



BSI Standards Publication

Communication networks and systems for power utility automation

Part 7-6: Guideline for definition of Basic Application
Profiles (BAPs) using IEC 61850

National foreword

This Published Document is the UK implementation of IEC TR 61850-7-6:2024. It supersedes PD IEC TR 61850-7-6:2019, which is withdrawn.

The UK participation in its preparation was entrusted to Technical Committee PEL/57, Power systems management and associated information exchange.

A list of organizations represented on this committee can be obtained on request to its committee manager.

Contractual and legal considerations

This publication has been prepared in good faith, however no representation, warranty, assurance or undertaking (express or implied) is or will be made, and no responsibility or liability is or will be accepted by BSI in relation to the adequacy, accuracy, completeness or reasonableness of this publication. All and any such responsibility and liability is expressly disclaimed to the full extent permitted by the law.

This publication is provided as is, and is to be used at the recipient's own risk.

The recipient is advised to consider seeking professional guidance with respect to its use of this publication.

This publication is not intended to constitute a contract. Users are responsible for its correct application.

This publication is not to be regarded as a British Standard.

© The British Standards Institution 2024
Published by BSI Standards Limited 2024

ISBN 978 0 539 34711 1

ICS 33.200

Compliance with a Published Document cannot confer immunity from legal obligations.

This Published Document was published under the authority of the Standards Policy and Strategy Committee on 31 December 2024.

Amendments/corrigenda issued since publication

Date	Text affected

TECHNICAL REPORT



**Communication networks and systems for power utility automation –
Part 7-6: Guideline for definition of Basic Application Profiles (BAPs) using
IEC 61850**

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

ICS 33.200

ISBN 978-2-8327-0059-4

Warning! Make sure that you obtained this publication from an authorized distributor.

CONTENTS

FOREWORD.....	10
INTRODUCTION.....	12
1 Scope.....	13
1.1 General.....	13
1.2 Published versions of the standard and related namespace names.....	13
1.3 Identification of the code components.....	14
1.3.1 General.....	14
1.3.2 IEC 61850-6-100 XML namespace.....	14
1.3.3 IEC 61850-7-6 ASD example.....	14
1.4 Code Component distribution.....	15
1.4.1 General.....	15
1.4.2 SCL extension namespace code component.....	15
1.4.3 ASD example code component.....	15
2 Normative references.....	16
3 Terms and definitions.....	16
4 Methodology for profiling.....	18
4.1 General.....	18
4.2 IEC 61850 profiling concept.....	19
4.2.1 General.....	19
4.2.2 IEC 61850 profile definition.....	20
4.3 Basic Application Profiles (BAPs).....	21
4.4 Basic Application Interoperability Profiles (BAIOPs).....	23
4.5 Process from a use case to interoperability on SGAM function layer.....	25
4.6 Managing profiles.....	26
4.7 Implementation of BAPs in real projects.....	26
5 Requirements to describe machine-processable BAP.....	27
5.1 General purpose.....	27
5.2 High-level needs to address.....	28
5.2.1 Elements to be described in a SCL BAP.....	28
5.2.2 Use cases illustration.....	30
5.2.3 Requirements for the engineering process using BAP.....	31
5.2.4 Requirements regarding conformance of files.....	31
6 Methodology to describe BAP in SCL.....	31
6.1 SCL extensions.....	31
6.1.1 General.....	31
6.1.2 Cardinality.....	31
6.1.3 Application.....	33
6.1.4 AllocationRoleRef.....	34
6.1.5 Functional variants.....	35
6.1.6 Function Roles.....	38
6.1.7 Function Reference.....	40
6.1.8 Variable reference.....	43
6.1.9 Behavior description reference.....	44
6.1.10 Process resource reference.....	46
6.1.11 Power system relation reference.....	46
6.1.12 Function category reference.....	47

6.1.13	Interaction between the different elements of a BAP template.....	48
6.2	Engineering process	48
6.2.1	Introducing the .ASD file extension	48
6.2.2	Structure of an ASD.....	48
6.2.3	BAP templates: process and tools	49
6.2.4	Instantiating a BAP application template	52
6.2.5	Actions to perform during the instantiation process	52
6.2.6	Validation of a machine processable BAP	55
6.2.7	Requirements for CDT to deal with namespace conflicts between templates and instances	56
6.2.8	Use of SCL UUID elements for traceability.....	56
6.2.9	Editing an existing system with new or modified applications	57
Annex A (informative)	Example for BAP of distributed automation function "reverse blocking" using BAP template	59
A.1	Functional description	59
A.2	Description of use case and associated roles/actors	61
A.2.1	List of roles / actors	61
A.2.2	Use case	61
A.2.3	Sequence diagram of typical interactions	62
A.3	Logical architecture.....	63
A.4	Example of SCL BAP representation.....	63
A.5	Allocation variants (conditional)	65
A.6	Functional variants.....	65
A.6.1	Core functional variants	65
A.6.2	Noncore functional variants	65
A.7	Performance requirements	65
A.7.1	Functional related	65
A.7.2	Service related	66
A.8	Description of data model per actor.....	66
A.8.1	General	66
A.8.2	PTOC for blocked function (infeed bay)	66
A.8.3	PTOC for blocking function 1 to n (outflow bay(s))	68
A.8.4	Monitoring	69
A.9	Communication services	69
A.10	Device related requirements (conditional) – Test behavior	70
A.11	Naming rules	70
A.12	Capabilities for testing	70
Annex B (informative)	Example for BAP of "condition monitoring diagnosis functions of on-load tap changer" using BAP template	71
B.1	Functional description	71
B.2	Description of use case and associated roles/actors	73
B.2.1	List of roles / actors	73
B.2.2	Use case	74
B.2.3	Sequence diagram of typical interactions	75
B.3	Logical Architecture	78
B.3.1	Overview	78
B.3.2	Monitoring operation property	79
B.3.3	Monitoring operation counts.....	79
B.3.4	Monitoring contact abrasion.....	80

B.3.5	Monitoring LTC oil temperature and flow.....	80
B.3.6	Monitoring operation of oil filter unit.....	81
B.4	Allocation variants (conditional)	81
B.5	Functional variants.....	81
B.6	Performance requirements	81
B.6.1	Functional related	81
B.6.2	Service related	81
B.7	Description of data model per actor.....	82
B.7.1	General	82
B.7.2	SLTC	82
B.7.3	YLTC	85
B.7.4	TTRQ	85
B.7.5	TCTR.....	86
B.7.6	SIML.....	86
B.7.7	TTMP	87
B.7.8	KFIL	87
B.8	Communication services	87
B.9	Device related requirements (conditional)	87
B.10	Naming rules	87
B.11	Capabilities for testing	87
Annex C (informative) Example for BAP of protection function "line distance protection" using BAP template.....		88
C.1	Functional description	88
C.2	Description of use case and associated roles/actors	88
C.2.1	List of roles / actors	88
C.2.2	Use case	89
C.2.3	Sequence diagram of typical interactions	90
C.3	Logical architecture.....	91
C.4	Allocation variants (conditional)	91
C.5	Functional variants.....	91
C.5.1	Core functional variants	91
C.5.2	Noncore functional variants (different features for testing)	92
C.6	SCL BAP representation	92
C.7	Performance requirements	93
C.7.1	Functional related	93
C.7.2	Service related	93
C.8	Description of data model per actor.....	94
C.9	Communication services	96
C.9.1	General	96
C.9.2	Variant FA:	96
C.9.3	Variant FB:	97
C.10	Device related requirements (conditional) – Degraded operation behavior	97
C.11	Naming rules	98
C.12	Capabilites for testing	98
Annex D (informative) Example of BAiop for BAP reverse blocking (without process bus)...		99
D.1	General.....	99
D.2	Test description	99
D.2.1	General	99
D.2.2	Normal sequence of reverse blocking	99

D.2.3	Functional description of test environment	101
D.2.4	Test reverse blocking – role blocking (without output to process).....	102
D.2.5	Test reverse blocking – role blocked	104
Annex E (informative) Example of BAIOp for BAP of "condition monitoring diagnosis functions of on-load tap changer"		107
E.1	General.....	107
E.2	Test description	107
E.2.1	Overview	107
E.2.2	Sequence of monitoring the motor drive current value	108
Annex F (informative) Examples using SCL elements related to BAP		110
F.1	General prerequisites for use case understanding.....	110
F.2	BAP1: Creating dataflow between two BAP instances with the use of ProcessResource.....	110
F.2.1	Use case scope	110
F.2.2	Use case implementation.....	110
F.3	BAP2: Using PowerSystemRelation to link functions and applications to conducting equipments	113
F.3.1	Use case scope	113
F.3.2	Usecase Implementation.....	115
F.4	BAP3: Extending an existing substation with a new bay, integrating the new bay in the existing interlocking schema	116
F.4.1	Use case scope	116
F.4.2	Use case implementation.....	118
F.5	FV1: Circuit Breaker (CB) application single/three pole operated, with or without auto reclosing command input	120
F.5.1	Use case scope	120
F.5.2	Use case implementation.....	121
F.5.3	Defining and implementing functional variant actions	125
F.5.4	Automate the selection of the Functional Variant	128
F.6	FV2 SF6 variant of the CB application.....	129
F.6.1	Use case scope	129
F.6.2	Use case implementation.....	129
F.7	AR1: Definition of allocation roles to define physical allocation possibilities	131
F.7.1	Use case scope	131
F.7.2	Use case implementation.....	132
F.8	AR2: Editing of allocation roles with functional variants.....	133
F.8.1	Use case scope	133
F.8.2	Use case implementation.....	133
F.9	VE1: use of a variable element to edit datamodel naming elements	134
F.9.1	General	134
F.9.2	Use case scope	134
F.10	VE2: use of a variable element to edit common setting values (DirMod on multiple PTOC instances)	136
F.11	CA1: use of cardinality to instantiate function roles (busbar protection application)	138
F.11.1	Use case scope	138
F.11.2	Use case implementation.....	139
F.12	BD1: Textual behavior descriptions	140
F.13	BD2: Graphical behavior descriptions	141
F.14	BD3: IEC61131 format behavior descriptions	141

F.15 ASD file example	142
Annex G (informative) Concept Definition Tool (CDT) role description	143
Bibliography.....	144

Figure 1 – Stakeholders collaborate in user groups to create a common IOP profile	19
Figure 2 – Framework for profiling IEC 61850	20
Figure 3 – Aggregating BAPs	23
Figure 4 – Framework for testing a profile	24
Figure 5 – Relation between BAP and SGAM interoperability	25
Figure 6 – Device features covered by profiles depending on compatibility levels	26
Figure 7 – BAPs and BAIOs as building blocks for user/project specific implementation and testing	27
Figure 9 – Example of behavior description for distance protection BAP	29
Figure 10 – Interaction in a BAP template	48
Figure 11 – Structure of an ASD file	49
Figure 12 – Extracting an .ASD from an .SSD or .SCD	50
Figure 13 – creation and instantiation of an .ASD in a .SSD	51
Figure 14 – instantiation of an .ASD in an .SCD	51
Figure 15 – Selecting variants during .SSD instantiation	53
Figure 16 – Function C shared by Applications X and Y	53
Figure 17 – Overcurrent protection BAP template	54
Figure 18 – differential protection BAP template	54
Figure 19 – Merging of BAP template	54
Figure 20 – Edition workflow of an existing system	57
Figure A.1 – Behavior in the event of faults on an outflow bay	59
Figure A.2 – Behavior in the event of busbar faults	60
Figure A.3 – List of roles / actors reverse blocking	61
Figure A.4 – Use case reverse blocking	61
Figure A.5 – Sequence diagram reverse blocking	62
Figure A.6 – Logical architecture reverse blocking	63
Figure A.7 – SCL representation of BAP	64
Figure B.1 – Structure of LTC	71
Figure B.2 – Overview of system configuration of LTC condition monitoring	72
Figure B.3 – Typical system configuration of LTC condition monitoring system	72
Figure B.4 – Use cases	74
Figure B.5 – Sequence diagram for monitoring operation property	75
Figure B.6 – Sequence diagram for monitoring operation counts	76
Figure B.7 – Sequence diagram for monitoring contact abrasion	76
Figure B.8 – Sequence diagram for monitoring oil temperature and flow	77
Figure B.9 – Sequence diagram for monitoring operation of oil filter unit	77
Figure B.10 – Logical architecture	78
Figure B.11 – Logical architecture for monitoring operation property	79
Figure B.12 – Logical architecture for monitoring operation counts	79
Figure B.13 – Logical architecture for monitoring contact abrasion	80

Figure B.14 – Logical architecture for monitoring LTC oil temperature and flow	80
Figure B.15 – Logical architecture for monitoring operation of oil filter unit.....	81
Figure C.1 – Use case distance protection	89
Figure C.2 – Sequence diagram distance protection	90
Figure C.3 – Logical architecture distance protection	91
Figure C.4 – SCL BAP for distance protection.....	93
Figure D.1 – Normal sequence of application function reverse blocking	100
Figure D.2 – Functional test environment.....	102
Figure D.3 – Test of role "blocking"	102
Figure D.4 – Test of role "blocked".....	104
Figure E.1 – Sequence of monitoring the motor drive current value.....	108
Figure F.1 – Use case applications	110
Figure F.2 – Structure of SCL BAP for the use case applications	112
Figure F.3 – SSD template representing the use case applications	113
Figure F.4 – Illustration of ASD using PowerSystemRelation.....	114
Figure F.5 – Circuit breaker BAP template	115
Figure F.6 – Single Line Diagram of the extended substation.....	116
Figure F.7 – Interlocking schema of the coupling bay.....	116
Figure F.8 – Interlocking dataflow in the substation.....	117
Figure F.9 – Specific dataflow between one bay and the coupling bay	117
Figure F.10 – SCC illustration with engineering rights of the IEDs.....	118
Figure F.11 – Use of ProcessResource to create the dataflow of the new bay	119
Figure F.12 – Single pole CB with auto reclosing	120
Figure F.13 – Three pole CB with auto reclosing.....	120
Figure F.14 – Single pole CB without auto recloser	121
Figure F.15 – Three pole CB without auto recloser	121
Figure F.16 – Possibilities of creating Functional Variants	122
Figure F.17 – Option 1 actions	123
Figure F.18 – Option 2	124
Figure F.19 – Option 3	125
Figure F.20 – Actions associated to functional variants.....	126
Figure F.21 – Structure of the application	127
Figure F.22 – Creation and instantiation of an .ASD in a .SSD	128
Figure F.23 – instantiation of an .ASD in an .SCD.....	128
Figure F.24 – FunctionalVariantGroup example	129
Figure F.25 – FunctionalVariantGroup example	130
Figure F.26 – SCL structure with FunctionalVariantGroup.....	131
Figure F.27 – Allocation Variant use case	132
Figure F.28 – SCL structure for allocation variants.....	132
Figure F.29 – Description of the use case application	133
Figure F.30 – SCL structure of the application	134
Figure F.31 – Description of the application for variable element use case	135
Figure F.32 – Variable element prefix modification.....	136

Figure F.33 – Description of a second application for variable element use case	137
Figure F.34 – Variable element Direction modification	137
Figure F.35 – Description of the application for cardinality use case	138
Figure F.36 – BAP structure for cardinality use case	139
Figure F.37 – SCL structure for cardinality use case	140
Figure F.38 – Graphical representation of the SCL content	140
Figure F.39 – Graphical behavior description of the overcurrent application	141
Figure F.40 – IEC 61131 format behavior description of the overcurrent application	142
Table 1 – Reference between published versions of the standard and related namespace name	14
Table 2 – Attributes of the IEC 61850-6-100:2019C1 XML namespace	14
Table 3 – Attributes of the IEC 61850-7-6 ASD example	15
Table 4 – Attributes for cardinality and selector	33
Table 5 – Attributes of the Application element	34
Table 6 – Attributes of the AllocationRoleRef element	35
Table 7 – Attributes of the FunctionalVariant and FunctionalSubVariant elements	36
Table 8 – Attributes of the FunctionalVariantGroup element	37
Table 9 – Attributes of the FunctionalVariantRef element	38
Table 10 – Attributes of the FunctionRole element	39
Table 11 – Attributes of the FunctionRoleContent element	40
Table 12 – Attributes of the FunctionRef element	41
Table 13 – Attributes of the SignalRole element	42
Table 14 – Attributes of the LNodeInputRef element	43
Table 15 – Attributes of the LNodeOutputRef element	43
Table 16 – Attributes of the LNodeDataRef element	43
Table 17 – Attributes of the VariableRef element	44
Table 18 – Attributes of the BehaviorDescriptionRef element	45
Table 19 – Attributes of the <i>InputVarRef</i> and <i>OutputVarRef</i> element	45
Table 20 – Attributes of the ProcessResourceRef element	46
Table 21 – Attributes of the PowerSystemRelationRef element	47
Table 22 – Attributes of the FunctionCategoryRef element	48
Table A.1 – Selection of data attributes for PTOC of actor blocked	66
Table A.2 – Selection of data attributes for PTOC of actor blocking	68
Table A.3 – Selection of data attributes of PTOC for monitoring	69
Table B.1 – List of actors	73
Table B.2 – Selection of data attributes of SLTC	82
Table B.3 – Selection of data attributes of YLTC	85
Table B.4 – Selection of data attributes of TTRQ	85
Table B.5 – Selection of data attributes of TCTR	86
Table B.6 – Selection of data attributes of SIML	86
Table B.7 – Selection of data attributes of TTMP	87
Table B.8 – Selection of data attributes of KFIL	87

Table C.1 – Description of data model	94
Table C.2 – Services for variant FA	96
Table C.3 – Services for variant FB	97
Table C.4 – Degraded operation behaviors	98
Table D.1 – Description of normal operation of application function reverse blocking	100
Table D.2 – Description of sequence for test of role "blocking"	103
Table D.3 – Description of sequence for test of role "blocked"	105
Table E.1 – Description of the sequence of monitoring the motor drive current value	109

INTERNATIONAL ELECTROTECHNICAL COMMISSION

**COMMUNICATION NETWORKS AND SYSTEMS
FOR POWER UTILITY AUTOMATION –****Part 7-6: Guideline for definition of
Basic Application Profiles (BAPs) using IEC 61850****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 not 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 TR 61850 has been prepared by IEC technical committee 57: Power systems management and associated information exchange. It is a Technical Report.

This second edition cancels and replaces the first edition published in 2019. This edition constitutes a technical revision.

This edition includes the following significant technical changes with respect to the previous edition:

- a) New Clause 5 added to describe the way to express Basic Application Profile in SCL files;
- b) New Annex F and Annex G added to list specific use cases and roles of the Concept Definition Tool.

The text of this Technical Report is based on the following documents:

Draft TR	Report on voting
57/2710/DTR	57/2735/RVDTR

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 Technical Report 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.

A list of all parts in the IEC 61850 series, published under the general title *Communication networks and systems for power utility automation*, can be found on the IEC website.

NOTE The following print types are used:

Element and attribute types: in italic type.

This IEC technical report includes Code Components i.e. components that are intended to be directly processed by a computer. In this technical report, the code component includes all specifications of the XSD schema as well as all SCL examples.

The purchase of this IEC technical report carries a copyright license for the purchaser to sell software containing Code Components from this technical report to end users either directly or via distributors, subject to IEC software licensing conditions, which can be found at: <http://www.iec.ch/CCv1>.

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.

IMPORTANT – The "colour inside" logo on the cover page of this document indicates that it contains colours which are considered to be useful for the correct understanding of its contents. Users should therefore print this document using a colour printer.

INTRODUCTION

The IEC 61850 series of standards offers a broad basis for communication networks and systems in power utility automation. Due to its broad coverage of power utility automation applications, it is up to the standard's user (utility, vendor, system integrator, etc.) to pick and choose specific options from the standard in order to meet the requirements of the intended objective. As a consequence, implementations of IEC 61850 represent specific subsets of the standard.

In the context of standards, the term "profile" is commonly used to describe a subset of an entity (e.g. standard, model, rules).

Accordingly, an IEC 61850 standard profile contains a selection of data models (mandatory elements), applicable communication services and relevant engineering conventions (based on the Substation Configuration Language SCL defined in IEC 61850-6) for an application function of a specific use case in the domain of power utility automation.

Depending on the scope and objective different profile types can be distinguished:

- User profile – defined subset that is valid for a specific user / organization (e.g. utility)
- Product / device profile – implemented subset in a specific vendor product / device
- Domain profile – defined subset for a specific domain and relevant use cases (e.g. monitoring of substation)
- Application / function profile – subset covering a specific application or function (e.g. substation interlocking)

These profile types target the reduction of complexity and facilitation of interoperability for their specific scope and during engineering and device / substation lifetime. In order to achieve both these goals, a properly defined profile and appropriate implementations (processes, tools, products) that support the profile are required.

This Technical Report was first published in 2019, to cover the methodology for profiling Basic Application Profile. The text of the first edition of the Technical Report is based on the following documents:

Draft TR	Report on voting
57/1986/DTR	57/2034/RVDTR

A request for revision of the Technical Report was circulated in 2022, in order to add information about machine processable Basic Application Profiles, described in SCL.

Revision Request	Report on voting
57/2475/DC	57/2493/INF

This document related to the second edition of the Technical Report, prepared, and circulated as a CD in 2023. Clauses 1, 2, 3 and 4 are taken from the first edition of the Technical Report (IEC 61850-7-6:2019) and reproduced here without modifications. A new clause (5) has been added to describe the way to express the Basic Application Profile in SCL files. Specific use cases and roles of the Concept Definition Tool are added in Annex G in relation with this new clause.

COMMUNICATION NETWORKS AND SYSTEMS FOR POWER UTILITY AUTOMATION –

Part 7-6: Guideline for definition of Basic Application Profiles (BAPs) using IEC 61850

1 Scope

1.1 General

This part of IEC 61850, which is a Technical Report, is focused on building application / function profiles and specifies a methodology to define Basic Application Profiles (BAPs), in textual documents (edition 1, 2019) or in a machine processable SCL format (current edition). These Basic Application Profiles provide a framework for interoperable interaction within or between typical substation automation functions. BAPs are intended to define a subset of features of IEC 61850 in order to facilitate interoperability in a modular way in practical applications.

It is the intention of this document to provide a common and generic way to describe the functional behavior of a specific application function in the domain of power utility automation systems as a common denominator of various possible interpretations/implementations of using IEC 61850.

The guidelines in this document are based on the functional definitions of:

- IEC 61850-5, *Communication requirements for functions and device models*, which gives a comprehensive overview of all application functions needed in a state-of-the-art substation automation implementation.
- IEC TR 61850-7-500, *Basic information and communication structure – Use of logical nodes for modelling application functions and related concepts and guidelines for substations*, which illustrates and explains application functions for the substation/protection domain of Logical Nodes in modelling simple and complex functions, to improve common understanding in modelling and data exchange, and finally to lead to interoperable implementations.
- IEC TR 61850-90-3, *Using IEC 61850 for condition monitoring diagnosis and analysis*, which gives use cases and data modelling for condition monitoring diagnosis and analysis functions for substation and power grid facilities.
- IEC TR 61850-90-30, *IEC 61850 Function Modelling in SCL*, which describes extensions of the SCL Substation/Process Section allowing to create a comprehensive, IED and hardware independent specification of an IEC 61850 based power system.

This document does not describe the applications and respective implementation requirements; the focus is on their typical information exchange including data and communication services and engineering conventions.

1.2 Published versions of the standard and related namespace names

This technical report defines an SCL namespace extension. This allows the addition of these extensions to existing SCL files without requiring a new SCL schema release. They will be included in a future SCL release.

This technical report also defines ASD examples of files showing the possibilities offered by this new document. The content of the examples is defined in the code component 2024A1.

The SCL namespace extension is defined under IEC 61850-6-100 identifier which is a common extension between this document and TR IEC 61850-90-30:—¹.

Table 1 provides a reference between all published editions, amendments or corrigenda of this document and the full name of the namespace.

Table 1 – Reference between published versions of the standard and related namespace name

Edition	Publication date	Webstore	Namespace
Edition 1.0	2019-01	IEC 61850-7-6:2019	NA
Edition 2.0	2024	IEC 61850-7-6:2022	IEC 61850-6-100:2019C1

1.3 Identification of the code components

1.3.1 General

The namespace associated with this document is an XML schema (XSD) for an extension to the System Configuration Language (SCL) as defined in IEC 61850-6.

1.3.2 IEC 61850-6-100 XML namespace

The namespace associated with this document is an XML schema (XSD) for an extension to the System Configuration Language (SCL) as defined in IEC 61850-6. The parameters which identify the namespace are provided in Table 2:

Table 2 – Attributes of the IEC 61850-6-100:2019C1 XML namespace

Attribute	Content
Namespace nameplate	
Namespace Identifier (xmlns)	http://www.iec.ch/61850/2019/SCL/6-100
Version	2019
Revision	C
Release	1
XSD version header attribute	2019C1
Recommended reference name	eIEC61850-6-100
CodeComponentName	IEC_61850-6-100.XSD.2019C1.Full
Namespace dependencies	
includes	http://www.iec.ch/61850/2003/SCL version: 2007 revision: C release: 5

1.3.3 IEC 61850-7-6 ASD example

The example associated with this document is an SCL file (ASD) example based on IEC 61850-6-100 2019C1 and IEC 61850-6 2007C5. The parameters which identify the example are provided in Table 3:

¹ Under preparation. Stage at the time of publication: IEC/RDTR 61850-90-30:2024.

Table 3 – Attributes of the IEC 61850-7-6 ASD example

Attribute	Content
Example nameplate	
Version	2024
Revision	A
Release	1
Code Component Name	IEC_61850-7-6.ASD.2024A1.Full
Example dependencies	
includes	http://www.iec.ch/61850/2003/SCL version: 2007 revision: C release: 5
includes	http://www.iec.ch/61850/2019/SCL/6-100 version: 2019 revision: C release: 1

1.4 Code Component distribution

1.4.1 General

Each Code Component is a ZIP package containing the electronic representation of the Code Component itself, with a file describing the content of the package (IECManifest.xml).

The life cycle of a code component is not restricted to the life cycle of the related publication. The publication life cycle goes through two stages, Version (corresponding to an edition) and Revision (corresponding to an amendment). A third publication stage (Release) allows publication of Code Component in case of urgent fixes of InterOp Tissues, thus without need to publish an amendment.

Consequently, new releases of the Code Component may be released, which supersedes the previous release, and will be distributed through the IEC TC57 web site at:

<https://www.iec.ch/tc57/supportdocuments>

1.4.2 SCL extension namespace code component

The latest version/release of the document will be found by selecting the file for the code component with the highest value for VersionStateInfo, e.g. IEC_61850-6-100.XSD.{VersionStateInfo}.full.zip.

The code component associated to this TR is an XML schema file (XSD). It is available as a full version only. It is freely accessible on the IEC website for download at <http://www.iec.ch/tc57/supportdocuments>, but the usage remains under the licensing conditions.

In case of any differences between the downloadable code and the IEC pdf published content, the downloadable code component is the valid one; it may be subject to updates. See history files.

1.4.3 ASD example code component

The latest version/release of the ASD examples will be found by selecting the file for the code component with the highest value for VersionStateInfo, e.g. IEC_61850-7-6.ASD.{VersionStateInfo}.full.zip.

The code component associated to this TR is a list of examples packages as ZIP file used as informative. It is available as a full version only. It is freely accessible on the IEC website for download at <http://www.iec.ch/tc57/supportdocuments>, but the usage remains under the licensing conditions.

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 61131 (all parts), *Programmable controllers*

IEC 61850-5:2013, *Communication networks and systems for power utility automation – Part 5: Communication requirements for functions and device models*

IEC 61850-6:2009, *Communication networks and systems for power utility automation – Part 6: Configuration description language for communication in electrical substations related to IEDs*
IEC 61850-6:2009/AMD2:2024

IEC 61850-7-2, *Communication networks and systems for power utility automation – Part 7-2: Basic information and communication structure – Abstract communication service interface (ACSI)*

IEC TR 61850-90-30:2024, *Communication networks and systems for power utility automation – Part 90-30: Guideline for function modelling in SCL for substation automation*

IEC TR 62361-103:2018, *Power systems management and associated information exchange – Interoperability in the long term – Part 103: Standard profiling*

IEC TR 62390:2005, *Common automation device – Profile guideline*

3 Terms and definitions

For the purposes of this document, the following terms and definitions 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>

3.1

Basic Application Profile

BAP

user/user group agreed-upon selection and interpretation of relevant parts of the applicable standards and specifications, intended to be used as building blocks for interoperable user/project specifications

Note 1 to entry: BAPs shall not have options for a specific variant of an application; all selected criteria are required to facilitate interoperability. For implementation in projects, BAPs might be extended or refined to meet the user specific requirements.

[SOURCE: CEN-CENELEC-ETSI SG-CG Report on Interoperability CEN_9762_CLC_9624 – Clause 12.1, Terms and definitions]

3.2

Basic Application Interoperability Profile

BAIOP

interoperability test for BAPs defined by test sequences and test cases

[SOURCE: CEN-CENELEC-ETSI SG-CG Report on Interoperability CEN_9762_CLC_9624 – Clause 12.1, Terms and definitions]

3.3

conformance

accordance of the implementation of a product, process or service with all specified requirements or standards

Note 1 to entry: Additional features to those in the requirements / standards might be included.

Note 2 to entry: All features of the standard/specification are implemented and in accordance, but some additional features are not covered by the standard/specification.

[SOURCE: CEN-CENELEC-ETSI SG-CG Report on Interoperability CEN_9762_CLC_9624 – Clause 12.1, Terms and definitions]

3.4

conformance test

check of data flow on communication channels in accordance with the standard conditions concerning access organization, formats and bit sequences, time synchronization, timing, signal form and level and reaction to errors

[SOURCE: IEC 61850-4:2011, 3.17, modified (removal of Note to entry)]

3.5

interoperability

ability of two or more IEDs from the same vendor, or different vendors, to exchange and use information for correct execution of the specified functions

[SOURCE: IEC 61850-2:2003, 2.85]

3.6

interoperability testing

testing performed to verify that communicating entities within a system are interoperable, i.e. they are able to exchange information in a semantically and syntactic correct way

Note 1 to entry: During interoperability testing, entities are tested against peer entities known to be correct (profiles).

[SOURCE: CEN-CENELEC-ETSI SG-CG Report on Interoperability CEN_9762_CLC_9624 – Clause 12.1, Terms and definitions]

3.7

profile

agreed-upon subset derived from a specification

Note 1 to entry: The content of the subset is user dependent.

Note 2 to entry: A common profile is required for achieving interoperability especially in those cases when a specification could have more than one interpretation and there are probably many optional features.

[SOURCE: IEC TR 62361-103:2018, 3.11]

3.8

SGAM

Smart Grid Architecture Model, the 3D-Model for Smart Grid mappings

high level conceptual model of the Smart Grid developed by the CENELEC M/490 Reference Architecture working group describing main actors of the Smart Grid and their main interactions

[SOURCE: CEN-CENELEC-ETSI SG-CG Report on Interoperability CEN_9762_CLC_9624 – Clause 12.1, Terms and definitions]

4 Methodology for profiling

4.1 General

In general, profiling within a standard and between standards and specifications helps to both facilitate interoperability and meet expectations of different projects where these will be implemented.

Out of this broad basis of the definitions of the IEC 61850 standard series, specific subsets (selected by vendors) are implemented in products and systems.

IEC 61850 applications can also differ dependent on user type, region and philosophy. Stakeholders officially request guidelines and tools to facilitate interoperability in projects and therefore the challenge is to find a common concept/guideline to both facilitate interoperability and meet expectations of different projects.

Flexibility of standards which were created to meet the requirements of the industry of the global world, containing many options, contradicts with the goal of interoperability of functions. If a function can be implemented in more than one way in products, the risk that those products will be unable to perform the function in an interoperable way is high. To facilitate the goal of interoperability in addition to the mandatory elements the mandatory inclusion of selected optional elements offered by the standard may be defined. That limitation of options offered by a standard is done by defining commonly agreed subsets (profiles), which might be interpreted as disadvantageous inflexibility against specific user requirements.

To facilitate the goal of interoperability, a common understanding and interpretation of the related standard and the identical use of functional elements for required layers to fulfil application functions is necessary. This can be achieved by defining profiles. They could be best provided by User Groups in the domain of substation automation. A User Group consists of interested parties, e.g. utilities, vendors, certification bodies, test labs, system integrators and regulators, see Figure 1.

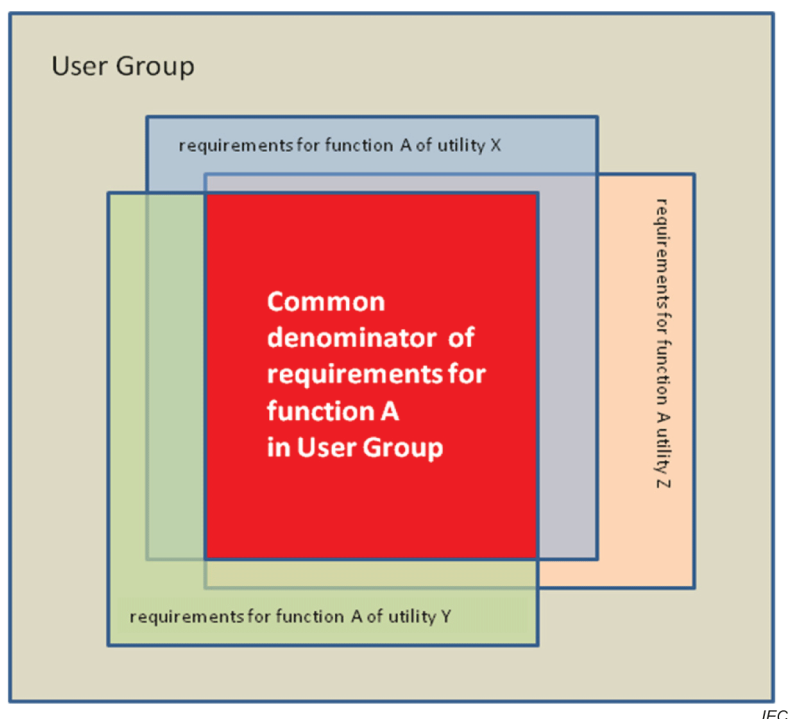


Figure 1 – Stakeholders collaborate in user groups to create a common IOP profile

IEC TR 62361-103:2018, 4.1, defines a common concept and framework for the process of profiling within the IEC (CIM and IEC 61850).

4.2 IEC 61850 profiling concept

4.2.1 General

The primary goal of the IEC 61850 series is to facilitate interoperability in a modular way between subsystems and components in order to enable more or less complex system functions. Therefore, the IEC 61850 series covers specifications for functional and non-functional requirements, information and communication aspects for devices and systems as well as device and system engineering. With that the IEC 61850 series offers a broad basis for communication networks and systems in power utility automation.

Due to its broad coverage of power utility automation functions and applications, it is up to the user of the standard (utility, vendor, system integrator etc.) to pick and choose specific options offered by the standard in order to meet the requirements according to their intended project objectives. As a consequence, implementations of IEC 61850 represent specific subsets of the standard instead of covering it as a whole.

IEC 61850 profiles express (interoperability) requirements of actors' roles in a specific business context to be fulfilled by devices and systems.

4.2.2 IEC 61850 profile definition

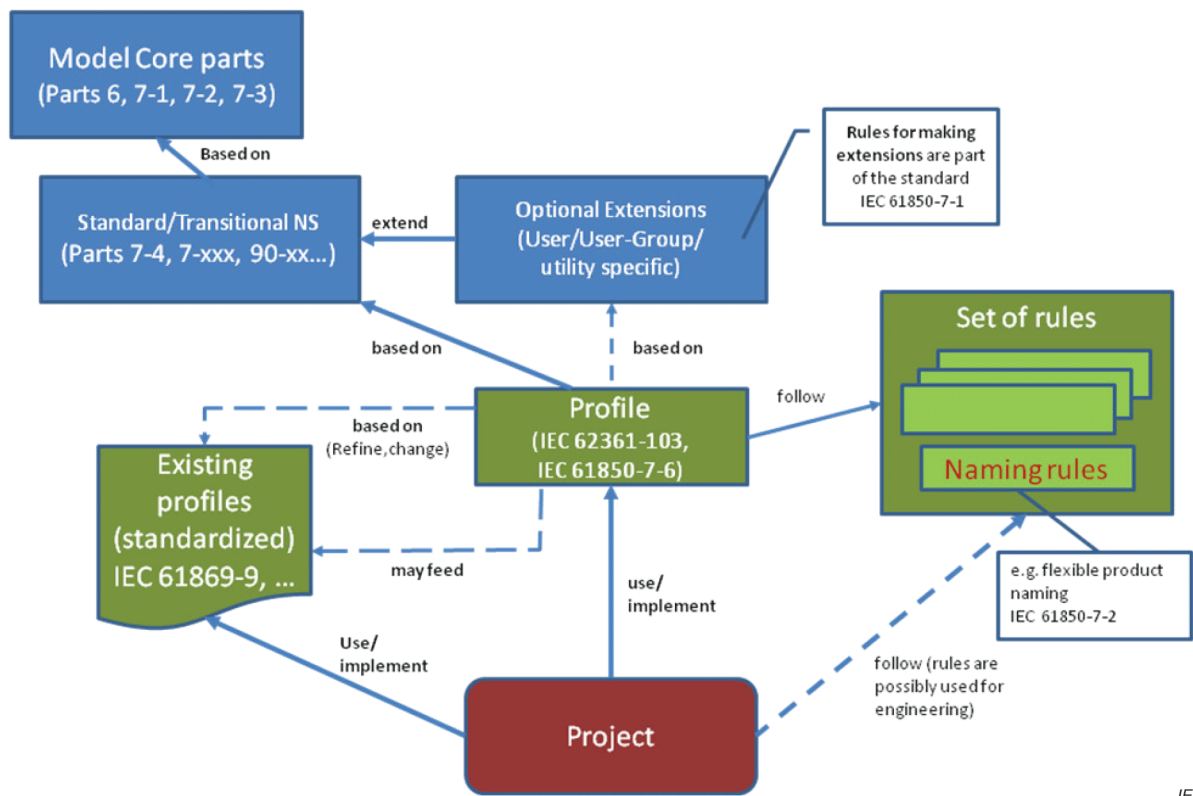


Figure 2 – Framework for profiling IEC 61850

Figure 2 shows the main relationship between the different elements constituting the profiling activity of IEC 61850.

An IEC 61850 standard profile may contain a selection of data models (with mandatory and optional elements) and communication services applicable for a specific role within a detailed use case. The data models and communication services are all based on the same core parts as described in Figure 2.

It is acknowledged that a profile cannot be less demanding than the standard itself, i.e. a field or item said to be mandatory (or forbidden) by the standard remains mandatory (or forbidden) in a profile. The standard represents a sort of minimum base line.

Furthermore, a profile may define specific features (e.g. pay load, specific device types) and procedures (e.g. programmable logics, message sequences). Subclause 4.3 provides possible content of a profile.

Depending on the scope and objective different profile types can be distinguished:

- User profile – defined subset that is valid for a specific user / community of users (e.g. utility)
- Domain profile – defined subset for a specific domain and relevant use cases (e.g. asset management)
- Basic Application Profile (BAP) – standardized subset² defining an atomic application function (e.g. reverse blocking)

² Most of the profile types above might follow a standardization process, in order to collect the widest range of comments and guarantee their stability for a defined period.

- Application profile – profile covering a specific application mostly based by aggregating BAPs (e.g. busbar protection)
- Device profile – profile covering a typical IED functionality (e.g. Merging Unit, IEC 61869-9)
- Product profile – implemented subset in a specific vendor product

All these profile types target reducing complexity and facilitate interoperability in a modular way for their specific scope. In order to achieve these goals, both a properly defined profile and appropriate implementations (processes, tools, products) that support the profile are required.

4.3 Basic Application Profiles (BAPs)

A Basic Application Profile (BAP) is based on system/subsystem specific basic application functions descriptions. The term "basic" means here that an elementary application function/subfunction is the chosen context for defining the profile. The level of what is perceived as elementary is application dependant and may include for example many Logical Node (LN) instances of many LN classes, when using IEC 61850.

A BAP is a user/user group agreed-upon selection and interpretation of relevant parts of the applicable standards and specifications and is intended to be used as building blocks for interoperable user/project specifications.

The key ideas of BAPs are:

- BAPs are elements in a modular framework for specific application systems/subsystems.
- Combinations of different BAPs can be used in real projects as building blocks.
- Project specific refinement additional to the BAP might be necessary to meet specific requirements for implementation in projects. These additional requirements should be frequently fed back into the User Group and might lead to a new or revised BAP based on user experiences and group decisions.
- BAPs represent a user agreed common denominator of a recommended implementation or a proven best practice implementation of an application function in the domain of substation automation but is not aimed to cover all possible implementation options.
- The user may specify functional variants of a BAP; once a functional variant is selected, all its elements will be implemented as mandatory, in the interest of interoperability.
- IEDs might contain more options as requested by BAPs to achieve more flexibility to satisfy the needs of the global market.
- BAPs are built on the basis of international standards and also might have an influence in the further development of standards by possible feedback and implementation of lessons learned.

BAPs include (BAP template):

Functional description

- Verbal description of the function supported by figures to explain the required functional behavior of the application function.

Description of the use case and the associated roles/actors

- List of roles /actors,
- Sequence diagram showing the typical interactions between the associated roles/actors.

Logical architecture

- Description of the application function by using LNs and description of the interaction between those LNs using attributes defined inside the LNs.

Allocation variants (conditional)

- This BAP section might be used if the allocation of sub-functions (LNs) into different physical devices can be done in different ways e.g. allocation of subfunctions (LNs) in configuration with or without process bus.

Functional variants

- To avoid multiple BAP definitions for the same application function caused by slightly different behavior this clause can be used to define so called implementation variants for the same application function the use of attributes needed for these functional variants has to be shown in the description of the data model e.g. with or without monitoring of status signals.

Performance requirements

- Functional related e.g. required behavior in case of loss of communication,
- Service related e.g. timing constraints, required reaction time, etc.,
- It is recommended to use the performance classes defined in IEC 61850-5:2013, Clause 11.

Description of data model per actor

- Semantic model,
In case functional variants are used, this can be shown in the semantic data model by using columns for each functional variant,
- Syntactic model.

Communication services

- Description of the IEC 61850 services used to perform the required behavior of the application function.

Device related requirements (conditional)

- This BAP section might be used if a device containing sub-functions of the BAP (LN's) requires special network or device configuration to achieve the required variant of the application function,
- Configuration capabilities
In case that a specific configuration of the device on the station network is needed to fulfil the achieved required variant of the application function is needed (e.g. works on process bus only).
- Communication capabilities
In case specific configuration of communication services or network configuration is needed for the device to achieve the required variant of the application function.

Naming rules

- There might be requirements for constraining names in the profile which should be defined in this BAP section,
- Any restrictions in naming might cause risk of limited cohabitation with other profiles in one device.

Capabilities for testing

- A taskforce within the IEC is currently working IEC 61850-10-3 which defines standardized capabilities of application functions and rules needed for interoperability testing.
e.g. the standardized use of the data attributes "Mod" and "Beh" as well the standardized use of the quality attribute "test" which might also lead to a test BAP.

The objective of a BAP is to reduce complexity, clarify vague or ambiguous specifications and so it aims to facilitate interoperability in a modular way.

The profiling concept selected for IEC 61850 is a classical modular bottom-up approach to facilitate interoperability in a modular way between functions/subfunctions located in IEDs forming a substation automation environment. IEC 61850-5 shows how the domain of substation automation can be decomposed into functions/sub functions.

That concept, as presented in Figure 3, is recommended by the Interoperability Report worked out by the Smart Grid Coordination Group of CENELEC (CEN-CENELEC-ETSI SG-CG Report on Interoperability CEN_9762_CLC_9624).

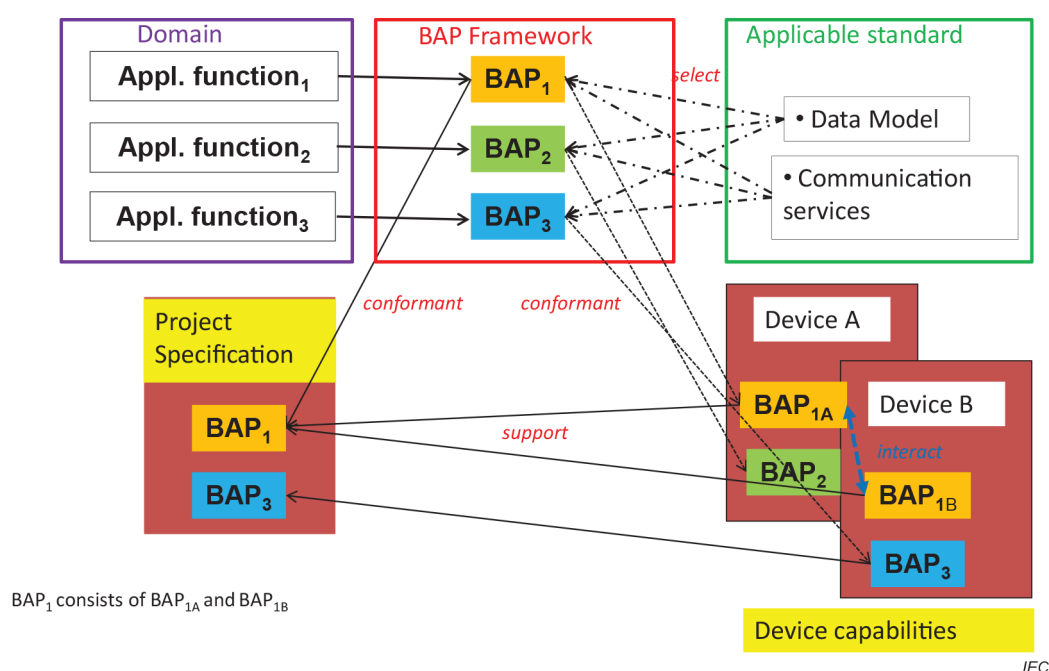


Figure 3 – Aggregating BAPs

Examples of BAPs are available in Annex A, Annex B and Annex C.

4.4 Basic Application Interoperability Profiles (BAIOPs)

The conformance test defined in IEC 61850-10 is a prerequisite for functional testing which is needed to facilitate interoperability.

The conformance test simply verifies if the implementation of IEC 61850 (described in PICS, PIXIT, MICS, SICS, SCL files etc.) is conformant to the definitions of the standard.

IEC 61850-10 does not contain any requirement for testing of functional interoperability by a device or between devices.

No testing to facilitate interoperability on functional level is performed in a conformance test.

Functional testing needs different test sequences and test cases compared to a conformance test.

The test cases defined in a BAIOP focus on the use case of the BAP to verify the functional interaction of LNs based on the selected attributes of the data model and the selected services for communication.

BAPs are defined by User-Groups representing the asset owner or a collective of asset owners e.g. typically utilities with a common interest. BAIOPs are normally defined and performed by different stakeholders like test labs, certification bodies etc. which can add value by contributing their special experience and knowledge about testing to facilitate interoperability.

For interoperability testing a BAP can be amended by:

- relevant engineering related requirements (e.g. concept of functional naming),
- device configuration,
- test configuration with communication infrastructure (topology),
- BAP related functional test sequence diagrams and test cases,
- specific capability descriptions (e.g. PICS, PIXIT, MICS in the case of IEC 61850),
- engineering framework for data modelling (instances) and communication infrastructure (topology, communication service mapping).

IEC TR 62361-103:2018, 4.1, also defines a common concept and framework for the process of testing profiles within the IEC (CIM and IEC 61850). The common concept and framework of testing is shown in Figure 4.

The tests defined in a BAIOP prove that the implementation of relevant parts of IEC 61850 in a device is both conformant to the corresponding BAP and facilitates interoperability in a modular way among devices from different vendors.

An organization which is qualified to perform interoperability testing (with a possible certification) for BAPs/BAIOPs still needs to be defined.

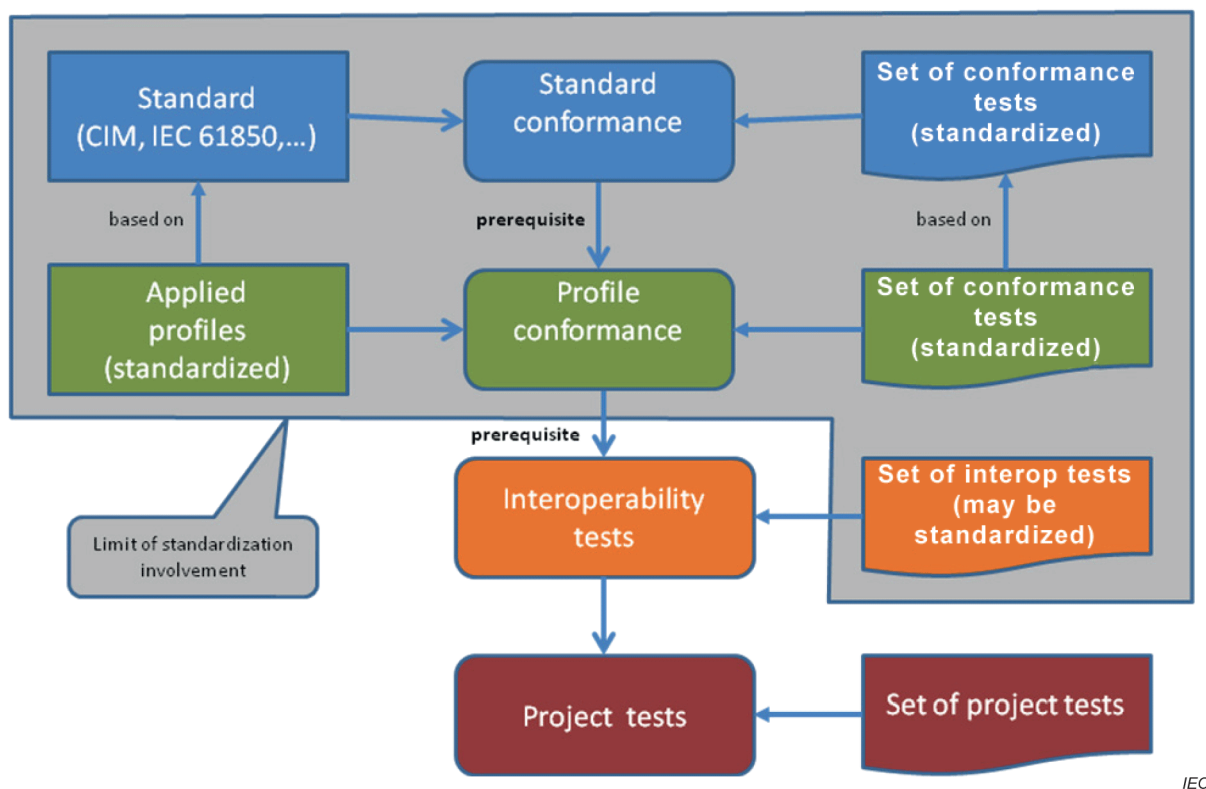


Figure 4 – Framework for testing a profile

The definition and common use of BAPs and BAIOPS should lead to a win-win situation for all stakeholders involved in the power utility automation domain in general, e.g.:

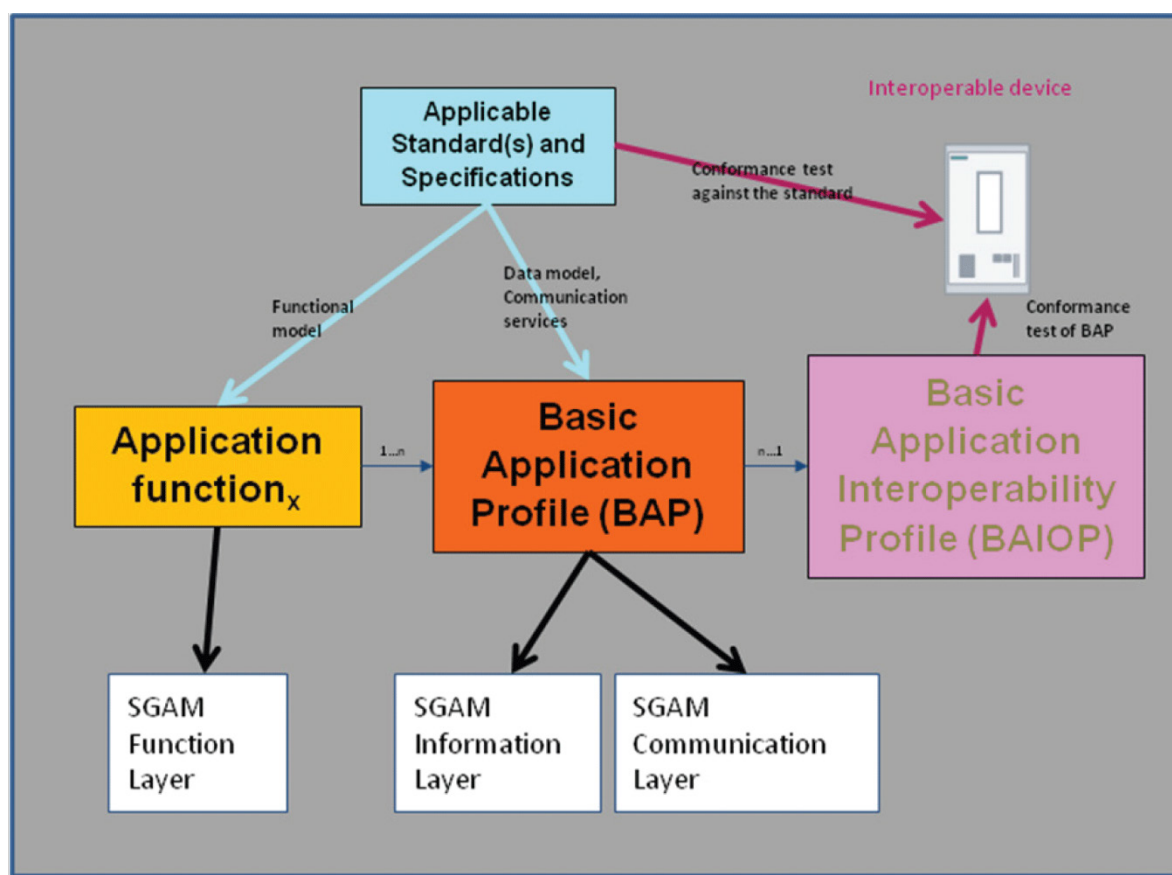
The benefit for users (e.g. asset owners) and user groups is the chance to harmonize the various company specific application function variants to a common denominator / best practice implementation for each basic application function. This reduces the risk of interoperability problems caused by products/systems as these may be selected from standardized BAP frameworks and tested according to BAIOPs.

The benefit for manufacturers which will use standardized BAPs in their products is the reduction of project specific or customer specific implementation variants of application functions and therefore reduce product complexity, development costs and parameterization efforts. BAIOPs can be used for internal tests before the product will be placed on the market. Two examples of BAIOPs are available in Annex D and Annex E.

The benefit for certification bodies / test labs is the ability to perform interoperability tests based on BAIOPs and create out of the need for interoperability a new business case. The benefit for system integrators is that they can specifically select products conformant with BAPs and tested according to BAIOPs. This should reduce the efforts for integration of subsystems or devices.

4.5 Process from a use case to interoperability on SGAM function layer

Figure 5 illustrates the process from a use case to facilitate interoperability on SGAM function layer in a modular way by using BAPs and BAIOPs.



IEC

Figure 5 – Relation between BAP and SGAM interoperability

To achieve the required level of compatibility (from coexistent up to interchangeability), it might be useful to consider the device compatibility levels derived from IEC TR 62390:2005 (IEC Technical Committee 65: Industrial-process measurement, control and automation) which are shown in Figure 6. With an increased required level of compatibility towards IOP, the necessary device features that need to be covered by profiles are also increasing.

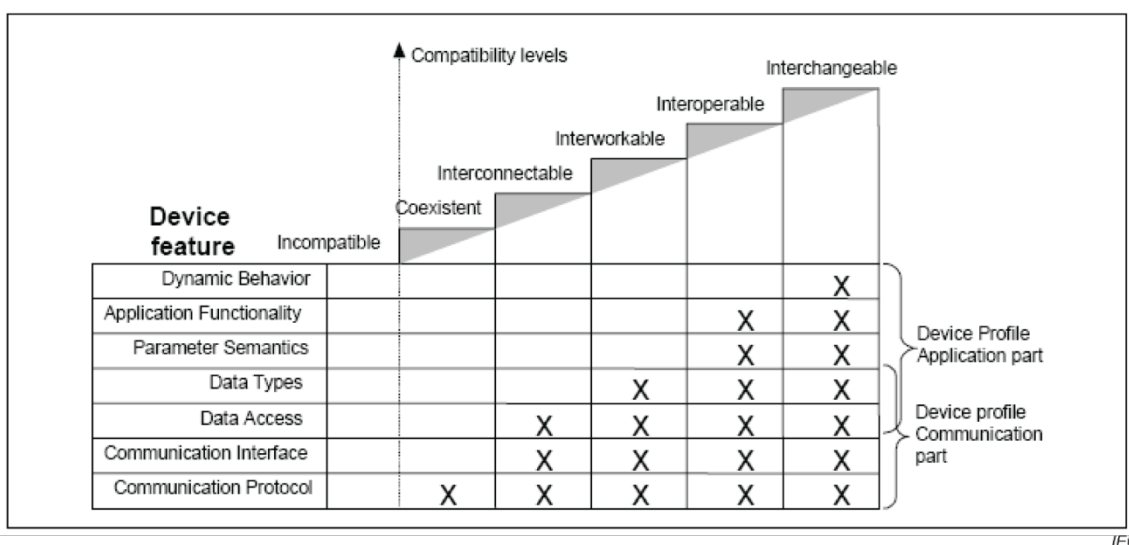


Figure 6 – Device features covered by profiles depending on compatibility levels

4.6 Managing profiles

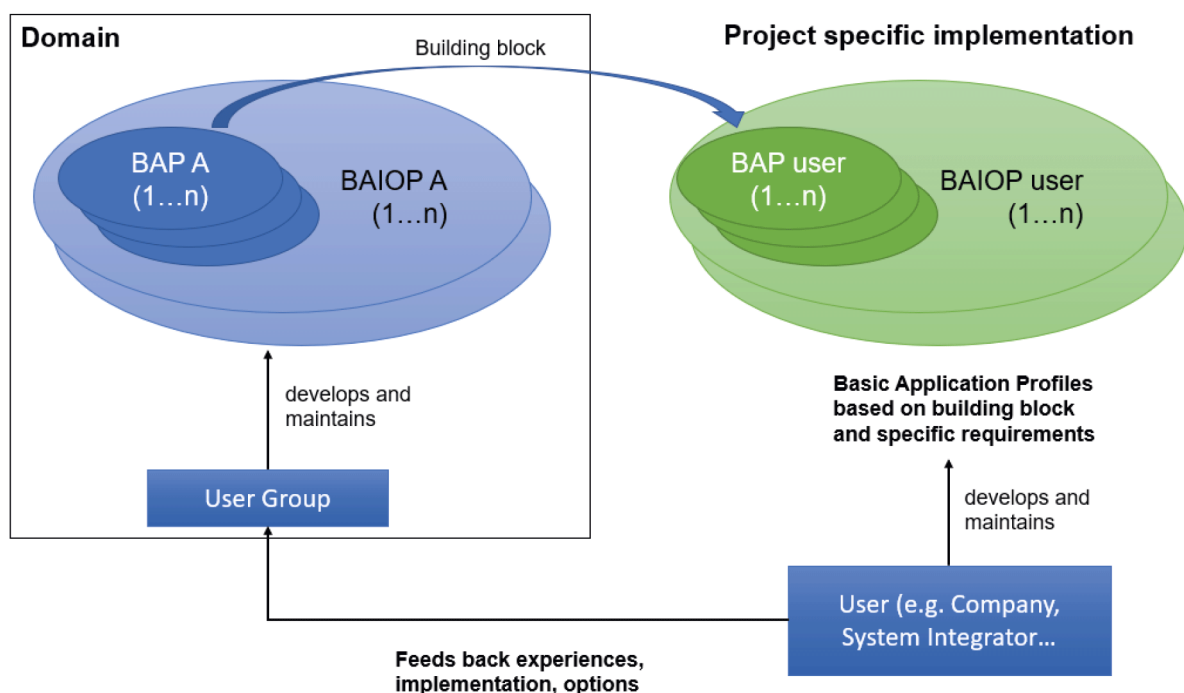
It is important that a profiling management is in place in order to ensure that profiles are applied and understood in the same way by all affected stakeholders, and to avoid that diverging profiles for the same purpose will be developed and applied in parallel. This mainly includes:

- the responsibilities and roles of the different actors which are involved to create and manage profiles,
- change management and versioning control of updated profiles,
- communication of changes to affected stakeholders.

It is recommended that user groups (e.g. UCA or ENTSO-E) take ownership of creating and managing profiles. This also means that lessons learned have interest to be fed back by users of the profiles to the corresponding user groups to enable them to improve their profiles according to predefined cycles. Therefore, clear contact information can be attached to profiles. The user group is also responsible for the change management and versioning control of updated profiles, and to communicate changes to the affected stakeholders and other user groups in an adequate way, e.g. by newsletters or information on user group websites.

4.7 Implementation of BAPs in real projects

BAPs and BAIOPs are elements in a modular framework for specific application systems/subsystems and can be used in combination as building blocks in real projects. The user involved in the project (e.g. a company or system integrator) is responsible for developing and maintaining project specific application profiles based on these building blocks, but specific refinement still might be necessary to meet the project requirements. Figure 7 illustrates this process.



IEC

Figure 7 – BAPs and BAIOPs as building blocks for user/project specific implementation and testing

5 Requirements to describe machine-processable BAP

5.1 General purpose

Clause 4 specifies a methodology to define Basic Application profiles (BAPs), with "office" tools as spreadsheets and text documents. These office tools are easy to use, and provide tables or schemas to describe the BAP as shown in Figure 8.

However, BAP, as specification of an application, should be able to be integrated in the SCL engineering cycle, with the existing configuration tools. This leads to describe a SCL BAP format which could be instantiated in SSD and SCD files.

It might also be useful to use machine-processable BAP (as UML for example) to get more advanced features: facilitate BAP evolutions management, creation of "packages" of several BAP. Auto-generation of BAP "text" description or SCL BAP template should be done from the machine-processable BAP format.

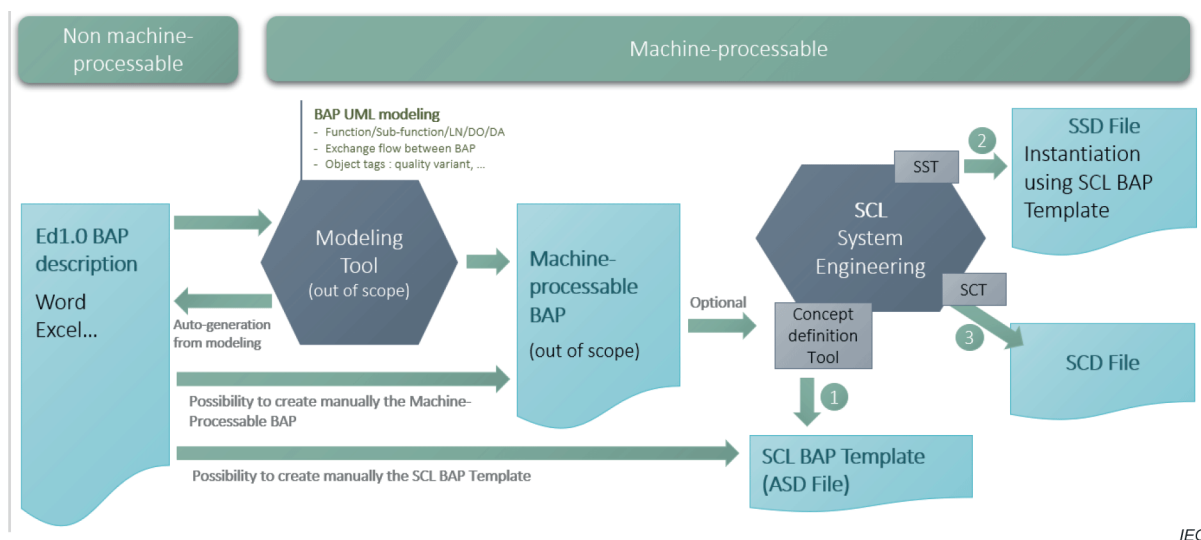


Figure 8 – Use of BAP in lifecycle of projects

The way to describe BAP in a modelling tool as UML, in order to translate BAP in a machine-processable format, is out of scope of this document. The machine-processable BAP may contain all element in the textual BAP, expressed in machine processable format like VSD, PLC Open.

5.2 High-level needs to address

5.2.1 Elements to be described in a SCL BAP

Subclause 4.3, Basic Application Profiles (BAPs), expresses what elements are included in BAPs based on textual documentation. It is accepted that not all these elements could be expressed in an SCL file representing BAP. The minimum needs to be covered by an SCL file are described in the following points:

- The functional description of the application, which explains the behavior of the application, can be expressed in SCL. Users should be able to understand the behavior of the application from the SCL file, and to identify which functions (as described in IEC TR 61850-90-30) are expected for the IEDs to perform the application. This could be done by the ability to link document to the application, or to list the functions needed to perform the application.
- The role or actors associated to the application can be described in order to ensure a clear understanding of the SCL model of the application, regarding the functional description; and to add capability to manage group of data in a coherent way. The role in SCL could apply to coherent sets regarding functional elements, signals or for instance physical allocation of the application.
- The logical architecture of the application is expected in the SCL BAP, to identify the data model (LNs used and inputs signals of LNs) required for the application.
- The functional variants or sub-variants of an application are used to describe several implementation variants in a single application. In order to avoid having multiple SCL files describing theses variants of an application, it is required to be able to express the functional variants in SCL. More particularly, the impact of each variant of an application on other elements of the SCL BAP should be identifiable. When the application is instantiated in a specific project, only one functional variant of the application should remain.
- The allocation role allows the expression of the variability of allocation of functions into physical devices, for a single application. In the SCL BAP, the allocation variant allows the user to specify the IEDs allocation of functions if needed. This allocation has to be coherent when the application is instantiated in a specific project.
- The communication services used to perform the application should be expressed in SCL BAP, in a consistent way with the definition of functions in IEC TR 61850-90-30.

- The naming rules could be expressed in SCL BAP by the use of variable elements, in order to automatize the naming of the application objects during the instantiation stage.

Regarding the expression of performance requirements, device, or capability for testing, which are part of a basic application template based on textual description, it is not the intention to integrate this need in an SCL file. However, if the introduced elements can be applied to describe these requirements, this can be seen as an opportunity. For instance, behavior descriptions (as introduced by IEC TR 61850-90-30:2024, 6.1.8) allow the creation of references between SCL BAP and non-SCL descriptions of textual, graphical or IEC 61131 type. An example is shown below, linking a graphic to an application explaining the use cases of the application. SCL elements shown in Figure 9 are described later in the document.

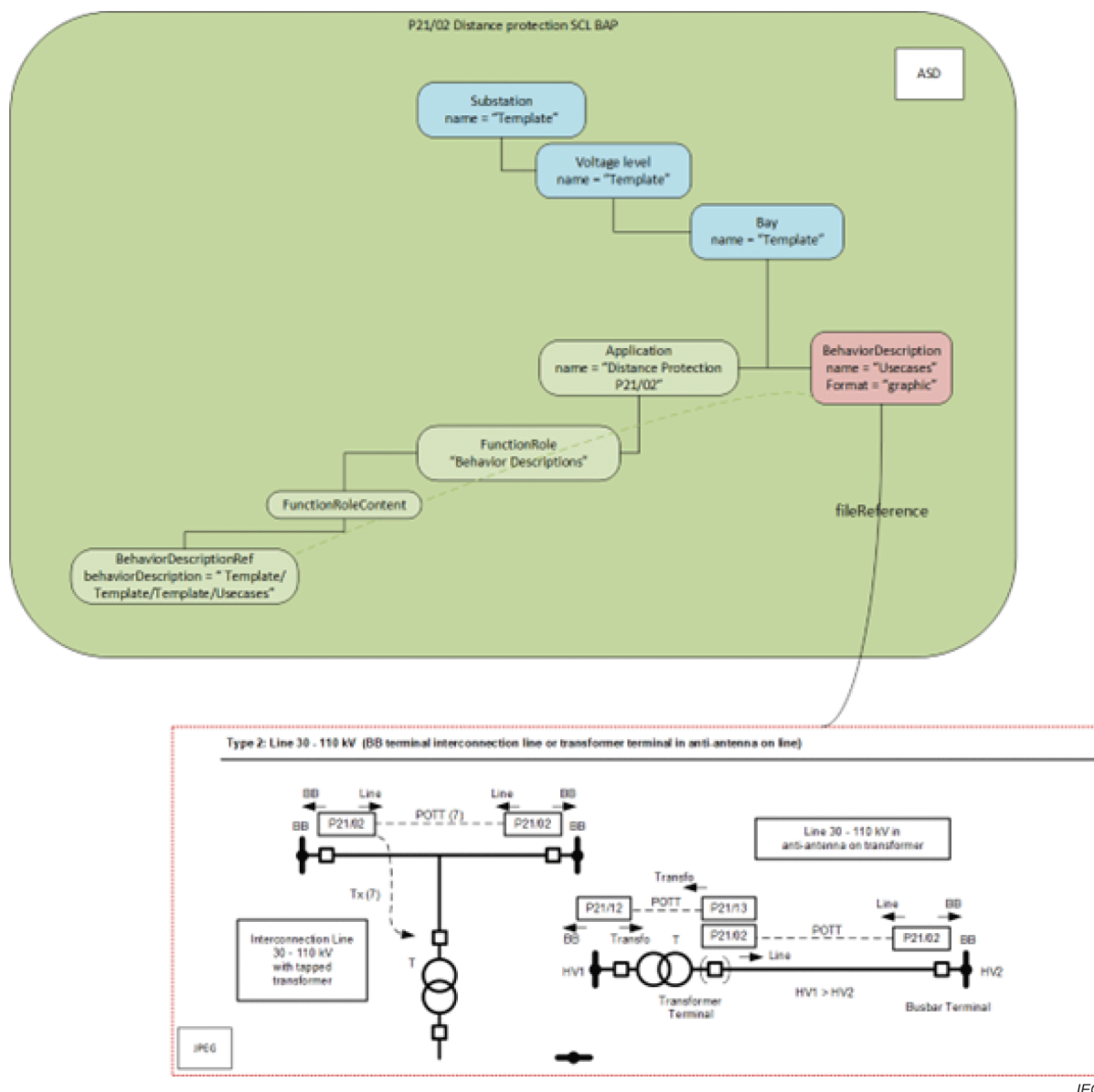


Figure 9 – Example of behavior description for distance protection BAP

Moreover, opportunities can be offered by the description of BAP in a structured format as SCL. For instance, variable elements (as defined in IEC 61850-90-30) could be defined to handle in an easy way the management of a set of related data of the application.

5.2.2 Use cases illustration

The goal of the use cases is to describe the needs from the users that will lead to the solutions described later in this document. These needs are related to the description of an application. Use cases are illustrated in Annex F.

Needs can be divided into six categories, depending on the context:

- 1) Using *PowerSystemRelation* elements (as defined in IEC TR 61850-90-30) to link functions to conducting equipment and using *ProcessResource* elements (as defined in IEC TR 61850-90-30) to create dataflow between BAP instances.
 - a) (BAP1) Being able to describe the template of several basic applications interacting with each other. This use case shows the use of a *ProcessResource* element to automate dataflow configuration between BAP instances. In this use case it shows the links between a current transformer application, overcurrent protection application and a circuit breaker application.
 - b) (BAP2) Being able to link participating functions of a BAP instance to a conducting equipment, using *PowerSystemRelation* elements. In this use case the BAP template of a circuit breaker is instantiated and linked to a circuit breaker conducting equipment.
 - c) (BAP3) Keeping *ProcessResource* elements in the SCD file to automate dataflow creation between existing applications and new applications added in a later stage. In this usecase an existing substation is extended with an additional bay, which needs to be integrated in the existing interlocking scheme.
- 2) Being able to design different variants of an application template in a centralized optimal way.
 - a) (FV1) CB application single/three pole operated, with or without autoreclosing command input.
 - b) (FV2) SF6 variant of the CB application: An SF6 variant of a circuit breaker application will include an additional SF6 gas insulation supervision alarm. This variant is considered independent from other variants like single/three pole operation of a circuit breaker.
- 3) Being able to document, in the context of an application, different physical allocation roles that host functions of an application.
 - a) (AR1) This first use case shows the use of allocation roles, with a CT application: the TCTR instance may be allocated inside the SAMU or the bay controller, depending on the implementation context.
 - b) (AR2) This second use case shows the use of allocation roles combined with functional variant. The interconnection distance protection functions are allocated, depending on the voltage level, in a P1 (150 kV and up) or combined P1/Bay controller IED (below 150 kV).
- 4) Documenting variables that are part of an application, and that can be controlled/changed depending on which variant of an application is required.
 - a) (VE1) A variable from a busbar switch application, which allows to set the prefix of the LNodes with the equipment name of the busbar switch. The value can differ between SR1 and SR2.
 - b) (VE2) Setting the direction of a directional overcurrent function from forward to backward depending on the application (towards an interconnection line, towards a transformer).
- 5) Adapting BAP instances dynamically depending on the system design it is applied to.
 - a) (CA1) Adapting applications to the size of the system with a busbar protection application instantiating the CT function role (TCTR LNode) depending on the number of bays (1 for each bay)
- 6) Documenting the required behavior of (a part of) an application, as a part of specification and/or testing.
 - a) Being able to link a description of the behavior of (a part of) the application to the BAP in multiple formats.

- i) (BD1) Textual behavior descriptions
- ii) (BD2) Graphical behavior descriptions
- iii) (BD3) IEC61131 format behavior descriptions, including the link between the logic variables and the *SourceRef/DOS* of a function, allowing to use this logic for implementation and/or simulation purposes. This usecase demonstrates how the expected behavior of a full application is described as a IEC 61131 logic. See IEC TR 61850-90-30 for usecases where only a part of the application's specified behavior is described as IEC 61131 logic.

5.2.3 Requirements for the engineering process using BAP

Users are able to create and manage BAP templates and instantiate them in a system specification. The system specification containing BAP instance provides the following capabilities:

- a) Trace SCL BAP templates used to build the system specification, and which variants of each application has been chosen,
- b) Automate the roll-out of SCL BAP template upgrades,
- c) Automate the data-flow configuration between BAP instances,
- d) Allowing multiple BAP instances to share the same element.

5.2.4 Requirements regarding conformance of files

Engineering tools might be able to verify the conformance of SCL files (ICD, SSD) against BAP SCL template. More particularly, it would be of interest to be possible to identify a way to detect differences between the instantiation of SCL BAP and the initial template.

6 Methodology to describe BAP in SCL

6.1 SCL extensions

6.1.1 General

New objects are needed to describe BAP in SCL. They are described in a specific namespace (IEC 61850-6-100), both covering functional description (IEC TR 61850-90-30) and application profile (IEC 61850-7-6).

The extension is specialized for IEC 61850-7-6 with a specific release as per definition in 1.3.

This clause details the different elements of the extension related to Basic Application Profile description.

Most of an application consists into referring to other elements defined somewhere in the SCL file. All these references are done by a path inside the SCL, based on the name of all elements in the hierarchy. The format of path is defined by the XSD type *tPathName*, described in TR IEC TR 61850-90-30. These references have to be maintained on any update of the SCL file.

6.1.2 Cardinality

The application profile needs to be adapted during instantiation process to automatically adapt itself to the context of the system, by multiplying selected elements based on system defined conditions and to fill in the links between the different parts of the projects and the application instance.

This is made possible with the usage of cardinality and selector attributes, as shown in Table 4.

```

<xs:simpleType name="tCardinalityEnum">
  <xs:restriction base="xs:normalizedString">
    <xs:enumeration value="0..1">
      <xs:annotation>
        <xs:documentation>Indicates that the element is optional and cannot be multiplied</xs:documentation>
      </xs:annotation>
    </xs:enumeration>
    <xs:enumeration value="1..1">
      <xs:annotation>
        <xs:documentation>Indicates that the element is not optional and cannot be multiplied</xs:documentation>
      </xs:annotation>
    </xs:enumeration>
    <xs:enumeration value="0..n">
      <xs:annotation>
        <xs:documentation>Indicates that the element is optional and can be multiplied</xs:documentation>
      </xs:annotation>
    </xs:enumeration>
    <xs:enumeration value="1..n">
      <xs:annotation>
        <xs:documentation>Indicates that the element is not optional and can be multiplied</xs:documentation>
      </xs:annotation>
    </xs:enumeration>
  </xs:restriction>
</xs:simpleType>
<xs:attributeGroup name="agCardinality">
  <xs:attribute name="cardinality" type="eIEC61850-6-100:tCardinalityEnum" use="optional" default="1..1">
    <xs:annotation>
      <xs:documentation>
        This attribute indicates if the user is able to duplicate the current element.
      </xs:documentation>
    </xs:annotation>
  </xs:attribute>
  <xs:attribute name="max" use="optional">
    <xs:annotation>
      <xs:documentation>
        This attribute indicates the maximum number of an element in case of cardinality allowing multiple instances. If
not defined, there is no limit
      </xs:documentation>
    </xs:annotation>
  </xs:attribute>
  <xs:simpleType>
    <xs:restriction base="xs:integer">
      <xs:minInclusive value="2"/>
    </xs:restriction>
  </xs:simpleType>
</xs:attribute>
</xs:attributeGroup>
<xs:attributeGroup name="agCardinalityWithSelector">
  <xs:attribute name="selector" type="xs:normalizedString" use="optional">
    <xs:annotation>
      <xs:documentation>
        Selector is used to give a searching rule to help user to select a target.
      </xs:documentation>
    </xs:annotation>
  </xs:attribute>
  <xs:attributeGroup ref="eIEC61850-6-100:agCardinality">
    <xs:annotation>
      <xs:documentation>
        Attributes used to select the cardinality and the max
      </xs:documentation>
    </xs:annotation>
  </xs:attributeGroup>
</xs:attributeGroup>

```

The *cardinality* and *selector* attributes are defined in Table 4.

Table 4 – Attributes for cardinality and selector

Attribute name	Description
cardinality	Indication of the possible number of instances of the given element which will be created during instantiation process. The values are in the following list: – 0..1: optional with a maximum of one instance – 1..1: mandatory with a maximum of one instance – 0..n: optional with multiple instances allowed – 1..n: mandatory with multiple instances allowed
max	When multiple instances are allowed, this attribute allows to restrict the maximum number of instances if needed.
selector	This attribute is used to identify the potential parents for the given element. This selector is defined by an XPath which is a query language for XML. This allows to indicate rules to search for specific elements in the SCL (e.g. searching a Bay with a specific name). XPath is a standard defined by W3C consortium freely available at https://www.w3.org/TR/xpath-3/

The rules associated to the *cardinality* and *selector* will depends on the context and are detailed in the element.

This cardinality definition is common to IEC TR 61850-90-30.

6.1.3 Application

The element *Application* describes what is composing the Application Profile.

```

<xs:complexType name="tApplication">
  <xs:annotation>
    <xs:documentation>Definition of the Application element</xs:documentation>
    <xs:appinfo>Definition in IEC 61850-7-6</xs:appinfo>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="eIEC61850-6-100:tBaseExtensionElementWithDesc">
      <xs:sequence>
        <xs:element name="FunctionRole" type="eIEC61850-6-100:tFunctionRole" maxOccurs="unbounded">
          <xs:annotation>
            <xs:documentation>Role attached to the application type</xs:documentation>
          </xs:annotation>
          <xs:unique name="uniqueRoleContent">
            <xs:selector xpath="/eIEC61850-6-100:FunctionRoleContent"/>
            <xs:field xpath="@roleInst"/>
          </xs:unique>
        </xs:element>
        <xs:element name="FunctionalVariant" type="eIEC61850-6-100:tFunctionalVariant" minOccurs="0"
maxOccurs="unbounded">
          <xs:annotation>
            <xs:documentation>Variant attached to the application type</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="FunctionalVariantGroup" type="eIEC61850-6-100:tFunctionalVariantGroup"
minOccurs="0" maxOccurs="unbounded">
          <xs:annotation>
            <xs:documentation>Group of variants attached to the application type</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="AllocationRoleRef" type="eIEC61850-6-100:tAllocationRoleRef" minOccurs="0"
maxOccurs="unbounded">
          <xs:annotation>
            <xs:documentation>Allocation role attached to the application type</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="ApplicationSclRef" type="eIEC61850-6-100:tApplicationSclRef" minOccurs="0">
          <xs:annotation>
            <xs:documentation>Reference to the ASD files used to create an Application</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

```

```

</xs:element>
</xs:sequence>
<xs:attribute name="type" type="scl:tID" use="optional">
  <xs:annotation>
    <xs:documentation>Identification of the application type if any</xs:documentation>
  </xs:annotation>
</xs:attribute>
<xs:attribute name="name" type="scl:tName" use="required">
  <xs:annotation>
    <xs:documentation>Name of the application</xs:documentation>
  </xs:annotation>
</xs:attribute>
</xs:extension>
</xs:complexContent>
</xs:complexType>

```

The attributes of the *Application* element are defined in Table 5.

Table 5 – Attributes of the Application element

Attribute name	Description
name	The name of the application used to identify it locally
desc	The textual description. Optional.
type	Identification of a type which could be used by Application owner to recognize it. Optional.
uuid	A unique identifier generated on creation of the Application instance.
templateUuid	The template unique identifier of the Application when created by importing a template file with an identifier already defined.
originUuid	The identifier of the original element in the source template when template is already based on another template.

Restrictions

- The *name* will be unique inside a parent.
- An application will be created inside a Process element (*Process*, *Line*, *Substation*, *VoltageLevel* or *Bay*)

Then, the Application is composed of Function Roles (see definition in 6.1.6) and could additionally defines Functional Variants (see definition in 6.1.5) and references Allocation Roles.

An application may also contain an *ApplicationSclReference*, which contains an *SCLFileReference* as per definition of IEC 61850-6:2009/AMD2:2024, which allows indication of the ASD file used to create the application instance, if any.

6.1.4 AllocationRoleRef

The *AllocationRoleRef* is used to refer an allocation role. An allocation role is defined by IEC TR 61850-90-30 and allows identification of how functions can be deployed to IEDs without specifying an S-IED.

```

<xs:complexType name="tAllocationRoleRef">
  <xs:annotation>
    <xs:documentation>Definition of the AllocationRoleRef element</xs:documentation>
    <xs:appinfo>Definition in IEC 61850-7-6</xs:appinfo>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="eIEC61850-6-100:tFunctionalVariantRefContainer">
      <xs:attribute name="allocationRole" type="eIEC61850-6-100:tPathName" use="required">
        <xs:annotation>
          <xs:documentation>Path to the allocation role referenced</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

```

The *AllocationRoleRef* is a *FunctionalVariantRefContainer* as defined in 6.1.5.

The attributes of the *AllocationRoleRef* element are defined in Table 6.

Table 6 – Attributes of the AllocationRoleRef element

Attribute name	Description
allocationRole	The path to the allocation role defined in the SCL. The path is the representation of the hierarchy from SCL top element down to allocation role element to reference. The path is the concatenation of the name all elements in the hierarchy, separated by the slash character '/'.
allocationRoleUuid	The UUID of the allocation role defined in the SCL.

Restrictions

- The *allocationRole* attribute is an absolute reference starting from the SCL root element.

6.1.5 Functional variants

Functional variants and subvariants are introduced within application templates to describe all variants of an application. Used within a specification tool, it allows a dynamic editing of an application instance. The goal is to limit the amount of application templates to be managed, to keep coherence between variants of the same application, and allow the user to select between different variants of an application. It is up to the user to select or not a functional variant, which will lead to a modification of the application content by:

- Adding elements (roles or references) within the application
- Removing elements (roles or references) within the application

A functional variant might also require overwriting the value of a variable. This will be done by defining a reference to this variable in the context of this variant with the specific value, by using a *VariableRef* element as defined in 6.1.7 with the specific value to use.

```

<xs:complexType name="tAbstractFunctionalVariant" abstract="true">
  <xs:annotation>
    <xs:documentation>Definition of a functional variant of the Application</xs:documentation>
    <xs:appinfo>Definition in IEC 61850-7-6</xs:appinfo>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="eIEC61850-6-100:tBaseExtensionElementWithDesc">
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element name="FunctionalSubVariant" type="eIEC61850-6-100:tFunctionalSubVariant" minOccurs="0"
maxOccurs="unbounded">
          <xs:annotation>
            <xs:documentation>Sub variant of the current variant</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="VariableRef" type="eIEC61850-6-100:tVariableRef" minOccurs="0"
maxOccurs="unbounded">
          <xs:annotation>
            <xs:documentation>Reference to a variable attached to current variant, to parameterize the
instance with the value defined in the reference</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:choice>
      <xs:attribute name="name" type="scl:tName" use="required">
        <xs:annotation>
          <xs:documentation>Name of the variant</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="isBaseline" type="xs:boolean" use="optional" default="false">
        <xs:annotation>
          <xs:documentation>Indication if the functional variant is used as baseline variant used as reference for
other variants</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

```

```

</xs:complexContent>
</xs:complexType>
<xs:complexType name="tFunctionalVariant">
  <xs:annotation>
    <xs:documentation>Definition of a functional variant of the Application</xs:documentation>
    <xs:appinfo>Definition in IEC 61850-7-6</xs:appinfo>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="eIEC61850-6-100:tAbstractFunctionalVariant"/>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="tFunctionalSubVariant">
  <xs:annotation>
    <xs:documentation>Definition of a functional sub variant of a variant</xs:documentation>
    <xs:appinfo>Definition in IEC 61850-7-6</xs:appinfo>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="eIEC61850-6-100:tAbstractFunctionalVariant"/>
  </xs:complexContent>
</xs:complexType>

```

The attributes of the *FunctionalVariant* and *FunctionalSubVariant* elements are defined in Table 7.

Table 7 – Attributes of the FunctionalVariant and FunctionalSubVariant elements

Attribute name	Description
name	The name of the variant / sub-variant.
desc	The textual description. Optional.
isBaseline	Indication if the functional variant is used as baseline variant used as reference for other variants. Optional. Default=True.
uuid	A unique identifier generated on creation of the functional (sub)variant instance.
templateUuid	The template unique identifier of the functional (sub)variant when created by importing a template file with an identifier already defined.
originUuid	The identifier of the original element in the source template when template is already based on another template.

Restrictions

- The variant *name* attribute will be unique among variant/sub-variant of the application.

A *FunctionalVariant* may contain *FunctionalSubVariant* to define a hierarchy of variants.

When *FunctionalSubVariant* are defined, during instantiation of the application, only one leaf variant will be selected. After selection, *FunctionalVariant* or *FunctionalSubVariant* not selected, and the associated elements will be removed from the instance (as explained in 6.2.5).

In addition to the hierarchy of variants, there is also the *FunctionalVariantGroup* which allows to select a set of *FunctionalVariant* which are exclusive. Only one variant of the group will be used during instantiation of the application.

```

<xs:complexType name="tFunctionalVariantGroup">
  <xs:annotation>
    <xs:documentation>
      Definition of a group of functional variant which are exclusive to each other.
      But an application may contains multiple groups which can be combined together
    </xs:documentation>
    <xs:appinfo>Definition in IEC 61850-7-6</xs:appinfo>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="eIEC61850-6-100:tBaseExtensionElementWithDesc">
      <xs:sequence>
        <xs:element name="FunctionalVariant" type="eIEC61850-6-100:tFunctionalVariant" minOccurs="0"
maxOccurs="unbounded">
          <xs:annotation>

```

```

        <xs:documentation>Variant of the current group</xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="name" type="scl:tName" use="required">
    <xs:annotation>
      <xs:documentation>Name of the group of variant</xs:documentation>
    </xs:annotation>
  </xs:attribute>
</xs:extension>
</xs:complexContent>
</xs:complexType>

```

The attributes of the *FunctionalVariantGroup* element are defined in Table 8.

Table 8 – Attributes of the FunctionalVariantGroup element

Attribute name	Description
name	The name of the group.
desc	The textual description. Optional.
uuid	A unique identifier generated on creation of the functional variant group instance.
templateUuid	The template unique identifier of the functional variant group when created by importing a template file with an identifier already defined.
originUuid	The identifier of the original element in the source template when template is already based on another template.

And finally, any element part of the Application content could be associated to one or multiple variants to be processed on Variant selection during instantiation process.

All elements which can be associated to variants will be a *FunctionalVariantRefContainer* which contains a list of *FunctionalVariantRef* identifying the variant and the modification to apply compared to the baseline variant if it is defined. The baseline variant is considered as the variant that upon selection defines all default functions of the application, meaning that the baseline variant cannot have any associated *FunctionalVariantRef* elements in the BAP.

```

<xs:complexType name="tFunctionalVariantRefContainer" abstract="true">
  <xs:annotation>
    <xs:documentation>Type which could be linked to functional variants</xs:documentation>
    <xs:appinfo>Definition in IEC 61850-7-6</xs:appinfo>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="eIEC61850-6-100:tBaseExtensionElementWithDesc">
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element name="FunctionalVariantRef" type="eIEC61850-6-100:tFunctionalVariantRef" minOccurs="0"
maxOccurs="unbounded">
          <xs:annotation>
            <xs:documentation>Ref to a functional variant of the Application</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:choice>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="tFunctionalVariantRef">
  <xs:annotation>
    <xs:documentation>Definition of a ref to a functional variant of the Application</xs:documentation>
    <xs:appinfo>Definition in IEC 61850-7-6</xs:appinfo>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="eIEC61850-6-100:tBaseExtensionElement">
      <xs:attribute name="functionalVariant" type="scl:tName" use="required">
        <xs:annotation>
          <xs:documentation>Name of the variant</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="update" use="optional" default="add">
        <xs:annotation>

```

```

        <xs:documentation>kind of update to apply to the variant</xs:documentation>
      </xs:annotation>
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="add"/>
        <xs:enumeration value="remove"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
</xs:extension>
</xs:complexContent>
</xs:complexType>

```

The attributes of the *FunctionalVariantRef* element are defined in Table 9.

Table 9 – Attributes of the FunctionalVariantRef element

Attribute name	Description
functionalVariant	The name to the functional variant defined in the application.
functionalVariantUuid	The UUID of the functional variant defined in the application.
update	The update action to apply to the element when the variant is selected. Could be add or remove. Default = add.

6.1.6 Function Roles

A function role is a role inside an application that can only contain references to elements participating to the Application (*Function/Subfunction*, *BehaviorDescription*, *ProcessResource* and *Variable*). Each function role identifies the elements which will be created for a given purpose represented by this role. Each role may also be created multiple times, depending on a cardinality defined in the application.

```

<xs:complexType name="tFunctionRole">
  <xs:annotation>
    <xs:documentation>Definition of the Function Role element</xs:documentation>
    <xs:appinfo>Definition in IEC 61850-7-6</xs:appinfo>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="eIEC61850-6-100:tFunctionalVariantRefContainer">
      <xs:sequence>
        <xs:choice minOccurs="1" maxOccurs="unbounded">
          <xs:element name="FunctionRoleContent" type="eIEC61850-6-100:tFunctionRoleContent">
            <xs:annotation>
              <xs:documentation>an instance of the current role. In case of application type definition,
instance 1 is the default instance</xs:documentation>
            </xs:annotation>
          </xs:element>
        </xs:choice>
      </xs:sequence>
      <xs:attribute name="name" type="scl:tName" use="required">
        <xs:annotation>
          <xs:documentation>Name of the role</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="type" type="xs:normalizedString">
        <xs:annotation>
          <xs:documentation>Type of the role (Predefined value = EXTERNAL)</xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attributeGroup ref="eIEC61850-6-100:agCardinalityWithSelector">
        <xs:annotation>
          <xs:documentation>
            Selector is used to give a searching rule to help user to select target(s) to deploy the given role.
            When the target is obvious it could even more be automatically deployed.
            When a selector is defined, the user may select different candidates matching this selector, and then
the FunctionRole

            The cardinality, indicated the rule to instanciate the role multiple times.
          </xs:documentation>
        </xs:annotation>
      </xs:attributeGroup>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

```

```

</xs:attributeGroup>
</xs:extension>
</xs:complexContent>
</xs:complexType>

```

The *FunctionRole* is a *FunctionalVariantRefContainer* as defined in 6.1.5.

The attributes of the *FunctionRole* element are defined in Table 10.

Table 10 – Attributes of the FunctionRole element

Attribute name	Description
name	The name of the role used to identify it in the application.
desc	The textual description. Optional.
type	Identification of a type which could be used by Application owner to recognize the role. Optional.
cardinality	Indication of the possible number of instances of the role which have to be created during instantiation process. Optional. (See Table 4 for details)
max	When multiple instances are allowed, this attribute allows to restrict the maximum number of instances if needed. Optional. (See Table 4 for details)
selector	This attribute is used to identify the potential parents for the elements composing the role. Optional. (See Table 1 for details)
uuid	A unique identifier generated on creation of the function role instance.
templateUuid	The template unique identifier of the function role when created by importing a template file with an identifier already defined.
originUuid	The identifier of the original element in the source template when template is already based on another template.

A *FunctionRole* is composed of *FunctionRoleContent* defining the elements participating to this specific role of the Application. This allows the instantiation of the content multiple times when cardinality allows multiple instances.

```

<xs:complexType name="tFunctionRoleContent">
  <xs:annotation>
    <xs:documentation>
      Definition of the content of a Function Role
      In case of a content which can be instantiated multiple times when Cardinality is greater than 1, then this content will
      be instantiated multiple times
    </xs:documentation>
    <xs:appinfo>Definition in IEC 61850-7-6</xs:appinfo>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="eIEC61850-6-100:tBaseExtensionElement">
      <xs:sequence>
        <xs:choice minOccurs="0" maxOccurs="unbounded">
          <xs:element name="FunctionRef" type="eIEC61850-6-100:tFunctionRef">
            <xs:annotation>
              <xs:documentation>Reference to a function definition attached to current
            </xs:documentation>
          </xs:element>
          <xs:element name="BehaviorDescriptionRef" type="eIEC61850-6-100:tBehaviorDescriptionRef">
            <xs:annotation>
              <xs:documentation>Reference to a behavior description attached to current
            </xs:documentation>
          </xs:element>
          <xs:element name="ProcessResourceRef" type="eIEC61850-6-100:tProcessResourceRef">
            <xs:annotation>
              <xs:documentation>Reference to a process resource attached to current
            </xs:documentation>
          </xs:element>
          <xs:element name="VariableRef" type="eIEC61850-6-100:tVariableRefWithUuid">
            <xs:annotation>

```

```

        <xs:documentation>Reference to a variable attached to current role</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="FunctionCategoryRef" type="eIEC61850-6-100:tFunctionCategoryRef">
      <xs:annotation>
        <xs:documentation>Reference to a function category attached to current
role</xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="PowerSystemRelationRef" type="eIEC61850-6-100:tPowerSystemRelationRef">
      <xs:annotation>
        <xs:documentation>Reference to a power system relation attached to current
role</xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:choice>
</xs:sequence>
<xs:attribute name="roleInst" use="optional" default="1">
  <xs:annotation>
    <xs:documentation>Instance of the role</xs:documentation>
  </xs:annotation>
  <xs:simpleType>
    <xs:restriction base="xs:integer">
      <xs:minInclusive value="1"/>
    </xs:restriction>
  </xs:simpleType>
</xs:attribute>
</xs:extension>
</xs:complexContent>
</xs:complexType>

```

The attributes of the *FunctionRoleContent* element are defined in Table 11.

Table 11 – Attributes of the FunctionRoleContent element

Attribute name	Description
roleInst	The instance number of the role content, when role is instantiated multiple time. Default = 1.

The *FunctionRoleContent* defines the list of references to SCL elements participating to the Application Role: *Function*, *BehaviorDescription*, *ProcessResource* and *Variable*.

6.1.7 Function Reference

The *FunctionRef* is used to make a reference to a function in the SCL which participate to the Application, with the different signals to include.

```

<xs:complexType name="tFunctionRef">
  <xs:annotation>
    <xs:documentation>Definition of the FunctionRef element</xs:documentation>
    <xs:appinfo>Definition in IEC 61850-7-6</xs:appinfo>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="eIEC61850-6-100:tFunctionalVariantRefContainer">
      <xs:sequence>
        <xs:element name="SignalRole" type="eIEC61850-6-100:tSignalRole" minOccurs="0"
maxOccurs="unbounded">
          <xs:annotation>
            <xs:documentation>Signal list attached to the referenced function in the context of the current Role
definition</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="function" type="eIEC61850-6-100:tPathName" use="required">
        <xs:annotation>
          <xs:documentation>Path to the function referenced</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

```

The *FunctionRef* is a *FunctionalVariantRefContainer* as defined in 6.1.5.

The attributes of the *FunctionRef* element are defined in Table 12.

Table 12 – Attributes of the FunctionRef element

Attribute name	Description
function	The path to the <i>Function</i> defined in the SCL. The path is the representation of the hierarchy from SCL top element down to <i>Function</i> element to reference. The path is the concatenation of the name all elements in the hierarchy, separated by the slash character '/
functionUuid	The UUID of the <i>Function</i> defined in the SCL.
desc	The textual description. Optional.

Restrictions

- The *function* attribute is an absolute reference starting from the SCL root element.

The *SignalRole* is used to describe the different data of a specific *Function* which needs to be included in the Application referring to the function. This includes inputs (*SourceRef*) and outputs (*ControlRef* and *DO / DA*). Any reference to *DO / DA* will refer the *LNodeType* definition even if *DOS / DAS* are not created. Creating individual data references of data participating to the application is mandatory, even if all elements of a function participate to the application. This will allow the documentation of whether that added data or dataflow during instantiation is part of the application or not.

```

<xs:complexType name="tSignalRole">
  <xs:annotation>
    <xs:documentation>Definition of the SignalRole element</xs:documentation>
    <xs:appinfo>Definition in IEC 61850-7-6</xs:appinfo>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="eIEC61850-6-100:tFunctionalVariantRefContainer">
      <xs:sequence>
        <xs:element name="LNodeInputRef" type="eIEC61850-6-100:tLNodeInputRef" minOccurs="0"
maxOccurs="unbounded">
          <xs:annotation>
            <xs:documentation>Reference to SourceRefs belonging to current role</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="LNodeOutputRef" type="eIEC61850-6-100:tLNodeOutputRef" minOccurs="0"
maxOccurs="unbounded">
          <xs:annotation>
            <xs:documentation>Reference to ControlRefs belonging to current role</xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="LNodeDataRef" type="eIEC61850-6-100:tLNodeDataRef" minOccurs="0"
maxOccurs="unbounded">
          <xs:annotation>
            <xs:documentation>Reference to Data belonging to current role</xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="name" type="scl:tName" use="required">
        <xs:annotation>
          <xs:documentation>Identification of the current signal role</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

```

The *SignalRole* is a *FunctionalVariantRefContainer* as defined in 6.1.5.

The attributes of the *SignalRole* element are defined in Table 13.

Table 13 – Attributes of the SignalRole element

Attribute name	Description
name	Name of the signal role in the context of the function
desc	The textual description. Optional.
uuid	A unique identifier generated on creation of the signal role instance.
templateUuid	The template unique identifier of the signal role when created by importing a template file with an identifier already defined.
originUuid	The identifier of the original element in the source template when template is already based on another template.

The *SignalRole* contains *LNodeInputRef*, *LNodeOutputRef* and *LNodeDataRef*, to reference respectively *SourceRef*, *ControlRef* and *DO/DA*.

```

<xs:complexType name="tLNodeInputRef">
  <xs:annotation>
    <xs:documentation>Definition of the LNodeInputRef element</xs:documentation>
    <xs:appinfo>Definition in IEC 61850-7-6</xs:appinfo>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="eIEC61850-6-100:tFunctionalVariantRefContainer">
      <xs:attribute name="sourceRef" type="eIEC61850-6-100:tPathName" use="required">
        <xs:annotation>
          <xs:documentation>Identification of the SourceRef belonging to the current role (reference the input
attribute of the SourceRef)</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="tLNodeOutputRef">
  <xs:annotation>
    <xs:documentation>Definition of the LNodeOutputRef element</xs:documentation>
    <xs:appinfo>Definition in IEC 61850-7-6</xs:appinfo>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="eIEC61850-6-100:tFunctionalVariantRefContainer">
      <xs:attribute name="controlRef" type="eIEC61850-6-100:tPathName" use="required">
        <xs:annotation>
          <xs:documentation>Identification of the ControlRef belonging to the current role (reference the output
attribute of the ControlRef)</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="tLNodeDataRef">
  <xs:annotation>
    <xs:documentation>Definition of the LNodeDataRef element</xs:documentation>
    <xs:appinfo>Definition in IEC 61850-7-6</xs:appinfo>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="eIEC61850-6-100:tFunctionalVariantRefContainer">
      <xs:attribute name="data" type="eIEC61850-6-100:tPathName" use="required">
        <xs:annotation>
          <xs:documentation>Identification of the Data (DO/DA) belonging to the current role (reference the Data
name)</xs:documentation>
        </xs:annotation>
      </xs:attribute>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

```

The *LNodeInputRef*, *LNodeOutputRef* and *LNodeDataRef* are *FunctionalVariantRefContainer* as defined in 6.1.5.

The attributes of the *LNodeInputRef* element are defined in Table 14.

Table 14 – Attributes of the LNodeInputRef element

Attribute name	Description
sourceRef	The path to the <i>SourceRef</i> (as per IEC TR 61850-90-30) defined in the <i>Function</i> . The path is the representation of the hierarchy from the <i>Function</i> element down to <i>SourceRef</i> element to reference. The path is the concatenation of the name all elements in the hierarchy, separated by the slash character '/'.
sourceRefUuid	The UUID of the <i>SourceRef</i> defined in the <i>Function</i> .
desc	The textual description. Optional.

The attributes of the *LNodeOutputRef* element are defined in Table 15.

Table 15 – Attributes of the LNodeOutputRef element

Attribute name	Description
controlRef	The path to the <i>ControlRef</i> (as per IEC TR 61850-90-30) defined in the <i>Function</i> . The path is the representation of the hierarchy from the <i>Function</i> element down to <i>ControlRef</i> element to reference. The path is the concatenation of the name all elements in the hierarchy, separated by the slash character '/'.
controlRefUuid	The UUID of the <i>ControlRef</i> defined in the <i>Function</i> .
desc	The textual description. Optional.

The attributes of the *LNodeDataRef* element are defined in Table 16.

Table 16 – Attributes of the LNodeDataRef element

Attribute name	Description
data	A reference to the source of the data (LNodeSpecNaming). The reference is a pathname in the Process Section. Down to the Logical Node Level the segments are separated by "/". The segments of the Data Object / Attribute are separated by ".". There is no Functional Constraint element. The source of the data is identified by the specified LNode. This keeps the reference valid even when the LNode is mapped.
InodeUuid	UUID of the LNode which contain the data defined in the <i>Function</i> .
doName	Name of the DO defined in the <i>Function</i> when InodeUuid is defined.
daName	Name of the DA defined in the <i>Function</i> when InodeUuid is defined.
desc	The textual description. Optional.

Restrictions

- The *sourceRef*, *controlRef* and *data* attributes is a relative reference starting from the *Function* element, without including the *Function* itself.

6.1.8 Variable reference

Variable elements, as described in IEC TR 61850-90-30, exist inside the substation section to allow specifying a value and applying it to multiple elements of the specification. In this way the user can easily specify the value of a single variable and applying it to many elements which use this value.