
9.1.1	SafetyBaseID and SafetyProviderID	84
9.1.2	SafetyConsumerID	85
9.2	Initialization of the MNR in the SafetyConsumer.....	86
9.3	Constraints on the calculation of system characteristics	86
9.3.1	Probabilistic considerations (informative).....	86
9.3.2	Safety related assumptions (informative)	88
9.4	PFH and PFD values of a logical safety communication link	88
9.5	Safety manual	89
9.6	Indicators and displays.....	90
10	Assessment.....	90
10.1	Safety policy	90
10.2	Obligations.....	91
10.3	Index of requirements (informative)	91
11	Profiles and conformance units	94
12	Namespaces	94
12.1	Namespace metadata.....	94
12.2	Handling of IEC 62541 namespaces	95
Annex A (normative)	Safety namespace and mappings	96
Annex B (informative)	Additional information	97
B.1	CRC calculation using tables, for the polynomial 0xF4ACFB13.....	97
B.2	Use cases	98
B.2.1	Unidirectional communication	98
B.2.2	Bidirectional communication	99
B.2.3	Safety multicast	99
B.3	Use cases for operator acknowledgment.....	100
B.3.1	Explanation.....	100
B.3.2	Use case 1: unidirectional communication and OA on the SafetyConsumer side	100
B.3.3	Use case 2: bidirectional communication and dual OA	101
B.3.4	Use case 3: bidirectional communication and single, one-sided OA	101
B.3.5	Use case 4: bidirectional communication and single, two-sided OA	102
Annex C (informative)	Information for assessment	103
Bibliography		104

Figure 1 – Relationships of OPC UA safety with other standards.....8

Figure 2 – Safety layer architecture.....20

Figure 3 – Server Objects for OPC UA Safety.....	24
Figure 4 – Instances of Server Objects for this document.....	25
Figure 5 – Safety multicast with three recipients using IEC 62541 PubSub.....	31
Figure 6 – Safety parameters for the SafetyProvider and the SafetyConsumer	32
Figure 7 – Safety communication layer overview.....	39
Figure 8 – SafetyProvider interfaces.....	40
Figure 9 – Example combinations of SIL capabilities.....	46
Figure 10 – SafetyConsumer interfaces	47
Figure 11 – RequestSPDU	56
Figure 12 – ResponseSPDU.....	57
Figure 13 – Sequence diagram for requests and responses (Client/Server)	59
Figure 14 – Sequence diagram for requests and responses (PubSub).....	60
Figure 15 – Duration of demand example for missed demand value in case of currently available SafetyData not being provided until second change of MNR	61
Figure 16 – Duration of demand example for received demand value in case of currently available SafetyData being provided	62
Figure 17 – Simplified representation of the state diagram for the SafetyProvider.....	62
Figure 18 – Principle state diagram for SafetyConsumer.....	65
Figure 19 – Sequence diagram for OA.....	75
Figure 20 – Overview of task for SafetyProvider	76
Figure 21 – Calculation of the SPDU_ID	77
Figure 22 – Example for the calculation of SPDU_ID_1, SPDU_ID_2 and SPDU_ID_3.....	78
Figure 23 – Calculation of the CRC (on little-endian machines, CRC32_Backward)	81
Figure 24 – Calculation of the CRC (on big-endian machines, CRC32_Forward)	82
Figure 25 – Overview of delay times and watchdogs	83
Figure 26 – Conditional residual error probability of the CRC check	87
Figure 27 – Counter example: data lengths not supported by OPC Safety	88
Figure 28 – Facets and ConformanceUnits	94
Figure B.1 – Unidirectional communication	99
Figure B.2 – Bidirectional communication	99
Figure B.3 – Safety multicast	99
Figure B.4 – OA in unidirectional safety communication.....	100
Figure B.5 – Two-sided OA in bidirectional safety communication	101
Figure B.6 – One sided OA in bidirectional safety communication	101
Figure B.7 – One sided OA on each side is possible	102
Table 1 – Conventions used in state machines	16
Table 2 – Deployed safety measures to detect communication errors.....	18
Table 3 – SafetyACSet definition.....	22
Table 4 – SafetyObjectsType definition	26
Table 5 – SafetyProviderType definition	26
Table 6 – SafetyConsumerType definition	27
Table 7 – ReadSafetyData Method arguments	28
Table 8 – ReadSafetyData Method AddressSpace definition	29

Table 9 – ReadSafetyDiagnostics Method arguments	30
Table 10 – ReadSafetyDiagnostics Method AddressSpace definition	30
Table 11 – SafetyPDUsType definition	31
Table 12 – SafetyProviderParametersType definition	33
Table 13 – SafetyConsumerParametersType definition	34
Table 14 – InFlagsType values	35
Table 15 – InFlagsType definition	35
Table 16 – OutFlagsType values	35
Table 17 – OutFlagsType definition	36
Table 18 – RequestSPDUDataType structure	36
Table 19 – RequestSPDUDataType definition	36
Table 20 – ResponseSPDUDataType structure	37
Table 21 – ResponseSPDUDataType definition	37
Table 22 – NonSafetyDataPlaceholderDataType structure	37
Table 23 – SAPI of the SafetyProvider	41
Table 24 – SPI of the SafetyProvider	42
Table 25 – SAPI of the SafetyConsumer	47
Table 26 – SPI of the SafetyConsumer	50
Table 27 – Example "application variables with qualifier"	53
Table 28 – Safety layer diagnostic messages	54
Table 29 – Symbols used for state machines	62
Table 30 – SafetyProvider instance internal items	63
Table 31 – States of SafetyProvider instance	64
Table 32 – SafetyProvider transitions	64
Table 33 – SafetyConsumer internal items	66
Table 34 – SafetyConsumer states	70
Table 35 – SafetyConsumer transitions	71
Table 36 – Presentation of the SPDU_ID	77
Table 37 – Coding for the SafetyProviderLevel_ID	78
Table 38 – Examples for cryptographically strong random number generators	85
Table 39 – The total residual error rate for the safety communication channel	89
Table 40 – Information to be included in the safety manual	89
Table 41 – Index of requirements (informative)	92
Table 42 – NamespaceMetadata Object for this document	95
Table 43 – Namespaces used in a safety Server	95
Table B.1 – The CRC32 lookup table for 32-bit CRC signature calculations	98

INTERNATIONAL ELECTROTECHNICAL COMMISSION

OPC UNIFIED ARCHITECTURE –

Part 15: Safety

FOREWORD

- 1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
- 2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
- 3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
- 4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
- 5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.
- 6) All users should ensure that they have the latest edition of this publication.
- 7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.
- 8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.
- 9) IEC draws attention to the possibility that the implementation of this document may involve the use of (a) patent(s). IEC takes no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, IEC had received notice of (a) patent(s), which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at <https://patents.iec.ch>. IEC shall not be held responsible for identifying any or all such patent rights.

IEC 62541-15 has been prepared by subcommittee 65C: Industrial networks, of IEC technical committee 65: Industrial-process measurement, control and automation. It is an International Standard.

The text of this International Standard is based on the following documents:

Draft	Report on voting
65C/1334/FDIS	65C/1339/RVD

Full information on the voting for its approval can be found in the report on voting indicated in the above table.

The language used for the development of this International Standard is English.

This document was drafted in accordance with ISO/IEC Directives, Part 2, and developed in accordance with ISO/IEC Directives, Part 1 and ISO/IEC Directives, IEC Supplement, available at www.iec.ch/members_experts/refdocs. The main document types developed by IEC are described in greater detail at www.iec.ch/publications.

Throughout this document and the referenced other parts of the IEC 62541 series, certain document conventions are used:

Italics are used to denote a defined term or definition that appears in Clause 3 in one of the parts of the series.

Italics are also used to denote the name of a service input or output parameter or the name of a structure or element of a structure that are usually defined in tables.

The *italicized terms* and *names* are also, with a few exceptions, written in camel-case (the practice of writing compound words or phrases in which the elements are joined without spaces, with each element's initial letter capitalized within the compound). For example, the defined term is *AddressSpace* instead of Address Space. This makes it easier to understand that there is a single definition for *AddressSpace*, not separate definitions for Address and Space.

A list of all parts of the IEC 62541 series, published under the general title *OPC Unified Architecture*, can be found on the IEC website.

The committee has decided that the contents of this document will remain unchanged until the stability date indicated on the IEC website under webstore.iec.ch in the data related to the specific document. At this date, the document will be

- reconfirmed,
- withdrawn, or
- revised.

INTRODUCTION

OPC UA safety extends OPC UA to fulfill the requirements of functional safety as defined in the IEC 61508 series and IEC 61784-3 series of standards.

Figure 1 shows the relationship between this document and the relevant safety and OPC UA standards in an industrial environment. An arrow from Document A to Document B means "Document A is referenced in Document B". This reference can be either normative or informative. Not all of these standards are applicable or required for a given product.

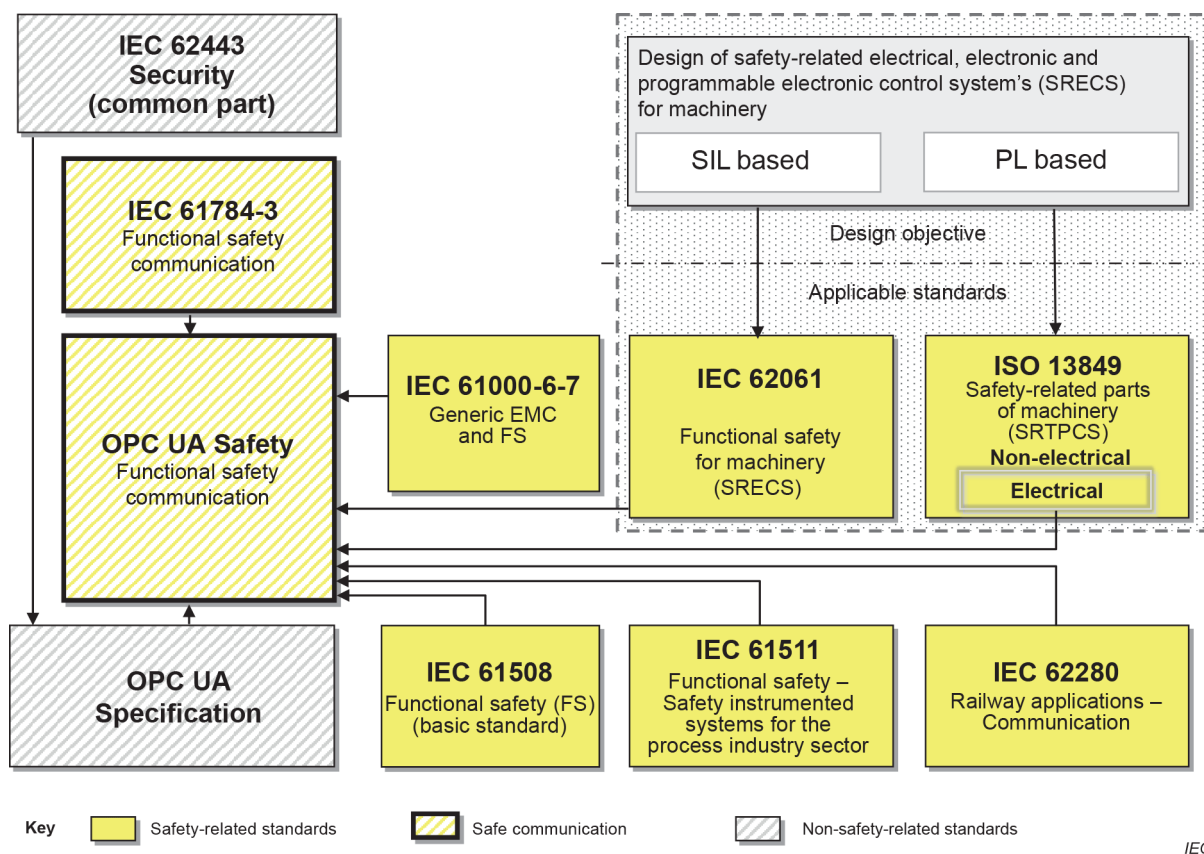


Figure 1 – Relationships of OPC UA safety with other standards

Implementing this document allows for detecting all types of communication errors encountered in the lower network layers. In case an error is detected, this information is shared with the safety applications in the user layer which can then act in an appropriate way, e.g. by switching to a safe state.

The document describes the behaviour of the individual endpoints for safe communication, as well as the OPC UA *Information Model* which is used to access these endpoints.

This document is application-independent and does not pose requirements on the structure and length of the application data. Application-specific requirements are expected to be described in appropriate companion specifications.

This document can be used for applications requiring functional safety up to the *safety integrity level (SIL) 4*.

OPC UNIFIED ARCHITECTURE –

Part 15: Safety

1 Scope

This document describes a *safety communication layer* (services and a protocol) for the exchange of *SafetyData* using IEC 62541 mechanisms. It identifies the principles for functional safety communications defined in IEC 61784-3 that are relevant for this *safety communication layer*. This *safety communication layer* is intended for implementation in *safety* devices only.

NOTE 1 This document targets controller-to-controller communication. However, easy expandability to other use-cases (e.g. OPC UA field level communication) has already been considered in the design of this document.

NOTE 2 This document does not cover electrical safety and intrinsic safety aspects. Electrical safety relates to hazards such as electrical shock. Intrinsic safety relates to hazards associated with potentially explosive atmospheres.

This document defines mechanisms for the transmission of safety-relevant messages among participants within a network using OPC UA technology in accordance with the requirements of the IEC 61508 series and IEC 61784-3 for functional safety. These mechanisms can be used in various industrial applications such as process control, manufacturing, automation, and machinery.

This document provides guidelines for both developers and assessors of compliant devices and systems.

NOTE 3 The resulting *SIL* claim of a system depends on the implementation of this document within the system – implementation of this document in a standard device is not sufficient to qualify it as a safety device.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 61508 (all parts), *Functional safety of electrical/electronic/programmable electronic safety-related systems*

IEC 61784-3:2021, *Industrial communication networks – Profiles – Part 3: Functional safety fieldbuses – General rules and profile definitions*

IEC 62443 (all parts), *Industrial communication networks – Network and system security*

IEC 62541-1:2020, *OPC Unified Architecture – Part 1: Overview and Concepts*

IEC 62541-3:2020, *OPC Unified Architecture – Part 3: Address Space Model*

IEC 62541-4:2020, *OPC Unified Architecture – Part 4: Services*

IEC 62541-5:2020, *OPC Unified Architecture – Part 5: Information Model*

IEC 62541-6:2020, *OPC Unified Architecture – Part 6: Mappings*

IEC 62541-14, *OPC Unified Architecture – Part 14: PubSub*

ISO/IEC 9834-8:2014, *Information technology – Procedures for the operation of object identifier registration authorities – Part 8: Generation of universally unique identifiers (UUIDs) and their use in object identifiers*

3 Terms, definitions, symbols, abbreviated terms and conventions

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC 62541-1:2020, IEC 62541-3:2020, IEC 62541-4:2020, IEC 62541-6:2020 and the following apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

- IEC Electropedia: available at <https://www.electropedia.org/>
- ISO Online browsing platform: available at <https://www.iso.org/obp>

NOTE This document uses concepts of IEC 62541 information modeling to describe the concepts in this document.

3.1.1 Common terms and definitions

3.1.1.1

Cyclic Redundancy Check

CRC

<value> redundant data derived from, and stored or transmitted together with, a block of data in order to detect data corruption

<method> procedure used to calculate the redundant data

Note 1 to entry: Terms "CRC code" and "CRC signature", and labels such as CRC1, CRC2, may also be used in this document to refer to the redundant data.

[SOURCE: IEC 61784-3:2021, 3.10]

3.1.1.2

error

discrepancy between a computed, observed or measured value or condition and the true, specified or theoretically correct value or condition

Note 1 to entry: Errors may be due to design mistakes within hardware/software and/or corrupted information due to electromagnetic interference and/or other effects.

Note 2 to entry: Errors do not necessarily result in a failure or a fault.

[SOURCE: IEC 60050-192:2024, 192-03-02, modified – notes added]

3.1.1.3

failure

termination of the ability of a functional unit to perform a required function or operation of a functional unit in any way other than as required

Note 1 to entry: Failure can be due to an error (for example, problem with hardware/software design or message disruption).

[SOURCE: IEC 61508-4:2010, 3.6.4, modified – notes and figures deleted, new note to entry added]

3.1.1.4**fault**

abnormal condition that may cause a reduction in, or loss of, the capability of a functional unit to perform a required function

Note 1 to entry: IEC 191-05-01 defines "fault" as a state characterized by the inability to perform a required function, excluding the inability during preventive maintenance or other planned actions, or due to lack of external resources.

[SOURCE: IEC 61508-4:2010, 3.6.1, modified – figure reference deleted]

message

<information theory and communication theory>ordered sequence of characters (usually octets) intended to convey information

[SOURCE: ISO/IEC 2382:2015, 2123031, modified – insertion of "(usually octets)", deletion of notes and source]

3.1.1.5**performance level**

PL

discrete level used to specify the ability of safety-related parts of control systems to perform a safety function under foreseeable conditions

[SOURCE: ISO 13849-1:2023, 3.1.5]

3.1.1.6**residual error probability**

probability of an error undetected by the *SCL* safety measures

[SOURCE: IEC 61784-3:2021, 3.1.35]

3.1.1.7**residual error rate**

statistical rate at which the *SCL* safety measures fail to detect errors

[SOURCE: IEC 61784-3:2021, 3.1.36]

3.1.1.8**safety communication layer**

SCL

communication layer above the IEC 62541 communication stack that includes all necessary additional measures to ensure safe transmission of data in accordance with the requirements of IEC 61508

Note 1 to entry: The *SCL* provides several services, the most important ones being the *SafetyProvider* and the *SafetyConsumer*.

[SOURCE: IEC 61784-3:2021, 3.1.39, modified – "FAL" replaced by "IEC 62541 communication stack", note to entry added]

3.1.1.9**safety function response time**

worst case elapsed time following an actuation of a safety sensor connected to a fieldbus, until the corresponding safe state of its safety actuator(s) is achieved in the presence of errors or failures in the safety function

Note 1 to entry: This concept is introduced in IEC 61784-3:2021, 5.2.4 and is addressed by the functional safety communication profiles defined in the IEC 61784-3 series of documents.

[SOURCE: IEC 61784-3:2021, 3.1.44]

3.1.1.10**safety integrity level**

SIL

discrete level (one out of a possible four), corresponding to a range of safety integrity values, where safety integrity level 4 has the highest level of safety integrity and safety integrity level 1 has the lowest

Note 1 to entry: The target failure measures (see IEC 61508-4:2010, 3.5.17) for the four *safety integrity levels* are specified in Table 2 and Table 3 of IEC 61508-1:2010.

Note 2 to entry: *Safety integrity levels* are used for specifying the safety integrity requirements of the safety functions to be allocated to the E/E/PE safety-related systems.

Note 3 to entry: A *safety integrity level (SIL)* is not a property of a system, subsystem, element or component. The correct interpretation of the phrase "*SIL n* safety-related system" (where *n* is 1, 2, 3 or 4) is that the system is potentially capable of supporting safety functions with a *safety integrity level* up to *n*.

[SOURCE: IEC 61508-4:2010, 3.5.8]

3.1.1.11**safety measure**

measure to control possible communication errors that is designed and implemented in compliance with the requirements of IEC 61508

Note 1 to entry: In practice, several safety measures are combined to achieve the required *safety integrity level*.

Note 2 to entry: Communication errors and related safety measures are detailed in IEC 61784-3:2021, 5.3 and 5.4.

[SOURCE: IEC 61784-3:2021, 3.1.46]

3.1.1.12**safety PDU**

SPDU

PDU transferred through the *safety communication channel*

Note 1 to entry: The SPDU may include more than one copy of the *SafetyData* using differing coding structures and hash functions together with explicit parts of additional protections such as a key, a sequence count, or a time stamp mechanism.

Note 2 to entry: Redundant *SCLs* may provide two different versions of the SPDU for insertion into separate fields of the IEC 62541 frame.

[SOURCE: IEC 61784-3:2021, 3.1.47]

3.1.2 Additional terms and definitions**3.1.2.1****fail-safe**

ability of a system that, by adequate technical or organizational measures, prevents from hazards either deterministically or by reducing the risk to a tolerable measure

Note 1 to entry: Equivalent to functional safety.

3.1.2.2**fail-safe substitute values**

FSV

values which are issued or delivered instead of process values when the safety function is set to a fail-safe state

Note 1 to entry: In this document, the fail-safe substitute values (FSV) are always set to binary "0".

3.1.2.3**flag**

one-bit value used to indicate a certain status or control information

3.1.2.4**globally unique identifier**

GUID

128-bit number used to identify information in computer systems

Note 1 to entry: The term universally unique identifier (UUID) is also used.

Note 2 to entry: In this document, UUID version 4 is used.

3.1.2.5**MonitoringNumber**

MNR

means used to ensure the correct order among transmitted safety PDUs and to monitor the communication delay

Note 1 to entry: Instance of sequence number as described in IEC 61784-3.

Note 2 to entry: The *MNR* starts at a random value and is incremented with each request. It rolls over to a minimum threshold value that is not zero.

Note 3 to entry: The transmitted *MNR* is protected by the transmitted *CRC* signature of the ResponseSPDU.

3.1.2.6**non-safety-**

predicate meaning that the respective object is a "standard" object and has not been designed and implemented to fulfil any requirements with respect to functional safety

3.1.2.7**OPC UA mapper**

non-safety-related part of the implementation of this document which maps the SPDU to the actual IEC 62541 services

Note 1 to entry: Depending on which services of IEC 62541 are being used (e.g. *Client/Server* or *PubSub*), different mappers can be specified.

3.1.2.8**process values**

PV

input and output data (in a safety PDU) that are required to control an automated process

3.1.2.9**qualifier**

attribute (bit or Boolean), indicating whether the corresponding value is valid or not (e.g. being a fail-safe substitute value)

3.1.2.10**SafetyAutomationComponent**

SafetyAC

communication partner in a unidirectional safety link

Note 1 to entry: A *SafetyAutomationComponent* can be a *SafetyProvider* (data source), a *SafetyConsumer* (data sink), or both.

3.1.2.11**SafetyConsumer**

entity (usually software) that implements the data sink of a unidirectional safety link

3.1.2.12

SafetyData

application data transmitted across a safety network using a safety protocol

Note 1 to entry: The *safety communication layer* does not ensure the safety of the data itself, but only that the data is transmitted safely.

3.1.2.13

SafetyProvider

entity (usually software) that implements the data source of a unidirectional safety link

3.1.2.14

SafetyBaseID

randomly generated authenticity ID which is used to safely authenticate *SafetyProviders* having the same *SafetyProviderID*

Note 1 to entry: Together with the *SafetyProviderID*, it is an instance of connection authentication as described in IEC 61784-3.

3.1.2.15

SafetyProviderID

user-assigned, locally unique identifier which is used to safely authenticate *SafetyProviders* within a certain area

Note 1 to entry: Together with the *SafetyBaseID*, it is an instance of connection authentication as described in IEC 61784-3.

Note 2 to entry: All *SafetyProviders* within an area such defined may share an identical *SafetyBaseID*.

3.1.2.16

standard transmission system

part of the transmission system (implemented in hardware and software) that is not implemented according to any safety standards

Note 1 to entry: This document is using the services of the *standard transmission system* to transmit prebuilt safety packets.

3.2 Symbols and abbreviated terms

For the purposes of this document, the following symbols and abbreviated terms apply.

3.2.1 Abbreviated terms from IEC 61784-3

CRC	cyclic redundancy check	
PDU	protocol data unit	[ISO/IEC 7498-1]
PL	performance level	[ISO 13849-1]
PLC	programmable logic controller	
SCL	safety communication layer	
SIL	safety integrity level	[IEC 61508-4]
SPDU	safety PDU, safety protocol data unit	

3.2.2 Additional symbols and abbreviated terms

3.2.2.1 Abbreviated terms

FSV	fail-safe substitute values
HMI	human-machine interface
ID	identifier
LSB	least significant bit
MNR	MonitoringNumber
MSB	most significant bit
OA	operator acknowledgment
OPC UA PI	OPC UA platform interface
PI	platform interface
PV	process values
SAPI	safety application program interface
SFRT	safety function response time
SPI	safety parameter interface
STrailer	safety trailer
TRA	threat and risk analysis

3.2.2.2 Symbols

p	bit error probability
$P_{re,cond}$	conditional residual error probability

3.3 Conventions

3.3.1 General conventions

Terms or names where two capital letters of abbreviations are in sequence or for separation to a suffix are written with underscores in between.

The abbreviation "F" is an indication for *safety-related* items, technologies, systems, and units (*fail-safe*, functional safe).

The default data that are used in case of unit failures or errors, are called *fail-safe substitute values* (FSV) and are set to binary "0".

Reserved bits ("res") are set to "0" and ignored by the receiver to avoid problems with future versions of this document.

The notation 0x... represents a hexadecimal value.

3.3.2 Conventions for requirements numbering

Requirements in this document are designated as [RQx.yz], where x denotes the chapter number, y is a counter and z is an optional character to link closely related requirements. The following are examples of valid requirements designations: [RQ8.15] (requirement 15 in chapter 8); [RQ47.11a], [RQ47.11b] (requirements 11a and 11b in chapter 47, which are closely related).

The initial numbering of requirements was chosen such that counters within each chapter are in ascending order. However, the addition of further requirements leads to deviations from this rule since existing requirements shall keep their initial designation.

For an informative index of all the requirements in this document, see 10.3.

3.3.3 Conventions in state machines

See Table 1 for the conventions used in state machines.

Table 1 – Conventions used in state machines

Convention	Meaning
:=	Assignment: value of an item on the left is replaced by value of the item on the right.
<	Less than: a logical condition yielding TRUE if and only if an item on the left is less than the item on the right.
<=	Less or equal than: a logical condition yielding TRUE if and only if an item on the left is less or equal than the item on the right.
>	Greater than: a logical condition yielding TRUE if and only if the item on the left is greater than the item on the right.
>=	Greater or equal than: a logical condition yielding TRUE if and only if the item on the left is greater or equal than the item on the right.
==	Equality: a logical condition yielding TRUE if and only if the item on the left is equal to an item on the right.
<>	Inequality: a logical condition yielding TRUE if and only if the item on the left is not equal to an item on the right.
&&	Logical "AND" (Operation on binary values or results).
	Logical "OR" (Operation on binary values or results).
⊕	Logical "XOR" (Operation on binary values or digital values).
[..]	UML Guard condition, if and only if the guard is TRUE the respective transition is enabled.

4 Overview of OPC UA Safety

4.1 General

This document specifies a *safety communication layer (SCL)* allowing safety-related devices to use the services of IEC 62541 for the safe exchange of safety-related data. A safety device that implements OPC UA Safety correctly will be able to exchange safety-related data and hereby fulfill the requirements of the IEC 61508 series and IEC 61784-3. This document uses a *MonitoringNumber*, a timeout, a set of IDs and a *cyclic redundancy check (CRC)* for the detection of all possible communication errors which can happen in the underlying IEC 62541 *standard transmission system*. These *safety measures* have been quantitatively evaluated and offer a probability of dangerous failure per hour (PFH) and a probability of dangerous failure on demand (PFD) sufficing to build *safety-related* applications with a *safety integrity level* of up to SIL4.

OPC UA Safety itself is an application-independent, general solution. The length and structure of the data sent is defined by the safety application. However, application-dependent companion specifications (addressing for example electro-sensitive protective equipment, electric drives with safety functions, forming presses, robot safety, and automated guided vehicles) are expected to be defined by application-experts in appropriate OPC UA companion specifications.

4.2 Implementation aspects

[RQ4.1] All technical measures for error detection described in this document shall be implemented within the *SCL* in devices designed in accordance with the IEC 61508 series and shall meet the target *SIL*.

4.3 Features

- Runs on top of:
 - IEC 62541 *Client/Server* with the *Method Service Set*.
 - IEC 62541 *PubSub*.
- From an architectural point of view: easy extensibility for other ways of communication.
- Goal: no modification of existing OPC UA framework.
- The state machines of this document are independent from the *OPC UA mapper*, allowing for a simplified exchange of the mapper.
- Ready for wireless transmission channels.
- Modest requirements on safety network nodes:
 - No clock synchronization is necessary (no requirements regarding the accuracy between clocks at different nodes).
 - Within the *SafetyConsumer*, a safety-related, local timer is required for implementing the *SafetyConsumerTimeout*. The accuracy of this timer depends on the timing requirements of the safety application.
- End-to-End Safety: functional *SafetyData* is transported between two safety endpoint devices across a standard network that is not functionally safety compliant. This includes the lower transport layers such as the IEC 62541 stack, underlying physical media, and *non-safety* network elements (e.g. routers and switches).
- "Dynamic" systems:
 - Safety communication partners can change during runtime,
 - either an increase or decrease, or both, in the number of safety communication partners can occur.
- Well-defined text-strings are used for diagnostic purposes.
- Safety communication and standard communication are independent. However, standard devices and safety devices can use the same *standard transmission system* at the same time.
- Functional safety can be achieved without using structurally redundant *standard transmission systems* i.e. a single channel approach can be used. Redundancy can be used optionally for increased availability.
- For diagnostic purposes, the last *SPDU* sent and received is accessible in the *Information Model* of the *SafetyProvider*.
- Length of user data: 1 octet to 1 500 octets, structures of basic *DataTypes*, see 6.2.5.

4.4 Security policy

In the final application, an appropriate security environment is necessary to protect both the operational environment and the safety-related systems.

This document does not cover security aspects, nor does it provide any requirements for security.

A threat and risk analysis (TRA) according to the IEC 62443 series shall be carried out on a final application system level.

During compliance tests for this document, security aspects are not part of the scope, as it is assumed that the underlying base mechanisms (i.e. *Methods*) already provide adequate security.

5 General

5.1 External documents providing specifications for the profile

No other external documents are providing specifications in addition to the Normative references given in Clause 2.

5.2 Safety functional requirements

The following requirements apply for the development of this document:

- a) Safety communication suitable for *safety integrity level* up to *SIL4* (see the IEC 61508 series) and *PL e* (see ISO 13849-1).
- b) Combination of *SIL* 1 to 4 devices according to this document as well as *non-safety* devices on one communication network.
- c) Implementation of the safety transmission protocol is restricted to the safety layer.
- d) The safety-relevant time-out monitoring is implemented in the safety layer.
- e) Safety communication meet the requirements of IEC 61784-3.
- f) [RQ5.1] This document is intended for implementation in safety devices exclusively. Exceptions (e.g. for debugging, simulation, testing, and commissioning) shall be discussed with a notified body.

5.3 Safety measures

[RQ5.2] For an implementation of this document, the following *safety measures* shall be implemented: *MonitoringNumber*; timeout with receipt in the *SafetyConsumer*; set of IDs for the *SafetyProvider*; Data Integrity check.

Together, these *safety measures* address all possible transmission errors as listed in IEC 61784-3:2021, 5.5, see Table 2.

[RQ5.3] The *safety measures* shall be processed and monitored within the *SCL*.

Table 2 – Deployed safety measures to detect communication errors

Communication error	Safety measures			
	MonitoringNumber ^a	Timeout with receipt ^b	Set of IDs for SafetyProvider ^c	Data integrity check ^d
Corruption	–	–	–	X
Unintended repetition	X	X	–	–
Incorrect sequence	X	–	–	–
Loss	X	X	–	–
Unacceptable delay	–	X	–	–
Insertion	X	–	–	–
Masquerade	X	–	X	X
Addressing	–	–	X	–
^a Instance of "sequence number" of IEC 61784-3. ^b Instance of "time expectation" (timeout) and "feedback message" (receipt) of IEC 61784-3. ^c Instance of "connection authentication" of IEC 61784-3. ^d Instance of "data integrity assurance" of IEC 61784-3, based on <i>CRC</i> signature.				

The *SafetyConsumer* is specified in such a way that for any communication error according to Table 2, a defined fault reaction will occur.

In all cases, the faulty *SPDU* will be discarded, and not forwarded to the safety application.

Moreover, if the error rate is too high, the *SafetyConsumer* is defined in such a way that it will cease to deliver actual *process values* to the safety application but will deliver *fail-safe substitute values* instead. In addition, an indication at the *Safety Application Program Interface* is set which can be queried by the safety application.

In case the error rate is still considered acceptable, the state machine repeats the request, see 9.4.

5.4 Safety communication layer structure

This document is based on:

- the *standard transmission system* according to IEC 62541
- an additional safety transmission protocol on top of this *standard transmission system*

Safety applications and standard applications share the same standard IEC 62541 communication systems at the same time. The safe transmission function incorporates *safety measures* to detect faults or hazards that originate in the *standard transmission system* which have a potential to compromise the safety subsystems. This includes faults such as:

- Random errors, for example due to electromagnetic interference on the transmission channel;
- Failures or faults of the standard hardware;
- Systematic malfunctions of components within the standard hardware and software.

This principle delimits the assessment effort to the "safe transmission functions". The *standard transmission system* does not require any additional functional safety assessment.

The basic communication layers of this document are shown in Figure 2.

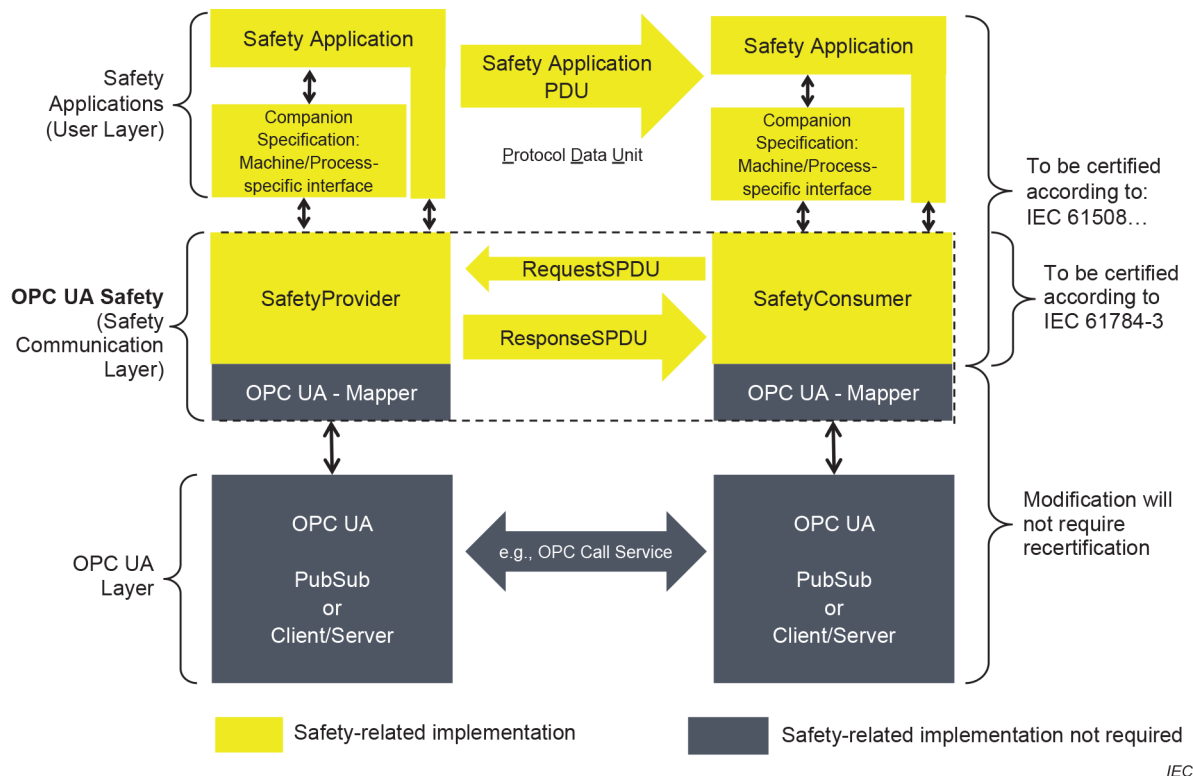


Figure 2 – Safety layer architecture

Summary of the architecture:

Part: User layer

The safety applications in the User Layer are either directly connected to the *SafetyProvider* or *SafetyConsumer*, or they are connected via a machine-specific or process-specific interface, which is described in companion specifications (e.g. sectoral).

The safety applications are expected to be designed and implemented according to the IEC 61508 series.

The Safety applications in the User Layer are not within the scope of this document.

Part: OPC UA Safety (Safety Communication Layer)

This layer is within the scope of this document. It defines the two services *SafetyProvider* and *SafetyConsumer* as basic building blocks. Together, they form the *safety communication layer* (SCL), implemented in a safety-related way according to the IEC 61508 series.

SafetyData is transmitted using point-to-point communication (unidirectional). Each unidirectional data flow internally communicates in both directions, using a request and response pattern. This allows for checking the timeliness of messages using a single clock in the *SafetyConsumer*, thus eliminating the necessity for synchronized clocks.

When *SafetyConsumers* connect to *SafetyProviders*, they have prior expectations regarding the pair of *SafetyProviderID* and *SafetyBaseID* (e.g. by configuration). If this expectation is not fulfilled by the *SafetyProvider*, *fail-safe substitute values* are delivered to the safety application instead of the received *process values*. In contrast, it is not necessary for a *SafetyProvider* to know the *SafetyConsumerID* of the *SafetyConsumer* and will provide its *process values* to any *SafetyConsumer* requesting it.

SafetyProviders can not detect communication errors. All required error detection is performed by the *SafetyConsumer*.

If it is necessary for a pair of safety applications to exchange *SafetyData* in both directions, two pairs of *SafetyProviders* and *SafetyConsumers* shall be established, one pair for each direction.

Refer to B.2 in Annex B for use cases how *SafetyProviders* and *SafetyConsumers* can be connected.

The OPC UA mapper implements the parts of the *safety communication layer* which are specific for the IEC 62541 communication *Service* in use, i.e. *PubSub* or *Client/Server*. Therefore, the remaining parts of the *safety communication layer* can be implemented independent of the IEC 62541 *Service* being used.

Part: OPC UA Layer

Client/Server:

- The *SafetyProvider* is implemented using an IEC 62541 *Server* providing a *Method*.
- The *SafetyConsumer* is implemented using an IEC 62541 *Client* calling the *Method* provided by the *SafetyProvider*.

PubSub:

- The *SafetyProvider* publishes the *ResponseSPDU* and subscribes to the *RequestSPDU*.
- The *SafetyConsumer* publishes the *RequestSPDU* and subscribes to the *ResponseSPDU*.

5.5 Requirements for CRC calculation

[RQ5.4] Any *CRC* signature calculation shall start with a preset value of "1".

[RQ5.5] Any *CRC* signature calculation resulting in a "0" value, shall use the value "1" instead.

[RQ5.6] *SPDUs* with all values (incl. *CRC* signature) being zero shall be ignored by the receiver (*SafetyConsumer* and *SafetyProvider*).

6 Safety communication layer services

6.1 General

Clause 6 describes the integration of this document into the IEC 62541 *Information Models* in 6.2 and the resulting *Service* interfaces in 6.3. Diagnostic services are described in 6.4.

6.2 Information models

6.2.1 General

Subclause 6.2 describes the identifiers, types and structure of the *Objects* and *Methods* that are used to implement the *OPC UA mappers* defined in this document. This implementation serves three purposes:

- support of the safe exchange of *SPDUs* at runtime
- online browsing, to identify *SafetyConsumers* and *SafetyProviders*, and to check their parameters for diagnostic purposes
- offline engineering: the *Information Model* of one controller can be exported in a standardized file on its engineering system, be imported in another engineering system, and finally deployed on another controller. This allows for a vendor-independent exchange of the communication interfaces of safety applications, e.g. for establishing connections between devices.

NOTE Neither online browsing nor offline engineering currently supports any features to detect errors. Hence, no guarantees with respect to functional safety are made. This means that online browsing can only be used for diagnostic purposes, and not for exchanging safety-relevant data. It is assumed that errors can occur during the transfer of the *Information Model* from one engineering system to another in the context of offline engineering. Therefore, the programmer of the safety application is responsible for the verification and validation of the safety application.

Consequently, all type values described in 6.2 are defined as read-only, i.e. they cannot be written by general IEC 62541 write commands.

6.2.2 Object and ObjectType Definitions

6.2.2.1 SafetyACSet Object

[RQ6.1] Each *Server* shall have a singleton *Folder* called *SafetyACSet* with a fixed *NodeID* in the *Namespace* of this document. Because all *SafetyProviders* and *SafetyConsumers* on this *Server* contain a hierarchical *Reference* from this *Object* to themselves, it can be used to directly access all *SafetyProviders* and *SafetyConsumers*. *SafetyACSet* is intended for safety-related purposes only. It should not reference *non-safety*-related items.

See Table 3 for the definition of the *SafetyACSet*.

Table 3 – SafetyACSet definition

Attribute	Value		
BrowseName	SafetyACSet		
References	NodeClass	BrowseName	Comment
OrganizedBy	by the Objects Folder defined in IEC 62541-5.		
HasTypeDefinition	ObjectType	FolderType	Entry point for all <i>SafetyProviders</i> and <i>SafetyConsumers</i>
Conformance units			
SafetyACSet			

[RQ6.2] In addition, a *Server* shall comprise one IEC 62541 *Object* derived from *DataType SafetyProviderType* for each *SafetyProvider* it implements, and one IEC 62541 *Object* derived from *DataType SafetyConsumerType* for each *SafetyConsumer* it implements. The corresponding *Information Models* shown in Figure 3 and Figure 4 shall be used.

A description of the graphical notation for the different types of Nodes and references (shown in Figure 3, Figure 4, and Figure 6) can be found in IEC 62541-3.

Figure 3 describes the *SafetyProvider* and the *SafetyConsumer*.

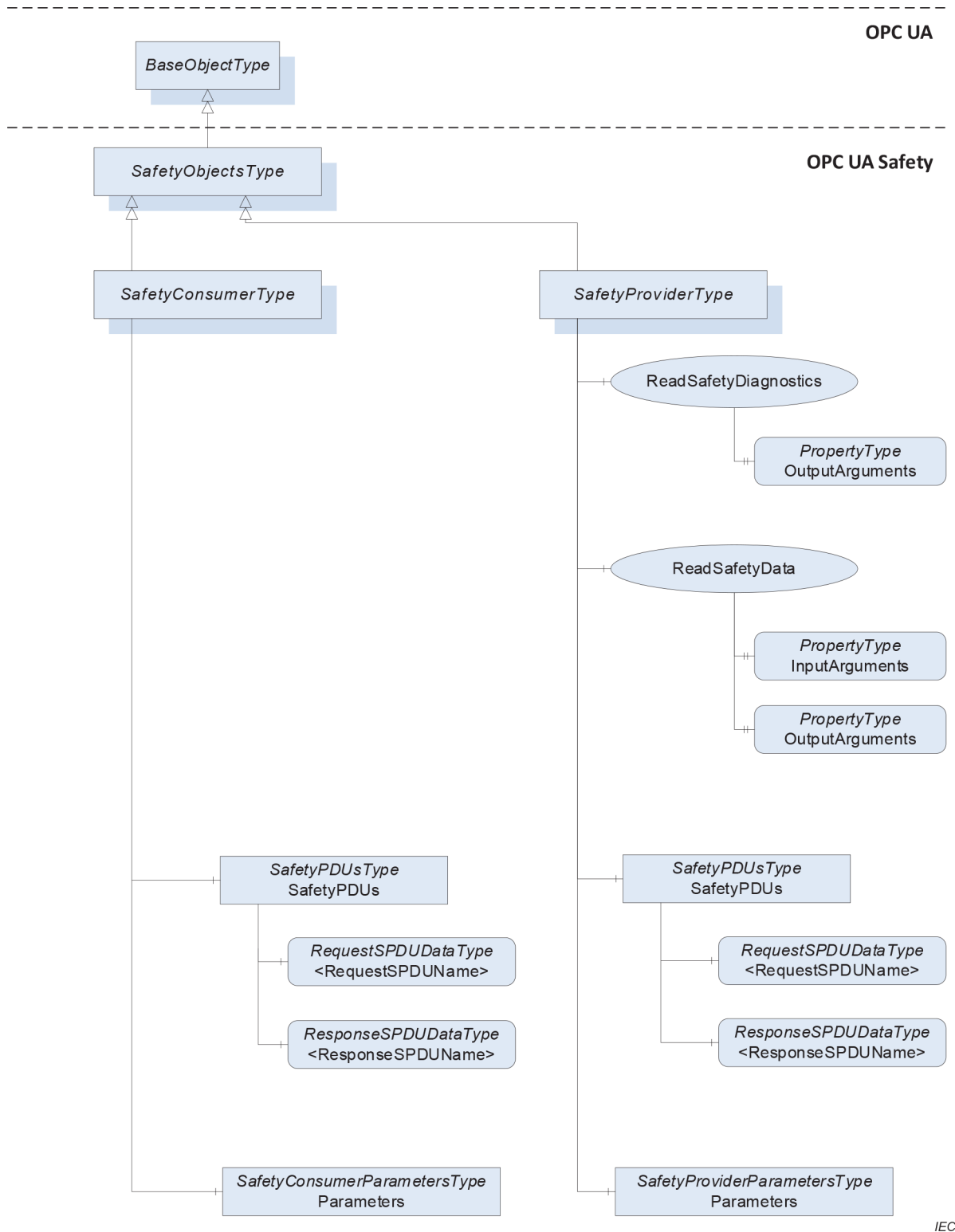
NOTE 1 This document assumes (atomic) consistent data exchange between OPC mappers of the two endpoints.

[RQ6.3a] For implementations supporting IEC 62541 *Client/Server*, the *Call Service* of the *Method Service Set* (see IEC 62541-4) shall be used. The *Method ReadSafetyData* has a set of input arguments that make up the *RequestSPDU* and a set of output arguments that make up the *ResponseSPDU*. The *SafetyConsumer* uses the IEC 62541 Client with the IEC 62541 *Service Call*.

[RQ6.3b] For implementations supporting IEC 62541 *PubSub*, the IEC 62541 *Object SafetyPDUs* with its *Properties RequestSPDU* and *ResponseSPDU* shall be used. *RequestSPDU* is published by the *SafetyConsumer* and subscribed by the *SafetyProvider*. *ResponseSPDU* is published by the *SafetyProvider* and subscribed by the *SafetyConsumer*.

NOTE 2 The terms "request" and "response" refer to the behaviour on the layer of this document. Within the *PubSub* context, both requests and responses are realized by repeatedly publishing and subscribing *Messages*, see Figure 14.

[RQ6.4] For diagnostic purposes, the *SPDUs* received and sent shall be accessible by calling the *Method ReadSafetyDiagnostics*.

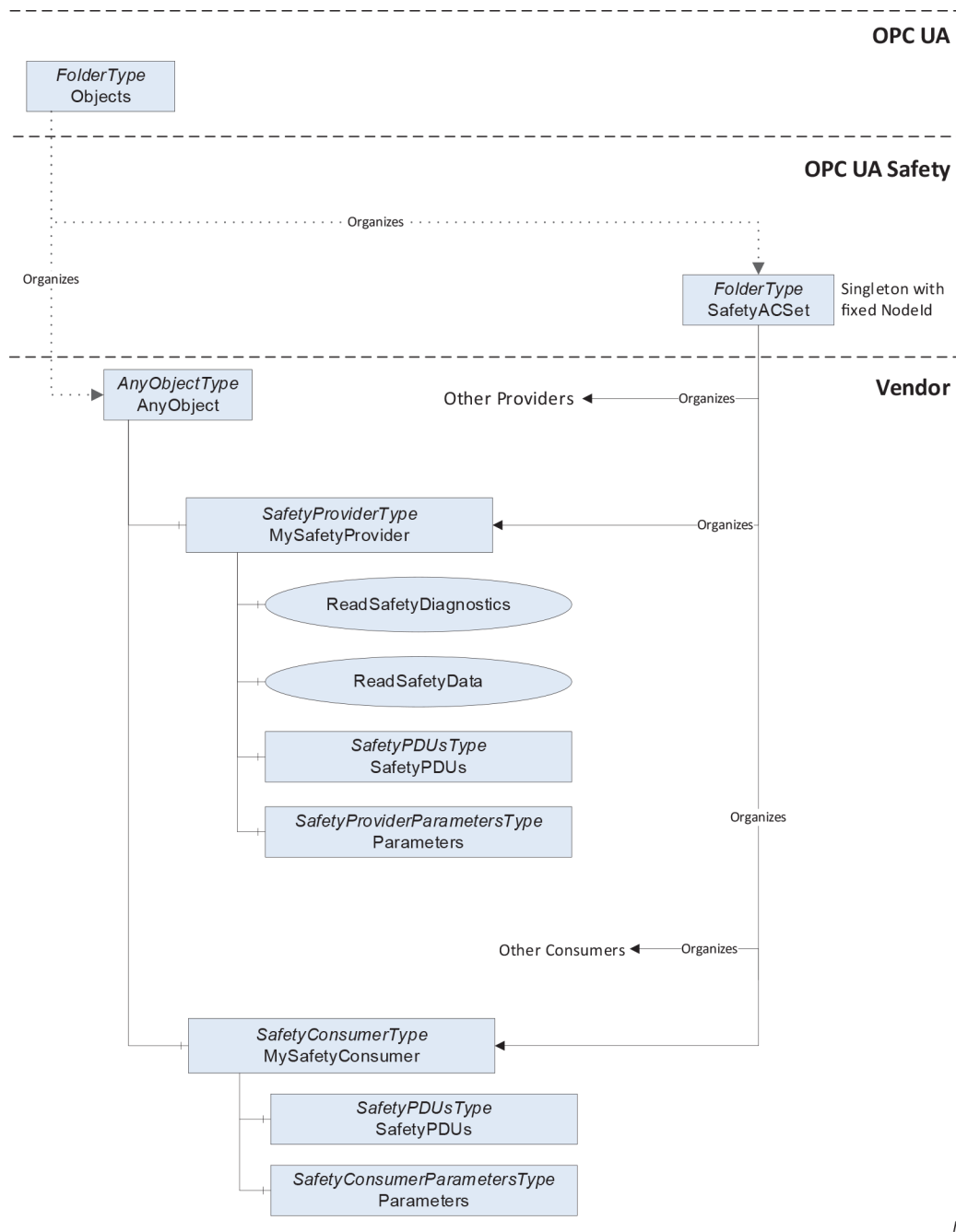


IEC

Figure 3 – Server Objects for OPC UA Safety

NOTE 3 For the input/output arguments of the *Methods* *ReadSafetyData* and *ReadSafetyDiagnostics*, see 6.2.2.3 and 6.2.2.4. For the parameters of the *SafetyProvider* and *SafetyConsumer*, see Figure 6, Table 12, and Table 13. For *RequestSPDU* and *ResponseSPDU*, see Table 7, Table 18, Table 20, and 7.2.1.

Figure 4 shows the instances of *Server Objects* for this document. The *ObjectType* for the *SafetyProviderType* contains *Methods* having outputs of the abstract *Data Type Structure*. Each instance of a *SafetyProvider* requires its own copy of the *Methods* which contain the concrete *DataTypes* for *OutSafetyData* and *OutNonSafetyData*.



IEC

Figure 4 – Instances of Server Objects for this document

6.2.2.2 Safety ObjectType definitions

[RQ6.5] To reduce the number of variations and to alleviate validation testing, the following restrictions apply to instances of *SafetyProviderType* and *SafetyConsumerType* (or instances of *DataTypes* derived from *SafetyProviderType* or *SafetyConsumerType*):

- 1) The references shown in Figure 4 originating at *SafetyProviderType* or *SafetyConsumerType* and below shall be of *ReferenceType HasComponent* (and shall not be derived from *ReferenceType HasComponent*) for *Object References* or *ReferenceType HasProperty* (and shall not be derived from *ReferenceType HasProperty*) for *Property References*.
- 2) As *BrowseNames* (i.e. name and *Namespace*) are used to find *Methods*, the names of *Objects* and *Properties* shall be locally unique.

- 3) The *DataType* of both *Properties* and *MethodArguments* shall be used as specified, and no derived *DataTypes* shall be used (exception: *OutSafetyData* and *OutNonSafetyData*).
- 4) In IEC 62541, the order of *Method* arguments is relevant.

See Table 4 for the definition of the *SafetyObjectsType*.

Table 4 – SafetyObjectsType definition

Attribute	Value				
BrowseName	SafetyObjectsType				
IsAbstract	True				
References	Node class	BrowseName	DataType	TypeDefinition	Modelling rule
Subtype of BaseObjectType					
Conformance units					
SafetySupport					

See Table 5 for the definition of the *SafetyProviderType*.

Table 5 – SafetyProviderType definition

Attribute	Value				
BrowseName	SafetyProviderType				
IsAbstract	False				
References	Node class	BrowseName	DataType	TypeDefinition	Modelling rule
Subtype of SafetyObjectsType					
HasComponent	Method	ReadSafetyData			Optional
HasComponent	Method	ReadSafetyDiagnostics			Optional
HasComponent	Object	SafetyPDUs		SafetyPDUsType	Optional
HasComponent	Object	Parameters		SafetyProviderParametersType	Mandatory
Conformance units					
SafetyProviderParameters					

[RQ6.6] Instances of *SafetyProviderType* shall use non-abstract *DataTypes* for the arguments *OutSafetyData* and *OutNonSafetyData*.

See Table 6 for the definition of the *SafetyConsumerType*.

Table 6 – SafetyConsumerType definition

Attribute	Value				
BrowseName	SafetyConsumerType				
IsAbstract	False				
References	Node class	BrowseName	DataType	TypeDefinition	Modelling rule
Subtype of SafetyObjectsType					
HasComponent	Object	SafetyPDUs		SafetyPDUsType	Optional
HasComponent	Object	Parameters		SafetyConsumerParameters Type	Mandatory
Conformance units					
SafetyConsumerParameters					

6.2.2.3 Method ReadSafetyData

This *Method* is mandatory for the *Facet SafetyProviderServerMapper*. It is used to read *SafetyData* from the *SafetyProvider*. It is in the responsibility of the safety application that this *Method* is not concurrently called by multiple *SafetyConsumers*. Otherwise, the *SafetyConsumer* can receive invalid responses resulting in a safe reaction which can lead to either spurious trips or system unavailability, or both.

See Table 7 for *Method ReadSafetyData*'s arguments and Table 8 for its *AdressSpace* definition.

The *Method* argument *OutSafetyData* has an application-specific *DataType* derived from *Structure*. This *DataType* (including the *DataTypeID*) is expected to be the same in both the *SafetyProvider* and the *SafetyConsumer*. Otherwise, the *SafetyConsumer* will not accept the transferred data and switch to *fail-safe substitute values* instead (see state S16 in Table 34 as well as 7.2.3.2 and 7.2.3.5). The *Method* argument *OutNonSafetyData* has an application-specific *DataType* derived from *Structure*.

Signature

```

ReadSafetyData (
    [in]    UInt32                InSafetyConsumerID,
    [in]    UInt32                InMonitoringNumber,
    [in]    InFlagsType           InFlags,
    [out]   Structure             OutSafetyData,
    [out]   OutFlagsType          OutFlags,
    [out]   UInt32                OutSPDU_ID_1,
    [out]   UInt32                OutSPDU_ID_2,
    [out]   UInt32                OutSPDU_ID_3,
    [out]   UInt32                OutSafetyConsumerID,
    [out]   UInt32                OutMonitoringNumber,
    [out]   UInt32                OutCRC,
    [out]   Structure             OutNonSafetyData)
;

```

Table 7 – ReadSafetyData Method arguments

Argument	Description
InSafetyConsumerID	"Safety Consumer Identifier", see <i>SafetyConsumerID</i> in Table 23.
InMonitoringNumber	" <i>MonitoringNumber</i> of the <i>RequestSPDU</i> ", see 7.2.1.3 and <i>MonitoringNumber</i> in Table 23.
InFlags	"Octet with <i>non-safety-related flags</i> from <i>SafetyConsumer</i> ", see 6.2.3.1.
OutSafetyData	" <i>SafetyData</i> ", see 7.2.1.5.
OutFlags	"Octet with safety-related <i>flags</i> from <i>SafetyProvider</i> ", see 6.2.3.2.
OutSPDU_ID_1	"Safety PDU Identifier Part1", see 7.2.3.2.
OutSPDU_ID_2	"Safety PDU Identifier Part2", see 7.2.3.2.
OutSPDU_ID_3	"Safety PDU Identifier Part3", see 7.2.3.2.
OutSafetyConsumerID	"Safety Consumer Identifier", see <i>SafetyConsumerID</i> in Table 23 and Table 26.
OutMonitoringNumber	<i>MonitoringNumber</i> of the <i>ResponseSPDU</i> , see 7.2.1.9, 7.2.3.1, and Figure 11.
OutCRC	<i>CRC</i> over the <i>ResponseSPDU</i> , see 7.2.3.6.
OutNonSafetyData	"Non-safe data" see 7.2.1.11.

Table 8 – ReadSafetyData Method AddressSpace definition

Attribute	Value				
BrowseName	ReadSafetyData				
References	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
HasProperty	Variable	InputArguments	Argument[]	PropertyType	Mandatory
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Mandatory
Conformance units					
ReadSafetyData					

6.2.2.4 Method ReadSafetyDiagnostics

This *Method* is mandatory for the *Facet SafetyProviderServerMapper* and optional for the *Facet SafetyProviderPubSubMapper*. It is provided for each *SafetyProvider* serving as a *Diagnostic Interface*, see 6.4.3.

See Table 9 for the arguments of *Method ReadSafetyDiagnostics* and Table 10 for its *AddressSpace* definition.

The *Method* arguments *OutSafetyData* and *OutNonSafetyData* are application-specific types derived from *Structure*.

Signature**ReadSafetyDiagnostics (**

```

[out]   UInt32                InSafetyConsumerID,
[out]   UInt32                InMonitoringNumber,
[out]   InFlagsType           InFlags,
[out]   Structure              OutSafetyData,
[out]   OutFlagsType          OutFlags,
[out]   UInt32                OutSPDU_ID_1,
[out]   UInt32                OutSPDU_ID_2,
[out]   UInt32                OutSPDU_ID_3,
[out]   UInt32                OutSafetyConsumerID,
[out]   UInt32                OutMonitoringNumber,
[out]   UInt32                OutCRC,
[out]   Structure              OutNonSafetyData)

```

;

Table 9 – ReadSafetyDiagnostics Method arguments

Argument	Description
InSafetyConsumerID	see Table 7
InMonitoringNumber	see Table 7
InFlags	see Table 7
OutSafetyData	see Table 7
OutFlags	see Table 7
OutSPDU_ID_1	see Table 7
OutSPDU_ID_2	see Table 7
OutSPDU_ID_3	see Table 7
OutSafetyConsumerID	see Table 7
OutMonitoringNumber	see Table 7
OutCRC	see Table 7
OutNonSafetyData	see Table 7

Table 10 – ReadSafetyDiagnostics Method AddressSpace definition

Attribute	Value				
BrowseName	ReadSafetyDiagnostics				
References	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
HasProperty	Variable	OutputArguments	Argument[]	PropertyType	Mandatory
Conformance units					
ReadSafetyDiagnostics					

6.2.2.5 Object SafetyPDUs

This *Object* is mandatory for the *Facet SafetyProviderPubSubMapper* and the *Facet SafetyConsumerPubSubMapper*. It is used by the *SafetyProvider* to subscribe to the *RequestSPDU* and to publish the *ResponseSPDU*. The *DataType* of *RequestSPDU* is structured in the same way as the input arguments of *ReadSafetyData*. The *DataType* of *ResponseSPDU* is structured in the same way as the output arguments of *ReadSafetyData*.

See Table 11 for the definition of the *SafetyPDUsType*.

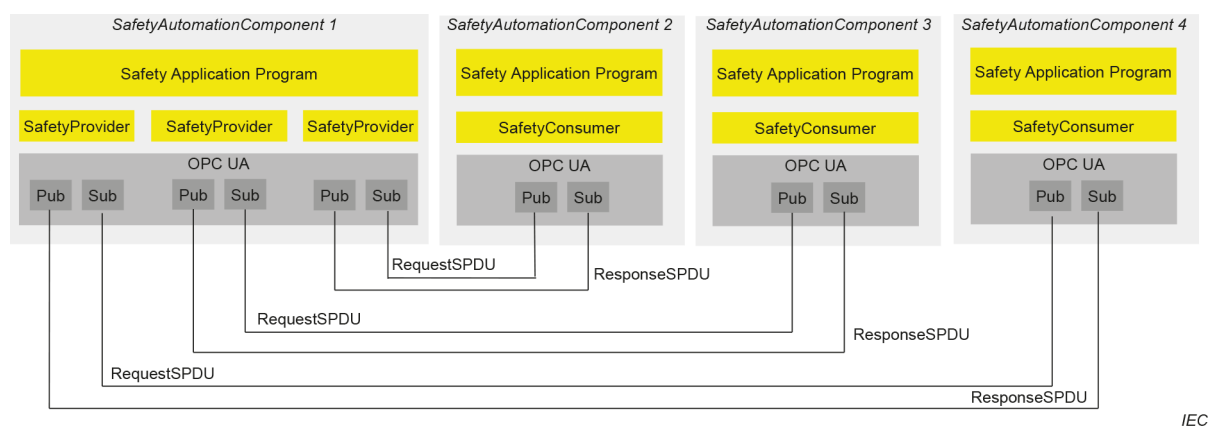
Both variables in the *SafetyPDUsType* have a counterpart within the *Information Model* of the *SafetyConsumer*. The *SafetyConsumer* publishes the *RequestSPDU* and subscribes to the *ResponseSPDU*.

Table 11 – SafetyPDUsType definition

Attribute	Value				
BrowseName	SafetyPDUsType				
IsAbstract	False				
References	Node class	BrowseName	DataType	TypeDefinition	Modelling rule
Subtype of BaseObjectType					
HasComponent	Variable	<RequestSPDU>	RequestSPDUDataType	BaseDataVariableType	Mandatory Placeholder
HasComponent	Variable	<ResponseSPDU>	ResponseSPDUDataType	BaseDataVariableType	Mandatory Placeholder
Conformance units					
SafetyPDUs					

The *Object SafetyPDUs* shall contain exactly one *Reference* to a *Variable* of *DataType RequestSPDUDataType* and exactly one *Reference* to a *Variable* of a subtype of *DataType ResponseSPDUDataType*.

For example, Figure 5 shows a distributed safety application with four *SafetyAutomationComponents*. It is assumed that *SafetyAutomationComponent 1* sends a value to the other three *SafetyAutomationComponents* using three *SafetyProviders*, each comprising a pair of *SPDUs*. For each recipient, there is an individual pair of *SPDUs*.



IEC

Figure 5 – Safety multicast with three recipients using IEC 62541 PubSub

6.2.2.6 Objects *SafetyProviderParameters* and *SafetyConsumerParameters*

Figure 6 shows the safety parameters for the *SafetyProvider* and the *SafetyConsumer*.

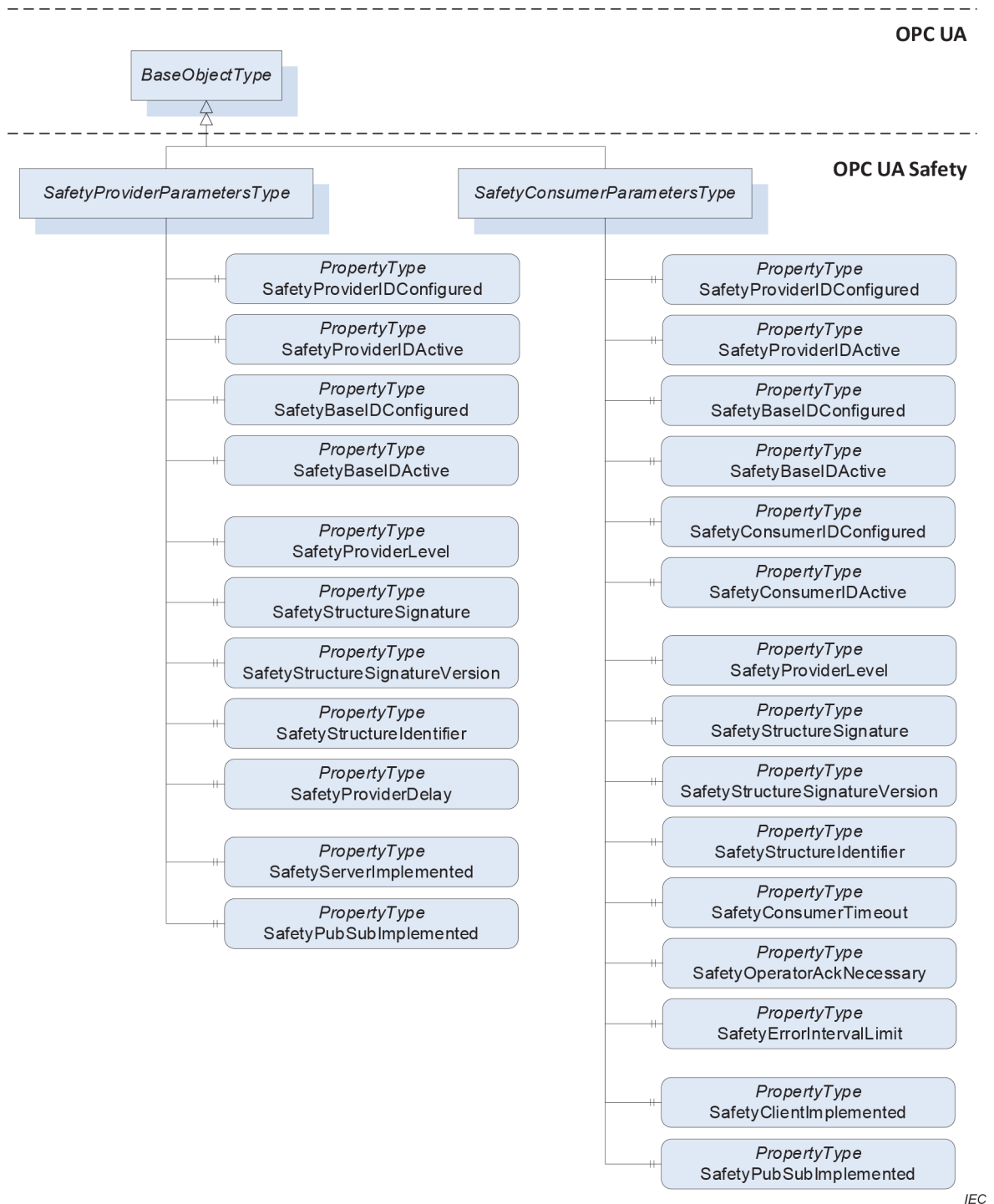


Figure 6 – Safety parameters for the *SafetyProvider* and the *SafetyConsumer*

Table 12 shows the definition for the *SafetyProviderParametersType*. Refer to 6.3.3.3 for more details on the *Safety Parameter Interface (SPI)* of the *SafetyProvider*.

Table 12 – SafetyProviderParametersType definition

Attribute	Value				
BrowseName	SafetyProviderParametersType				
IsAbstract	False				
References	Node class	BrowseName	DataType	TypeDefinition	Modelling rule
Subtype of BaseObjectType					
HasProperty	Variable	SafetyProviderIDConfigured	UInt32	PropertyType	Mandatory
HasProperty	Variable	SafetyProviderIDActive	UInt32	PropertyType	Mandatory
HasProperty	Variable	SafetyBaseIDConfigured	Guid	PropertyType	Mandatory
HasProperty	Variable	SafetyBaseIDActive	Guid	PropertyType	Mandatory
HasProperty	Variable	SafetyProviderLevel	Byte	PropertyType	Mandatory
HasProperty	Variable	SafetyStructureSignature	UInt32	PropertyType	Mandatory
HasProperty	Variable	SafetyStructureSignatureVersion	UInt16	PropertyType	Mandatory
HasProperty	Variable	SafetyStructureIdentifier	String	PropertyType	Mandatory
HasProperty	Variable	SafetyProviderDelay	UInt32	PropertyType	Mandatory
HasProperty	Variable	SafetyServerImplemented	Boolean	PropertyType	Mandatory
HasProperty	Variable	SafetyPubSubImplemented	Boolean	PropertyType	Mandatory
Conformance units					
SafetyProviderParameters					

The parameters for *SafetyProviderID* and *SafetyBaseID* exist in pairs for "Configured" and "Active" states:

- *SafetyProviderIDConfigured* and *SafetyProviderIDActive*,
- *SafetyBaseIDConfigured* and *SafetyBaseIDActive*.

The "[...]Configured" parameters shall always deliver the values as configured via the *SPI*. The "[...]Active" parameters shall deliver:

- the corresponding "[...]Configured" values if the system is still offline;
- the values which have been set during runtime via the *SAPI* parameters (*SafetyProviderID*, *SafetyBaseID*);
- the corresponding "[...]Configured" values if the active values have been set to zero via the *SAPI* parameters (*SafetyProviderID*, *SafetyBaseID*).

The *Property SafetyBaseIDConfigured* is shared for all *SafetyProviders* with the same *SafetyBaseIDConfigured* value. If multiple instances of *SafetyObjectsType* are running on the same *Node*, it is a viable optimization that a *Property SafetyBaseIDConfigured* is referenced by either multiple *SafetyProviders* or *SafetyConsumers*, or both.

For releases up to Release 2.0 of the document, the value for the *SafetyStructureSignatureVersion* shall be 0x0001 (see RQ7.21 in 7.2.3.5).

Table 13 shows the definition of the *SafetyConsumerParametersType*. The *Properties SafetyStructureIdentifier* and *SafetyStructureSignatureVersion* are optional, because *SafetyStructureSignature* is typically calculated in an offline engineering tool. For small devices, it could be beneficial to only upload the *SafetyStructureSignature* to the device, but not *SafetyStructureIdentifier* and *SafetyStructureSignatureVersion* in order to save either bandwidth or memory, or both. Refer to 6.3.4.4 for more details on the *Safety Parameter Interface (SPI)* of the *SafetyConsumer*.

Table 13 – SafetyConsumerParametersType definition

Attribute	Value				
BrowseName	SafetyConsumerParametersType				
IsAbstract	False				
References	Node class	BrowseName	DataType	TypeDefinition	Modelling rule
Subtype of BaseObjectType					
HasProperty	Variable	SafetyProviderIDConfigured	UInt32	PropertyType	Mandatory
HasProperty	Variable	SafetyProviderIDActive	UInt32	PropertyType	Mandatory
HasProperty	Variable	SafetyBaseIDConfigured	Guid	PropertyType	Mandatory
HasProperty	Variable	SafetyBaseIDActive	Guid	PropertyType	Mandatory
HasProperty	Variable	SafetyConsumerIDConfigured	UInt32	PropertyType	Mandatory
HasProperty	Variable	SafetyConsumerIDActive	UInt32	PropertyType	Mandatory
HasProperty	Variable	SafetyProviderLevel	Byte	PropertyType	Mandatory
HasProperty	Variable	SafetyStructureSignature	UInt32	PropertyType	Mandatory
HasProperty	Variable	SafetyStructureSignatureVersion	UInt16	PropertyType	Optional
HasProperty	Variable	SafetyStructureIdentifier	String	PropertyType	Optional
HasProperty	Variable	SafetyConsumerTimeout	UInt32	PropertyType	Mandatory
HasProperty	Variable	SafetyOperatorAckNecessary	Boolean	PropertyType	Mandatory
HasProperty	Variable	SafetyErrorIntervalLimit	UInt16	PropertyType	Mandatory
HasProperty	Variable	SafetyClientImplemented	Boolean	PropertyType	Mandatory
HasProperty	Variable	SafetyPubSubImplemented	Boolean	PropertyType	Mandatory
Conformance units					
SafetyConsumerParameters					

The parameters for *SafetyProviderID*, *SafetyBaseID* and *SafetyConsumerID* exist in pairs for "Configured" and "Active" states: *SafetyProviderIDConfigured* and *SafetyProviderIDActive*, *SafetyBaseIDConfigured* and *SafetyBaseIDActive*, and *SafetyConsumerIDConfigured* and *SafetyConsumerIDActive*.

The "[...]Configured" parameters shall always deliver the values as configured via the *SPI*. The "[...]Active" parameters shall deliver:

- the corresponding "[...]Configured" values if the system is still offline;
- the values which have been set during runtime via the *SAPI* parameters (*SafetyProviderID*, *SafetyBaseID*, *SafetyConsumerID*);
- the corresponding "[...]Configured" values if the active values have been set to zero via the *SAPI* parameters (*SafetyProviderID*, *SafetyBaseID*, *SafetyConsumerID*).

6.2.3 DataType definition

6.2.3.1 InFlagsType

The *InFlagsType* a subtype of the *Byte DataType* with the *OptionSetValues Property* defined. The *InFlagsType* is formally defined in Table 14.

CommunicationError can be used as a trigger, e.g. for a communication analysis tool. It is not forwarded to the safety application by the *SafetyProvider*. If *CommunicationError* is necessary in the safety application, bidirectional communication can be implemented and the value of *CommunicationError* can be put into the user data.

Table 14 – InFlagsType values

Value	Bit no.	Description
CommunicationError	0	0: No error 1: An error was detected in the previous <i>ResponseSPDU</i> .
OperatorAckRequested	1	Used to inform the <i>SafetyProvider</i> that operator acknowledgment is requested.
FSV_Activated	2	Is used for conformance test of <i>SafetyConsumer.SAPI.FSV_Activated</i>

Bits 3 to 7 are reserved for future use and shall be set to zero by the *SafetyConsumer*. They shall not be evaluated by the *SafetyProvider*.

The *InFlagsType* representation in the *AddressSpace* is defined in Table 15.

Table 15 – InFlagsType definition

Attribute		Value			
BrowseName		InFlagsType			
IsAbstract		False			
References	NodeClass	BrowseName	DataType	TypeDefinition	Other
Subtype of the <i>Byte DataType</i> defined in IEC 62541-3.					
0:HasProperty	Variable	0:OptionSetValues	0:LocalizedText []	0:PropertyType	
Conformance units					
SafetySupport					

6.2.3.2 OutFlagsType

The *OutFlagsType* is a subtype of the *Byte DataType* with the *OptionSetValues Property* defined. The *OutFlagsType* is formally defined in Table 16.

Table 16 – OutFlagsType values

Value	Bit no.	Description
<i>OperatorAckProvider</i>	0	Operator acknowledgment at the provider, hereby forwarded to the <i>SafetyConsumer</i> , see <i>OperatorAckProvider</i> in the <i>SAPI</i> of the <i>SafetyProvider</i> , 6.3.3.2.
<i>ActivateFSV</i>	1	Activation of <i>fail-safe</i> values by the safety application at the <i>SafetyProvider</i> , hereby forwarded to the <i>SafetyConsumer</i> , see <i>ActivateFSV</i> in the <i>SAPI</i> of the <i>SafetyProvider</i> , 6.3.3.2
<i>TestModeActivated</i>	2	<i>Enabling and disabling of test mode in the SafetyProvider</i> , hereby forwarded to the <i>SafetyConsumer</i> , see <i>EnableTestMode</i> in the <i>SAPI</i> of the <i>SafetyProvider</i> , 6.3.3.2

Bits 3 to 7 are reserved for future use and shall be set to zero by the *SafetyProvider*. They shall not be evaluated by the *SafetyConsumer*.

The *OutFlagsType* representation in the *AddressSpace* is defined in Table 17.

Table 17 – OutFlagsType definition

Attribute		Value			
BrowseName		OutFlagsType			
IsAbstract		False			
References	NodeClass	BrowseName	DataType	TypeDefinition	Other
Subtype of the <i>Byte DataType</i> defined in IEC 62541-3.					
0:HasProperty	Variable	0:OptionSetValues	0:LocalizedText []	0:PropertyType	
Conformance units					
SafetySupport					

6.2.3.3 RequestSPDUDataType

Table 18 shows the definition of the *RequestSPDUDataType*. The Prefix "In" is interpreted from the *SafetyProvider's* point of view and is used in a consistent manner to the parameters of the *Method ReadSafetyData* (see 6.2.2.3).

Table 18 – RequestSPDUDataType structure

Name	Type	Description
RequestSPDUDataType	structure	
InSafetyConsumerID	UInt32	See corresponding <i>Method</i> argument in Table 7.
InMonitoringNumber	UInt32	See corresponding <i>Method</i> argument in Table 7.
InFlags	InFlagsType	See corresponding <i>Method</i> argument in Table 7.

The representation in the *AddressSpace* of the *RequestSPDUDataType* is defined in Table 19.

Table 19 – RequestSPDUDataType definition

Attributes	Value				
BrowseName	RequestSPDUDataType				
IsAbstract	False				
References	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
Subtype of <i>Structure</i> defined in IEC 62541-3.					
Conformance units					
SafetyPDUs					

6.2.3.4 ResponseSPDUDataType

Table 20 shows the *ResponseSPDUDataType Structure*. The Prefix "Out" is interpreted from the *SafetyProvider's* point of view and is used in a consistent manner to the parameters of the *Method ReadSafetyData* (see 6.2.2.3).

Table 20 – ResponseSPDUDataType structure

Name	Type	Description
ResponseSPDUDataType	structure	
OutFlags	OutFlagsType	See corresponding <i>Method</i> argument in Table 7.
OutSPDU_ID_1	UInt32	See corresponding <i>Method</i> argument in Table 7.
OutSPDU_ID_2	UInt32	See corresponding <i>Method</i> argument in Table 7.
OutSPDU_ID_3	UInt32	See corresponding <i>Method</i> argument in Table 7.
OutSafetyConsumerID	UInt32	See corresponding <i>Method</i> argument in Table 7.
OutMonitoringNumber	UInt32	See corresponding <i>Method</i> argument in Table 7.
OutCRC	UInt32	See corresponding <i>Method</i> argument in Table 7.

[RQ6.7] To define the concrete *DataType* for the *ResponseSPDU* (which specifies the concrete *DataTypes* for *SafetyData* and *NonSafetyData*, respectively), proceed as follows: (1) Derive a concrete *DataType* from the abstract *ResponseSPDUDataType*. (2) In doing so, add the following fields to the *Structure* in the given order: (a) First, field *OutSafetyData* with the concrete *Structure DataType* for the *SafetyData* (see 7.2.1.5). (b) Second, field *NonSafetyData* with the concrete *Structure DataType* for the *NonSafetyData* (or a placeholder *DataType*, see requirement RQ6.8).

[RQ6.8] To avoid possible problems with empty *Structures*, the dummy *Structure NonSafetyDataPlaceholder* shall be used as *DataType* for *OutNonSafetyData* when no *NonSafetyData* is used. The *DataType Node* defining this *Structure* has a fixed *NodeID* and contains a single *Boolean*.

The representation in the *AddressSpace* of the *ResponseSPDUDataType* is defined in Table 21.

Table 21 – ResponseSPDUDataType definition

Attributes	Value				
BrowseName	ResponseSPDUDataType				
IsAbstract	True				
References	NodeClass	BrowseName	DataType	TypeDefinition	ModellingRule
Subtype of <i>Structure</i> defined in IEC 62541-3.					
Conformance units					
SafetyPDUs					

6.2.3.5 NonSafetyDataPlaceholderDataType

Table 22 shows the definition of the *NonSafetyDataPlaceholderDataType*. The receiver shall not evaluate the value of 'dummy'.

Table 22 – NonSafetyDataPlaceholderDataType structure

Name	Type	Description
NonSafetyDataPlaceholderDataType	Structure	
Dummy	Boolean	Dummy <i>Variable</i> to avoid empty structures.

6.2.4 SafetyProvider version

Future versions may use different identifiers (such as *ReadSafetyDataV2* for the *Method* when using *Client/Server* communication or *RequestSPDUV2DataType* and *ResponseSPDUV2DataType* for the *SPDU DataTypes* when using *PubSub* communication), allowing a *SafetyProvider* to implement multiple versions of this document at the same time. Hence, the same *SafetyProvider* can be accessed by *SafetyConsumers* of different versions.

6.2.5 DataTypes and length of SafetyData

This document supports sending of the *Built-in* and *Simple DataTypes* specified in the IEC 62541 series (see IEC 62541-3 and IEC 62541-6) within *SafetyData*. The supported *DataTypes* are vendor-specific.

[RQ6.9] Only scalar *DataTypes* shall be used. Arrays are currently not supported by this document.

The supported maximum length of the *SafetyData* is vendor-specific but still limited to 1 500 octets. Typical values for the maximum length include 1 octet, 16 octets, 64 octets, 256 octets, 1 024 octets, and 1 500 octets.

[RQ6.10] For controller-like devices, the supported *DataTypes* and the maximum length of the *SafetyData* shall be listed in the user manual.

[RQ6.11] For the *DataType Boolean*, the value 0x01 shall be used for 'true' and the value 0x00 shall be used for 'false'.

It is recommended to send multiple *Booleans* in separate variables. However, in small devices, it can be necessary to combine a set of 8 *Booleans* in one *Variable* for performance reasons. In this case, the *DataType Byte* can be used.

6.2.6 Connection establishment

This document uses the IEC 62541 services for connection establishment, it poses no additional requirement to these services.

This version of the document describes configuration only at engineering time. This means that the parameters defined in the *SPI* (see 6.3.3.3 and 6.3.4.4) are read-only via the interface described in this document. Changing of parameters is expected to be done in a safety-related way, using the respective tools and interfaces provided by the vendor. Future versions of this document may specify a vendor-independent interface for configuration.

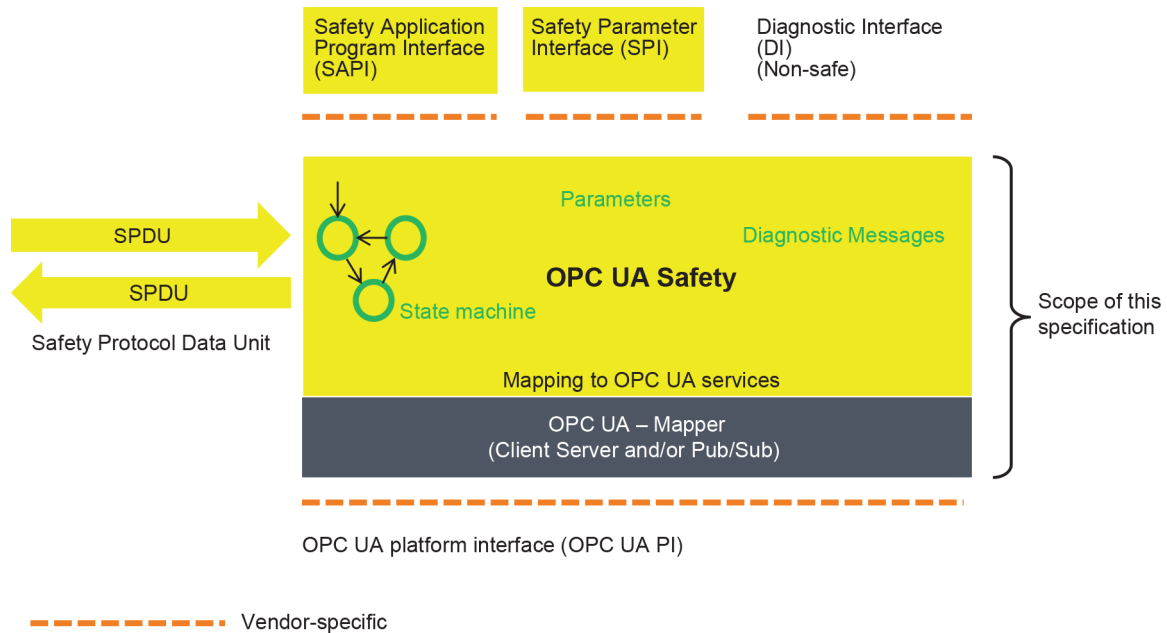
6.3 Service interfaces

6.3.1 Overview

Figure 7 gives an overview of the *safety communication layer* and its interfaces. It thereby also shows the scope of this document. The main function of the layer services is the state machine which handles the protocol. The state machines interact with the following interfaces:

- The *Safety Application Program Interface (SAPI)* is accessed by the safety application for exchanging *SafetyData* during runtime.
- The *Safety Parameter Interface (SPI)* is accessed during commissioning for setting safety parameters such as IDs or the timeout value in the *SafetyConsumer*.
- The *non-safety* related *Diagnostic Interface (DI)* can be accessed at runtime for troubleshooting the safety communication.
- The *OPC UA Platform Interface (OPC UA PI)* connects the *SCL* to the *non-safe* IEC 62541 stack and is used during runtime.

The interfaces (*SAPI*, *SPI*, *DI* and *OPC UA PI*) described in 6.3 are abstract and informative. They represent logical data inputs and outputs to this layer that are necessary for the proper operation of the state machine. No normative, concrete mappings are specified. The concrete implementations are vendor-specific and do not necessarily exactly match the abstract interfaces described in this document.



IEC

Figure 7 – Safety communication layer overview

6.3.2 OPC UA Platform interface (OPC UA PI)

The state machines of this document are independent from the actual IEC 62541 services used for data transmission. This is accomplished by introducing a so-called *OPC UA mapper*, serving as an interface between the *safety communication layer* and the IEC 62541 stack.

The mapper can either make use of IEC 62541 *Client/Server* and remote *Method* invocation or the publishing of and subscribing to remote *Variables* as defined in IEC 62541-14. The requirements on the implementation of the mapper are implicitly given in 6.2.

6.3.3 SafetyProvider interfaces

6.3.3.1 General

Figure 8 shows an overview of the *SafetyProvider* interfaces. The *SAPI* is specified in 6.3.3.2, the *SPI* is specified in 6.3.3.3.

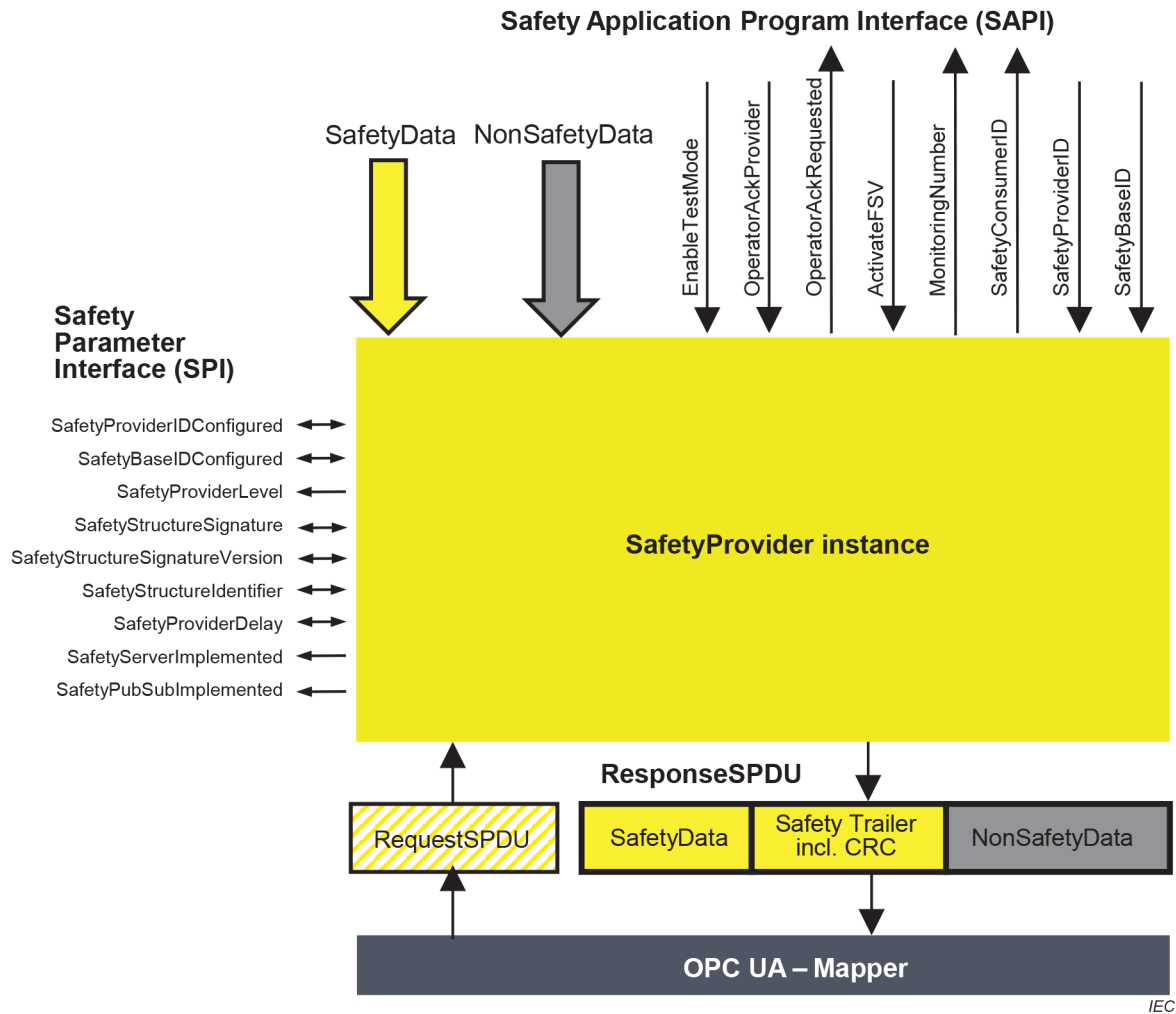


Figure 8 – SafetyProvider interfaces

6.3.3.2 SAPI of SafetyProvider

[RQ6.12] The *SAPI* of the *SafetyProvider* represents the *safety communication layer* services of the *SafetyProvider*. Table 23 lists all inputs and outputs of the *SAPI* of the *SafetyProvider*. Each *SafetyProvider* shall implement the *SAPI* as shown in Table 23, however, the details are vendor-specific.

Table 23 – SAPI of the SafetyProvider

SAPI Term	Type	I/O	Definition
SafetyData	Structure	I	This input is used to accept the user data which is then transmitted as <i>SafetyData</i> in the <i>SPDU</i>. NOTE 1 Whenever a new <i>MNR</i> is received from a <i>SafetyConsumer</i>, the state machine of the <i>SafetyProvider</i> will read a new value of the <i>SafetyData</i> from its corresponding Safety Application and use it until the next <i>MNR</i> is received. If no valid user data is available at the Safety Application, ActivateFSV can be set to "1" by the Safety Application.
NonSafetyData	Structure	I	Used to consistently transmit <i>NonSafetyData</i> values (e.g. diagnostic information) together with safe data, see 7.2.1.11
EnableTestMode	Boolean	I	By setting this input to "1" the remote <i>SafetyConsumer</i> is informed (by Bit 2 in <i>ResponseSPDU.Flags</i>, see 6.2.3.2) that the <i>SafetyData</i> are test data, and are not to be used for safety-related decisions. NOTE 2 This document is intended for implementation in safety devices exclusively, see requirement RQ4.1.
OperatorAckProvider	Boolean	I	This input is used to implement an operator acknowledgment on the <i>SafetyProvider</i> side. The value will be forwarded to the <i>SafetyConsumer</i> , where it can be used to trigger a return from <i>fail-safe</i> substitute values (<i>FSV</i>) to actual <i>process values</i> (<i>PV</i>), see B.3.4.
OperatorAckRequested	Boolean	O	Indicates that an operator acknowledge is requested by the <i>SafetyConsumer</i> . This <i>flag</i> is received within the <i>RequestSPDU</i> .
ActivateFSV (<i>Fail-safe</i> substitute values)	Boolean	I	By setting this input to "1" the <i>SafetyConsumer</i> is instructed (via Bit 1 in <i>ResponseSPDU.Flags</i>, see 6.2.3.2) to deliver <i>FSV</i> instead of <i>PV</i> to the safety application program. NOTE 3 If the replacement of <i>process values</i> by <i>FSV</i> is to be controllable in a more fine-grained way, this can be realized by using <i>qualifiers</i> within the <i>SafetyData</i>, see 6.3.6.
SafetyConsumerID	UInt32	O	This output yields the ConsumerID used in the last access to this <i>SafetyProvider</i> by a <i>SafetyConsumer</i> (see 6.2.2.3). Since all safety-related checks are executed by an implementation of this document, the safety application is not required to check this <i>SafetyConsumerID</i>.
MonitoringNumber	UInt32	O	This output yields the <i>MonitoringnNumber</i> (<i>MNR</i>). It is updated whenever a new request comes in from the <i>SafetyConsumer</i>. Since all safety-related checks are executed by an implementation of this document, the safety application is not required to check this <i>MonitoringNumber</i>.
SafetyProviderID	UInt32	I	For dynamic systems, this input can be set to a non-zero value. In this case, the <i>SafetyProvider</i> uses this value instead of the value from the <i>SPI</i> parameter <i>SafetyProviderIDConfigured</i>. If the value is changed to "0", the value of parameter <i>SafetyProviderIDConfigured</i> from the <i>SPI</i> will be used (again). See Figure 8, 3.1.2.15, and 9.1.1. For static systems, this input is usually always kept at value "0".
SafetyBaseID	Guid	I	For dynamic systems, this input can be set to a non-zero value. In this case, the <i>SafetyProvider</i> uses this value instead of the value of the <i>SPI</i> parameter <i>SafetyBaseIDConfigured</i>. If the value is changed to "0", the value of parameter <i>SafetyBaseIDConfigured</i> from the <i>SPI</i> will be used (again). See Figure 8, 3.1.2.14, and 9.1.1. For static systems, this input is usually always kept at value "0".

For additional information regarding operator acknowledgment use cases, see Clause B.3.

6.3.3.3 SPI of SafetyProvider

[RQ6.13a] Each *SafetyProvider* shall implement the parameters and constants [RQ6.13b] as shown in Table 24. The parameters (R/W in column "Access") can be set via the *SPI*, whereas the constants (R in column "Access") are read-only. The mechanisms for setting the parameters are vendor-specific. The attempt of setting a parameter to a value outside its range, or of the setting of a read-only parameter, shall not become effective, and a diagnostic message should be shown when appropriate. The values of the constants depend on the way the *SafetyProvider* is implemented. They never change and are therefore not writable via any of the interfaces.

Table 24 – SPI of the SafetyProvider

Identifier	Type	Range	Initial value (before configuration)	Access	Note
SafetyProviderIDConfigured	UInt32	0x0 to 0xFFFF FFFF	0x0	R/W	<i>SafetyProviderID</i> of the <i>SafetyProvider</i> that is normally used, see 3.1.2.13 and 3.1.2.15 and 9.1.1. For dynamic systems, the safety application program can overwrite this ID by providing a non-zero value at the input <i>SafetyProviderID</i> of the <i>SafetyProvider's</i> <i>SAPI</i>. This runtime value can be queried using the <i>SafetyProviderIDActive</i> parameter. See 6.2.2.6 for details on configured and active values. NOTE 1 If both the values provided at the <i>SPI</i> and the <i>SAPI</i> are 0x0, this means that the <i>SafetyProvider</i> is not properly configured. <i>SafetyConsumers</i> will never try to communicate with <i>SafetyProviders</i> having a <i>SafetyProviderID</i> of 0x0, see Transitions T13/T27 in Table 35 and the macro <ParametersOK?> in Table 33.

Identifier	Type	Range	Initial value (before configuration)	Access	Note
SafetyBaseIDConfigured	Guid	Any value which can be represented with sixteen octets	All sixteen octets are 0x00	R/W	<i>SafetyBaseID</i> of the <i>SafetyProvider</i> that is normally used, see 3.1.2.14 and 3.1.2.13 and 9.1.1. For dynamic systems, the safety application program can overwrite this ID by providing a non-zero value at the input <i>SafetyBaseID</i> of the <i>SafetyProvider's SAPI</i>. This runtime value can be queried using the <i>SafetyBaseIDActive</i> parameter. See 6.2.2.6 for details on configured and active values. NOTE 2 If both the values provided at the <i>SPI</i> and the <i>SAPI</i> are 0x0, this means that the <i>SafetyProvider</i> is not properly configured. <i>SafetyConsumers</i> will never try to communicate with <i>SafetyProviders</i> having a <i>SafetyBaseID</i> of 0x0, see Transitions T13/T27 in Table 35 and the macro <ParametersOK?> in Table 33. See 9.1.1 for more information on <i>GUID</i>.
SafetyProviderLevel	Byte	0x01 to 0x04	n.a.	R	The <i>SIL</i> the <i>SafetyProvider</i> implementation (hardware and software) is capable of, see Figure 9. NOTE 3 It is independent from the generation of the <i>SafetyData</i> at <i>SAPI</i>. NOTE 4 The <i>SafetyProviderLevel</i> is used to distinguish devices of a different <i>SIL</i>. As a result, <i>SPDUs</i> coming from a device with a low <i>SIL</i> will never be accepted when a <i>SafetyConsumer</i> is parameterized to implement a safety function with a high <i>SIL</i>.

Identifier	Type	Range	Initial value (before configuration)	Access	Note
SafetyStructureSignature	UInt32	0x0 to 0xFFFF FFFF	0x0	R/W	Signature of the <i>SafetyData</i> structure, for calculation see 7.2.3.5. NOTE 5 "0" would not be a valid signature and thus indicates a <i>SafetyProvider</i> which is not properly configured. <i>SafetyConsumers</i> will never try to communicate with <i>SafetyProviders</i> having a <i>SafetyStructureSignature</i> of 0x0, see Transitions T13/T27 in Table 35 and the macro <ParametersOK?> in Table 33.
SafetyStructureSignature Version	UInt16	0x1	0x1	R/W	Version used to calculate <i>SafetyStructureSignature</i> , see 7.2.3.5
SafetyStructureIdentifier	String	all strings	"" (the empty string)	R/W	Identifier describing the <i>Data Type</i> of the <i>SafetyData</i> , see 7.2.3.5.
SafetyProviderDelay	UInt32	0x0 to 0xFFFF FFFF	0x0	R/W	In microseconds (μ s). It can be set during the engineering phase of the <i>SafetyProvider</i> or set during online configuration as well. <i>SafetyProviderDelay</i> is the maximum time at the <i>SafetyProvider</i> from receiving the <i>RequestSPDU</i> to start the transmission of <i>ResponseSPDU</i> , see 8.1. The parameter <i>SafetyProviderDelay</i> has no influence on the functional behaviour of the <i>SafetyProvider</i> . Therefore, it is not necessary for this value to be generated in a safety-related way. However, it will be provided in the IEC 62541 <i>Information Model</i> of a <i>SafetyProvider</i> to inform about its worst-case delay time. The value can be used during commissioning to check whether the timing behaviour of the <i>SafetyProvider</i> is suitable to fulfill the watchdog delay of the corresponding <i>SafetyConsumer</i> .

Identifier	Type	Range	Initial value (before configuration)	Access	Note
SafetyServerImplemented	Boolean	0x0 or 0x1	n.a.	R	This read-only parameter indicates whether the <i>SafetyProvider</i> has implemented the <i>Server</i> part of IEC 62541 <i>Client/Server</i> communication (see 5.4): 1: <i>Server</i> for IEC 62541 <i>Client/Server</i> communication is implemented. 0: <i>Server</i> for IEC 62541 <i>Client/Server</i> communication is not implemented. The corresponding <i>Facets</i> are <i>SafetyProviderServer</i> and <i>SafetyProviderServer-Mapper</i>.
SafetyPubSubImplemented	Boolean	0x0 or 0x1	n.a.	R	This read-only parameter indicates whether the <i>SafetyProvider</i> has implemented the necessary publishers and subscribers for IEC 62541 <i>PubSub</i> communication (see 5.4): 1: IEC 62541 <i>PubSub</i> communication is implemented. 0: IEC 62541 <i>PubSub</i> communication is not implemented. The corresponding <i>Facets</i> are <i>SafetyProviderPubSub</i> and <i>SafetyProviderPubSub-Mapper</i>.

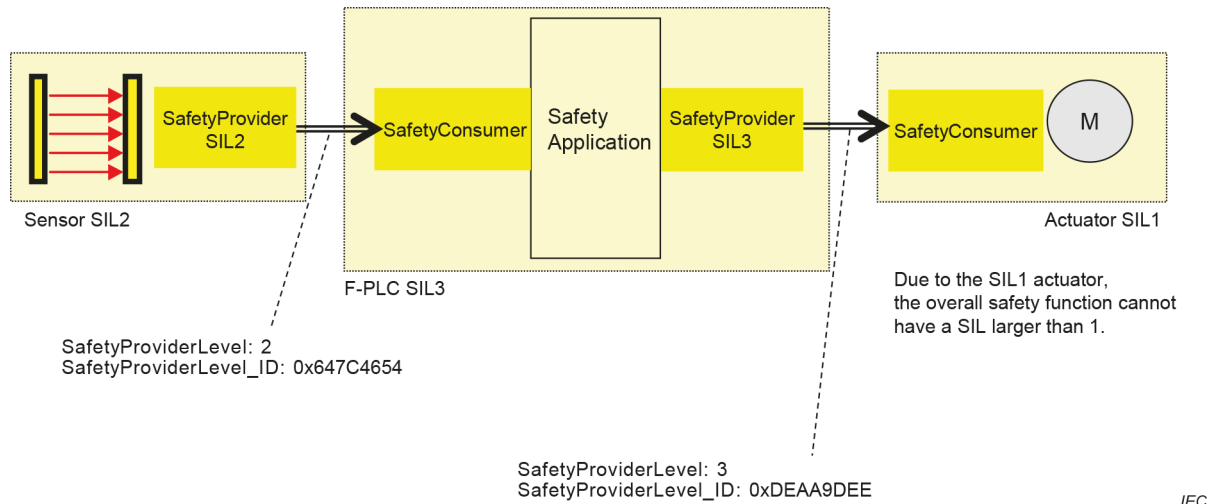


Figure 9 – Example combinations of SIL capabilities

The constant *SafetyProviderLevel* determines the value that is used for *SafetyProviderLevel_ID* when calculating the *SPDU_ID*, see 7.2.3.3.

NOTE *SafetyProviderLevel* is defined as the *SIL* the *SafetyProvider* implementation (hardware and software) is capable of, not to be confused with the *SIL* of the implemented safety function. For instance, Figure 9 shows a safety function which is implemented using a *SIL2*-capable sensor, a *SIL3*-capable PLC, and a *SIL1*-capable actuator. The overall *SIL* of the safety function is considered to be *SIL1*. Nevertheless, the *SafetyProvider* implemented on the sensor will use the constant value "2" as *SafetyProviderLevel*, whereas the *SafetyProvider* implemented on the PLC will use the constant value "3" as *SafetyProviderLevel*.

It is necessary for the respective *SafetyConsumers* (on the PLC and the actuator) to know the *SafetyProviderLevel* of their *SafetyProviders* in order to check the *SPDU_ID* (see 7.2.3.2).

6.3.4 SafetyConsumer interfaces

6.3.4.1 General

Figure 10 shows an overview of the *SafetyConsumer* interfaces. The *Safety Application Program Interface* (SAPI) is specified in 6.3.4.2, the *Safety Parameter Interface* (SPI) is specified in 6.3.4.4.

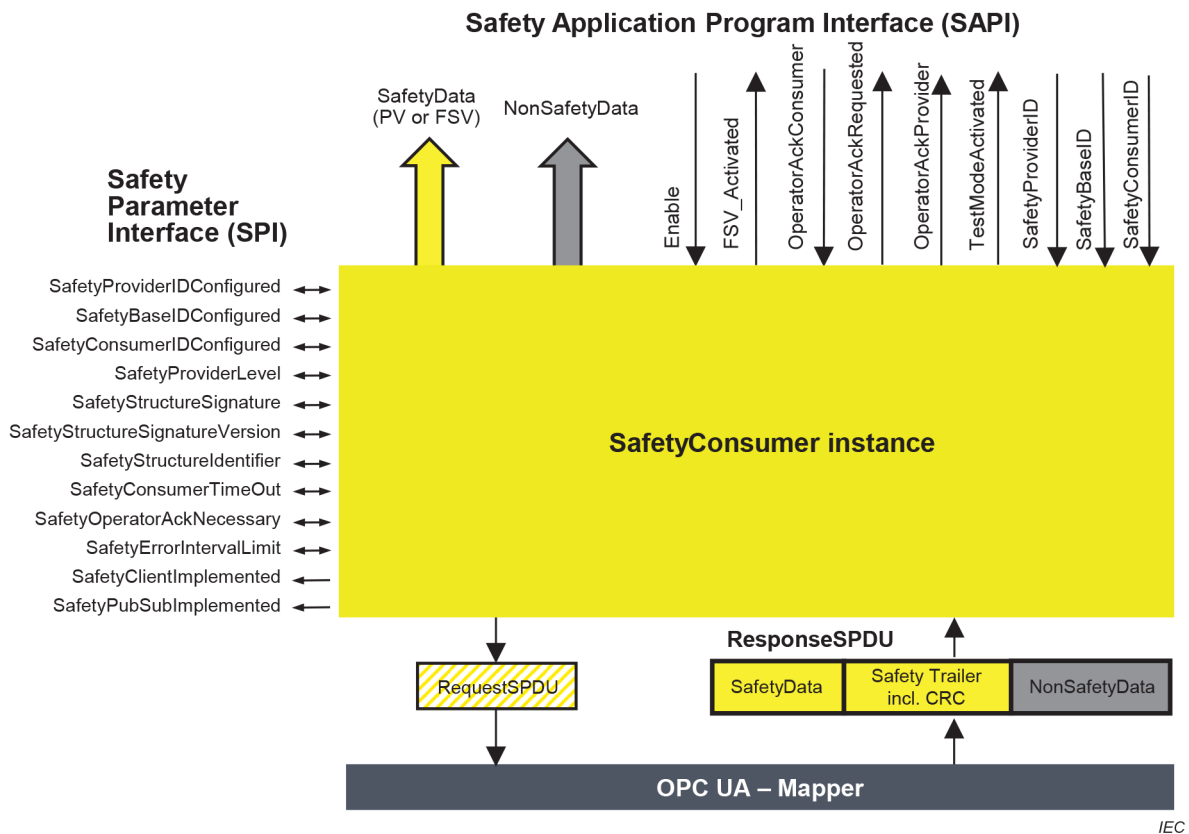


Figure 10 – SafetyConsumer interfaces

6.3.4.2 SAPI of SafetyConsumer

The *SAPI* of the *SafetyConsumer* represents the *safety communication layer* services of the *SafetyConsumer*. Table 25 lists all inputs and outputs of the *SAPI* of the *SafetyConsumer*.

[RQ6.14] Each *SafetyConsumer* shall implement the *SAPI* as shown in Table 25, however, the details are vendor-specific.

Table 25 – SAPI of the SafetyConsumer

SAPI Term	Type	I/O	Definition
SafetyData	Structure	O	This output either delivers the <i>process values</i> received from the <i>SafetyProvider</i> in the <i>SPDU</i> field <i>SafetyData</i> , or <i>FSV</i> .
NonSafetyData	Structure	O	This output delivers the <i>non-safety process values</i> (e.g. diagnostic information) which were sent together with safe data, see 7.2.1.11
Enable	Boolean	I	By changing this input to "0" the <i>SafetyConsumer</i> will change each and every <i>Variable</i> of the <i>SafetyData</i> to "0" ¹ and stop sending requests to the <i>SafetyProvider</i> . When changing <i>Enable</i> to "1" the <i>SafetyConsumer</i> will restart safe communication. The variable can be used to delay the start of the safety communication according to this document, after power on until "IEC 62541 connection ready" is set. The delay time is not monitored while enable is set to "0".
FSV_Activated	Boolean	O	This output indicates via "1", that on the output <i>SafetyData</i> <i>FSV</i> (all binary "0") are provided ¹ . NOTE 1 A <i>ResponseSPDU</i> which is checked with an error results to <i>FSV_Activated</i> being set to "1", see T24 in Table 35. There could be other reasons.

SAPI Term	Type	I/O	Definition
OperatorAckConsumer	Boolean	I	For motivation, see 6.3.4.3. After an indication of OperatorAckRequested this input can be used to signal an operator acknowledgment. By changing this input from "0" to "1" (rising edge) the <i>SafetyConsumer</i> is instructed to switch <i>SafetyData</i> from <i>FSV</i> to <i>PV</i>. OperatorAckConsumer is processed only if this rising edge arrives after OperatorAckRequested was set to "1", see Figure 19. If a rising edge of OperatorAckConsumer arrives before OperatorAckRequested becomes 1, this rising edge is ignored.
OperatorAckRequested	Boolean	O	This output indicates the request for operator acknowledgment. The bit is set to "1" by the <i>SafetyConsumer</i>, when three conditions are met: 1) Too many communication errors were detected in the past, so the <i>SafetyConsumer</i> decided to switch to <i>fail-safe</i> substitute values. 2) Currently, no communication errors occur, and hence operator acknowledgment is possible. 3) Operator acknowledgment (rising edge at input OperatorAckConsumer) has not yet occurred. The bit is reset to "0" when a rising edge at OperatorAckConsumer is detected.
OperatorAckProvider	Boolean	O	This output indicates that an operator acknowledgment has taken place on the <i>SafetyProvider</i>. If operator acknowledgment at the <i>SafetyProvider</i> should be allowed, this output is connected to OperatorAckConsumer, see B.3.4 and B.3.5. NOTE 2 If the <i>ResponseSPDU</i> is checked with error, this output remains at its last value, see T24 in Table 35.
TestModeActivated	Boolean	O	The safety application program is expected to evaluate this output for determining whether the communication partner is in test mode or not. A value of "1" indicates that the communication partner (source of data) is in test mode, e.g. during commissioning. Data coming from a device in test mode may be used for testing but is not intended to be used to control safety-critical processes. A value of "0" represents the "normal" safety-related mode. The test mode enables the programmer and commissioner to validate the safety application using test data. NOTE 3 If the <i>ResponseSPDU</i> check results in an error and the <i>ErrorIntervalTimer</i> (see 6.3.4.4) is also not expired, TestModeActivated is reset, see state S17_Error in Table 34.
SafetyProviderID	UInt32	I	For dynamic systems, this input can be set to a non-zero value. In this case, the <i>SafetyConsumer</i> uses this variable instead of the <i>SPI</i> parameter <i>SafetyProviderIDConfigured</i>. This input is only read in the first cycle, or when a rising edge occurs at the input Enable. See also Table 26. If it is changed to "0", the value of <i>SPI</i> parameter <i>SafetyProviderIDConfigured</i> will be used (again). For static systems, this input is usually always kept at value "0".
SafetyBaseID	Guid	I	For dynamic systems, this input can be set to a non-zero value. In this case, the <i>SafetyConsumer</i> uses this variable instead of the <i>SPI</i> parameter <i>SafetyBaseIDConfigured</i>. This input is only read in the first cycle, or when a rising edge occurs at the input Enable. See also Table 26. If it is changed to "0", the <i>SPI</i> parameter <i>SafetyBaseIDConfigured</i> will become activated. For static systems, this input is usually always kept at value "0".