European Network of
Transmission System Operators
for Electricity

STEADY STATE HYPOTHESIS SCHEDULE PROFILE SPECIFICATION

2026-09-10

ICTC APPROVED
VERSION 1.1.1

Copyright notice:

This document is maintained by the ENTSO-E CIM WG. Comments or remarks are to be provided at cim@entsoe.eu

NOTE CONCERNING WORDING USED IN THIS DOCUMENT

The force of the following words is modified by the requirement level of the document in which they are used.

- SHALL: This word, or the terms “REQUIRED” or “MUST”, means that the definition is an absolute requirement of the specification.
- SHALL NOT: This phrase, or the phrase “MUST NOT”, means that the definition is an absolute prohibition of the specification.
- SHOULD: This word, or the adjective “RECOMMENDED”, means that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.
- SHOULD NOT: This phrase, or the phrase “NOT RECOMMENDED”, means that there may exist valid reasons in particular circumstances when the particular behaviour is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behaviour described with this label.
- MAY: This word, or the adjective “OPTIONAL”, means that an item is truly optional.

33

Revision History

Version	Date	Paragraph	Comments
1.0.0-alpha	2024-03-20		For CIM WG review.
1.0.0-beta	2024-04-08		For StG Strategy review.
1.0.1-alpha	2024-09-07		For CIM WG review.
1.0.2-alpha	2025-01-04		For CIM WG review.
1.1.0	2025-09-11		ICTC approval.
1.1.1	2026-08-05		For CIM WG review.
1.1.1	2026-09-10		ICTC approval.

34 # CONTENTS

List of figures

List of tables

1 Introduction

The steady state hypothesis schedule profile enables an exchange of schedules of operating point (steady state hypothesis).

2 Application profile specification

2.1 Version information

The content is generated from UML model file CIM17-2_CGMES31v01_PROF-20v03_NC25v25_MS10v12_DES10v01.qea.

This edition is based on the IEC 61970 UML version ‘’, dated ‘’.

- Title:
- Keyword:
- Description:
- Version IRI:
- Version info:
- Prior version:
- Conforms to:
- Identifier:

2.2 Constraints naming convention

The naming of the rules shall not be used for machine processing. The rule names are just a string. The naming convention of the constraints is as follows.

“{rule.Type}:{rule.Standard}:{rule.Profile}:{rule.Property}:{rule.Name}”

where

rule.Type: C – for constraint; R – for requirement

rule.Standard: the number of the standard e.g. 301 for 61970-301, 456 for 61970-456, 13 for 61968-13. 61970-600 specific constraints refer to 600 although they are related to one or combination of the 61970-450 series profiles. For NC profiles, NC is used.

rule.Profile: the abbreviation of the profile, e.g. TP for Topology profile. If set to “ALL” the constraint is applicable to all IEC 61970-600 profiles.

rule.Property: for UML classes, the name of the class, for attributes and associations, the name of the class and attribute or association end, e.g. EnergyConsumer, IdentifiedObject.name, etc. If set to “NA” the property is not applicable to a specific UML element.

rule.Name: the name of the rule. It is unique for the same property.

Example: C:600:ALL:IdentifiedObject.name:stringLength

2.3 Profile constraints

This clause defines requirements and constraints that shall be fulfilled by applications that conform to this document.

This document is the master for rules and constraints tagged "NC". For the sake of self-containment, the list below also includes a copy of the relevant rules from IEC 61970-452, tagged "452".

- C:452:ALL:NA:datatypes

  According to 61970-501, datatypes are not exchanged in the instance data. The UnitMultiplier is 1 in cases none value is specified in the profile.

- R:452:ALL:NA:exchange

  Optional and required attributes and associations must be imported and exported if they are in the model file prior to import.

- R:452:ALL:NA:exchange1

  If an optional attribute does not exist in the imported file, it does not have to be exported in case exactly the same data set is exported, i.e. the tool is not obliged to automatically provide this attribute. If the export is resulting from an action by the user performed after the import, e.g. data processing or model update the export can contain optional attributes.

- R:452:ALL:NA:exchange2

  In most of the profiles the selection of optional and required attributes is made so as to ensure a minimum set of required attributes without which the exchange does not fulfil its basic purpose. Business processes governing different exchanges can require mandatory exchange of certain optional attributes or associations. Optional and required attributes and associations shall therefore be supported by applications which claim conformance with certain functionalities of the IEC 61970-452. This provides flexibility for the business processes to adapt to different business requirements and base the exchanges on IEC 61970-452 compliant applications.

- R:452:ALL:NA:exchange3

  An exporter may, at his or her discretion, produce a serialization containing additional class data described by the CIM Schema but not required by this document provided these data adhere to the conventions established in Clause 5.

- R:452:ALL:NA:exchange4

  From the standpoint of the model import used by a data recipient, the document describes a subset of the CIM that importing software shall be able to interpret in order to import exported models. Data providers are free to exceed the minimum requirements described herein as long as their resulting data files are compliant with the CIM Schema and the conventions established in Clause 5. The document, therefore, describes additional classes and class data that, although not required, exporters will, in all likelihood, choose to include in their data files. The additional classes and data are labelled as required (cardinality 1..1) or as optional (cardinality 0..1) to distinguish them from their required counterparts. Please note, however, that data importers could potentially receive data containing instances of any and all classes described by the CIM Schema.

- R:452:ALL:NA:cardinality

  The cardinality defined in the CIM model shall be followed, unless a more restrictive cardinality is explicitly defined in this document. For instance, the cardinality on the association between VoltageLevel and BaseVoltage indicates that a VoltageLevel shall

be associated with one and only one BaseVoltage, but a BaseVoltage can be associated with zero to many VoltageLevels.

- R:452:ALL:NA:associations

  Associations between classes referenced in this document and classes not referenced here are not required regardless of cardinality.

- R:452:ALL:IdentifiedObject.name:rule

  The attribute “name” inherited by many classes from the abstract class IdentifiedObject is not required to be unique. It must be a human readable identifier without additional embedded information that would need to be parsed. The attribute is used for purposes such as User Interface and data exchange debugging. The MRID defined in the data exchange format is the only unique and persistent identifier used for this data exchange. The attribute IdentifiedObject.name is, however, always required for CoreEquipment profile and Short Circuit profile.

- R:452:ALL:IdentifiedObject.description:rule

  The attribute “description” inherited by many classes from the abstract class IdentifiedObject must contain human readable text without additional embedded information that would need to be parsed.

- R:452:ALL:NA:uniqueIdentifier

  All IdentifiedObject-s shall have a persistent and globally unique identifier (Master Resource Identifier - mRID).

- R:452:ALL:NA:unitMultiplier

  For exchange of attributes defined using CIM Data Types (ActivePower, Susceptance, etc.) a unit multiplier of 1 is used if the UnitMultiplier specified in this document is “none”.

- C:452:ALL:IdentifiedObject.name:stringLength

  The string IdentifiedObject.name has a maximum of 128 characters.

- C:452:ALL:IdentifiedObject.description:stringLength

  The string IdentifiedObject.description is maximum 256 characters.

- C:452:ALL:NA:float

  An attribute that is defined as float (e.g. has a type Float or a type which is a Datatype with .value attribute of type Float) shall support ISO/IEC 60559:2020 for floating-point arithmetic using single precision floating point. A single precision float supports 7 significant digits where the significant digits are described as an integer, or a decimal number with 6 decimal digits. Two float values are equal when the significant with 7 digits are identical, e.g. 1234567 is equal 1.234567E6 and so are 1.2345678 and 1.234567E0.

- R:NC:ALL:NA:serialization

  The profiles are defined in the EnterpriseArchitect application and have multiple artifacts that describe them. The main artifacts are:

  1) the EAP file (EnterpriseArchitect project file),
  2) the profiles’ specification document and

3) the application profiles (RDFS and SHACL).

Due to the complexity of the profiles, there are various cross profile associations that, from profiling and profile maintenance point of view, it is not practical to include the complete inheritance structure in all profiles. If this is done the documentation provided for all profiles would also include duplicated information on the description of classes defined in other profiles. The following cases are often observed in profiles:

- Case 1: An association end refers to an abstract class
- Case 2: An abstract class (stereotyped with “Description”) has an association (direction to another class)
- Case 3: An abstract class (not stereotyped with “Description”) has an association (direction to another class)
- Case 4: An abstract class has attributes and subclasses are not in the profile

In all cases, the datasets shall only include the subtypes of the abstract classes with the related properties (i.e. association or attributes) defined in the profile. The information is taken from either canonical model or the profiles where complete (expected) inheritance structure for the related abstract class is described. SHACL based constraints include constraints only for the concrete classes that are subtypes of the abstract class in the profile, and this can be used to inform which are the concrete classes expected in a dataset that conforms to this profile.

It should be taken into account that this approach deviates from MVAL5 (IEC 61970-600-1:2021), which creates multiple inheritance at serialization. For instance, with this more explicit exchange the serialization of the association between abstract class Equipment and abstract class Circuit for a PowerTransformer will be serialized as follows:

- for association

```
<cim:PowerTransformer rdf:about="_c328f787-bc17-47ad-a59f-6ba7133340d0">
  <nc:Equipment.Circut rdf:resource="#_9ced16ac-d076-4ef9-a241-a998a579e77b"/>
</cim:PowerTransformer>
```

- for attribute

```
<cim:ACLineSegment rdf:about="_04f681aa-6999-4fb3-9775-acaa5eb7ceff">
  <cim:Equipment.inService>true</cim:Equipment.inService>
</cim:ACLineSegment>
```

The usage of rdf:ID or rdf:about depends on the stereotype of the class. rdf:about is used if the class has the stereotype “Description”.

An example of not allowed serialization, as the Equipment is an abstract class

```
<cim:Equipment rdf:about="_c328f787-bc17-47ad-a59f-6ba7133340d0">
  <nc:Equipment.Circut rdf:resource="#_9ced16ac-d076-4ef9-a241-a998a579e77b"/>
</cim:Equipment>
```

2.4 Metadata

ENTSO-E agreed to extend the header and metadata definitions by IEC 61970-552 Ed2. This new header definitions rely on W3C recommendations which are used worldwide and are positively recognized by the European Commission. The new definitions of the header mainly use Provenance ontology (PROV-O), Time Ontology and Data Catalog Vocabulary (DCAT). The

global new header applicable for this profile is included in the metadata and document header specification document.

The header vocabulary contains all attributes defined in IEC 61970-552. This is done only for the purpose of having one vocabulary for header and to ensure transition for data exchanges that are using IEC 61970-552:2016 header. This profile does not use IEC 61970-552:2016 header attributes and relies only on the extended attributes.

2.4.1 Constraints

The identification of the constraints related to the metadata follows the same convention for naming of the constraints as for profile constraints.

- R:NC:ALL:wasAttributedTo:usage

  The prov:wasAttributedTo should normally be the “X” EIC code of the actor or their URI (prov:Agent).

2.4.2 Reference metadata

The header defined for this profile requires availability of a set of reference metadata. For instance, the attribute prov:wasGeneratedBy requires a reference to an activity which produced the model or the related process. The activities are defined as reference metadata and their identifiers are referenced from the header to enable the receiving entity to retrieve the “static” (reference) information that is not modified frequently. This approach imposes a requirement that both the sending entity and the receiving entity have access to a unique version of the reference metadata. Therefore, each business process shall define which reference metadata is used and where it is located.

3 Package SteadyStateHypothesisScheduleProfile

3.1 General

This package contains steady state hypothesis schedule profile.

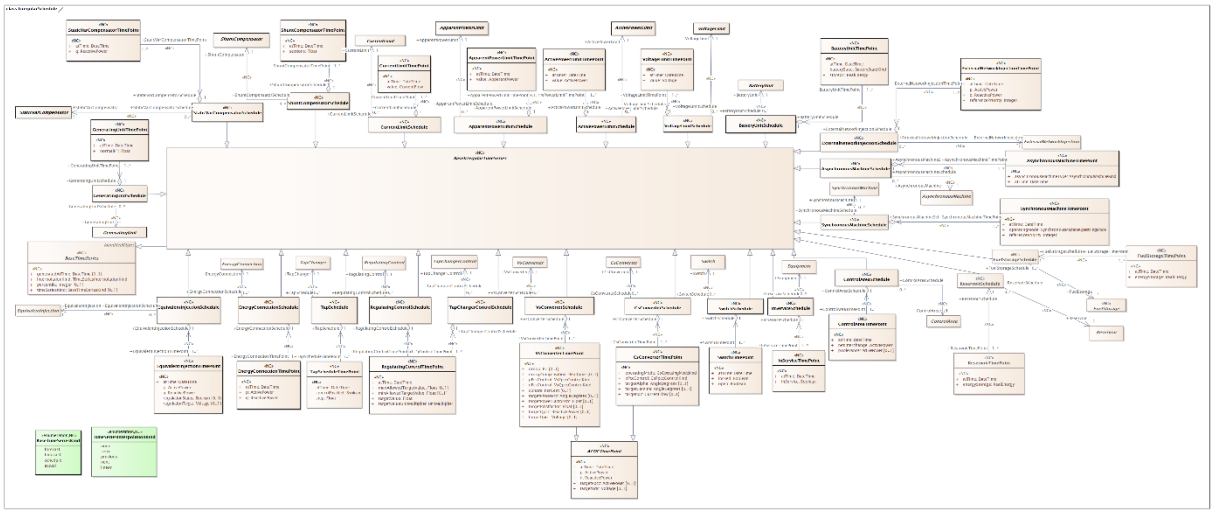

Figure 1 – Class diagram SteadyStateHypothesisScheduleProfile::IrregularSchedule

Figure 1: The diagram shows classes related to the irregular schedule.

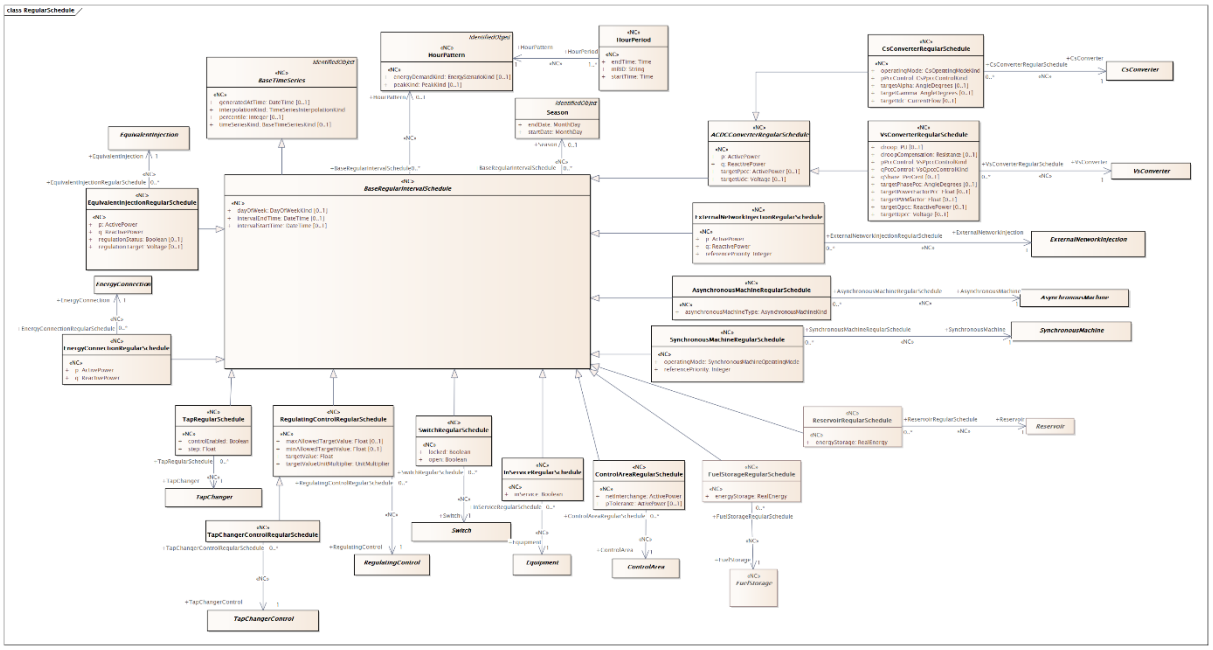

Figure 2 – Class diagram SteadyStateHypothesisScheduleProfile::RegularSchedule

Figure 2: The diagram shows classes related to the regular schedule.

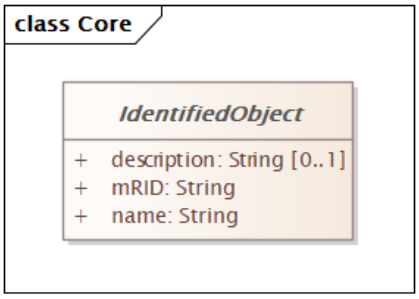

Figure 3 – Class diagram SteadyStateHypothesisScheduleProfile::Core

Figure 3: The diagram shows classes from Base CIM used in the profile.

3.2 (abstract) ActivePowerLimit root class

Limit on active power flow.

3.3 (abstract) ApparentPowerLimit root class

Apparent power limit.

3.4 (abstract) AsynchronousMachine root class

A rotating machine whose shaft rotates asynchronously with the electrical field. Also known as an induction machine with no external connection to the rotor windings, e.g. squirrel-cage induction machine.

3.5 (abstract) BatteryUnit root class

An electrochemical energy storage device.

3.6 (abstract) ControlArea root class

A control area is a grouping of generating units and/or loads and a cutset of tie lines (as terminals) which may be used for a variety of purposes including automatic generation control, power flow solution area interchange control specification, and input to load forecasting. All

generation and load within the area defined by the terminals on the border are considered in the area interchange control. Note that any number of overlapping control area specifications can be superimposed on the physical model. The following general principles apply to ControlArea:

1. The control area orientation for net interchange is positive for an import, negative for an export.
2. The control area net interchange is determined by summing flows in Terminals. The Terminals are identified by creating a set of TieFlow objects associated with a ControlArea object. Each TieFlow object identifies one Terminal.
3. In a single network model, a tie between two control areas must be modelled in both control area specifications, such that the two representations of the tie flow sum to zero.
4. The normal orientation of Terminal flow is positive for flow into the conducting equipment that owns the Terminal. (i.e. flow from a bus into a device is positive.) However, the orientation of each flow in the control area specification must align with the control area convention, i.e. import is positive. If the orientation of the Terminal flow referenced by a TieFlow is positive into the control area, then this is confirmed by setting TieFlow.positiveFlowIn flag TRUE. If not, the orientation must be reversed by setting the TieFlow.positiveFlowIn flag FALSE.

3.7 (abstract) CsConverter root class

DC side of the current source converter (CSC).
The firing angle controls the dc voltage at the converter, both for rectifier and inverter. The difference between the dc voltages of the rectifier and inverter determines the dc current. The extinction angle is used to limit the dc voltage at the inverter, if needed, and is not used in active power control. The firing angle, transformer tap position and number of connected filters are the primary means to control a current source dc line. Higher level controls are built on top, e.g. dc voltage, dc current and active power. From a steady state perspective it is sufficient to specify the wanted active power transfer (ACDCConverter.targetPpcc) and the control functions will set the dc voltage, dc current, firing angle, transformer tap position and number of connected filters to meet this. Therefore attributes targetAlpha and targetGamma are not applicable in this case.
The reactive power consumed by the converter is a function of the firing angle, transformer tap position and number of connected filter, which can be approximated with half of the active power. The losses is a function of the dc voltage and dc current.
The attributes minAlpha and maxAlpha define the range of firing angles for rectifier operation between which no discrete tap changer action takes place. The range is typically 10-18 degrees.
The attributes minGamma and maxGamma define the range of extinction angles for inverter operation between which no discrete tap changer action takes place. The range is typically 17-20 degrees.

3.8 (abstract) CurrentLimit root class

Operational limit on current.

3.9 (abstract) EnergyConnection root class

A connection of energy generation or consumption on the power system model.

3.10 (abstract) Equipment root class

The parts of a power system that are physical devices, electronic or mechanical.

3.11 (abstract) EquivalentInjection root class

This class represents equivalent injections (generation or load). Voltage regulation is allowed only at the point of connection.

3.12 (abstract) ExternalNetworkInjection root class

This class represents the external network and it is used for IEC 60909 calculations.

3.13 (abstract) GeneratingUnit root class

A single or set of synchronous machines for converting mechanical power into alternating-current power. For example, individual machines within a set may be defined for scheduling

purposes while a single control signal is derived for the set. In this case there would be a GeneratingUnit for each member of the set and an additional GeneratingUnit corresponding to the set.

3.14 (abstract) IdentifiedObject root class

This is a root class to provide common identification for all classes needing identification and naming attributes.
Table 1 shows all attributes of IdentifiedObject.

Table 1 – Attributes of SteadyStateHypothesisScheduleProfile::IdentifiedObject

name	mult	type	description
description	0..1	String	The description is a free human readable text describing or naming the object. It may be non unique and may not correlate to a naming hierarchy.
mRID	1..1	String	Master resource identifier issued by a model authority. The mRID is unique within an exchange context. Global uniqueness is easily achieved by using a UUID, as specified in RFC 4122, for the mRID. The use of UUID is strongly recommended. For CIMXML data files in RDF syntax conforming to IEC 61970-552, the mRID is mapped to rdf:ID or rdf:about attributes that identify CIM object elements.
name	1..1	String	The name is any free human readable and possibly non unique text naming the object.

3.15 (abstract) RegulatingControl root class

Specifies a set of equipment that works together to control a power system quantity such as voltage or flow.
Remote bus voltage control is possible by specifying the controlled terminal located at some place remote from the controlling equipment.
The specified terminal shall be associated with the connectivity node of the controlled point. The most specific subtype of RegulatingControl shall be used in case such equipment participate in the control, e.g. TapChangerControl for tap changers.
For flow control, load sign convention is used, i.e. positive sign means flow out from a TopologicalNode (bus) into the conducting equipment.
The attribute minAllowedTargetValue and maxAllowedTargetValue are required in the following cases:
- For a power generating module operated in power factor control mode to specify maximum and minimum power factor values;
- Whenever it is necessary to have an off center target voltage for the tap changer regulator. For instance, due to long cables to off shore wind farms and the need to have a simpler setup at the off shore transformer platform, the voltage is controlled from the land at the connection point for the off shore wind farm. Since there usually is a voltage rise along the cable, there is typical and overvoltage of up 3-4 kV compared to the on shore station. Thus in normal operation the tap changer on the on shore station is operated with a target set point, which is in the lower parts of the dead band.

The attributes minAllowedTargetValue and maxAllowedTargetValue are not related to the attribute targetDeadband and thus they are not treated as an alternative of the targetDeadband. They are needed due to limitations in the local substation controller. The attribute targetDeadband is used to prevent the power flow from move the tap position in circles (hunting) that is to be used regardless of the attributes minAllowedTargetValue and maxAllowedTargetValue.

3.16 (abstract) Reservoir root class

A water storage facility within a hydro system, including: ponds, lakes, lagoons, and rivers. The storage is usually behind some type of dam.

3.17 Season

Inheritance path = IdentifiedObject
A specified time period of the year.
Table 2 shows all attributes of Season.

Table 2 – Attributes of SteadyStateHypothesisScheduleProfile::Season

name	mult	type	description
endDate	1..1	MonthDay	Date season ends.
startDate	1..1	MonthDay	Date season starts.
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

3.18 (abstract) ShuntCompensator root class

A shunt capacitor or reactor or switchable bank of shunt capacitors or reactors. A section of a shunt compensator is an individual capacitor or reactor. A negative value for bPerSection indicates that the compensator is a reactor. ShuntCompensator is a single terminal device. Ground is implied.

3.19 (abstract) StaticVarCompensator root class

A facility for providing variable and controllable shunt reactive power. The SVC typically consists of a stepdown transformer, filter, thyristor-controlled reactor, and thyristor-switched capacitor arms.
The SVC may operate in fixed MVar output mode or in voltage control mode. When in voltage control mode, the output of the SVC will be proportional to the deviation of voltage at the controlled bus from the voltage setpoint. The SVC characteristic slope defines the proportion. If the voltage at the controlled bus is equal to the voltage setpoint, the SVC MVar output is zero.

3.20 (abstract) Switch root class

A generic device designed to close, or open, or both, one or more electric circuits. All switches are two terminal devices including grounding switches. The ACDCTerminal.connected at the two sides of the switch shall not be considered for assessing switch connectivity, i.e. only Switch.open, .normalOpen and .locked are relevant.

3.21 (abstract) SynchronousMachine root class

An electromechanical device that operates with shaft rotating synchronously with the network. It is a single machine operating either as a generator or synchronous condenser or pump.

3.22 (abstract) TapChanger root class

Mechanism for changing transformer winding tap positions.

3.23 (abstract) TapChangerControl root class

Describes behaviour specific to tap changers, e.g. how the voltage at the end of a line varies with the load level and compensation of the voltage drop by tap adjustment.

3.24 (abstract) VoltageLimit root class

Operational limit applied to voltage.
The use of operational VoltageLimit is preferred instead of limits defined at VoltageLevel. The operational VoltageLimits are used, if present.

3.25 (abstract) VsConverter root class

DC side of the voltage source converter (VSC).

3.26 AsynchronousMachineKind enumeration

Kind of Asynchronous Machine.
Table 3 shows all literals of AsynchronousMachineKind.

Table 3 – Literals of SteadyStateHypothesisScheduleProfile::AsynchronousMachineKind

literal	value	description
generator		The Asynchronous Machine is a generator.
motor		The Asynchronous Machine is a motor.

3.27 (NC) BaseTimeSeriesKind enumeration

Kind of time series.
Table 4 shows all literals of BaseTimeSeriesKind.

Table 4 – Literals of SteadyStateHypothesisScheduleProfile::BaseTimeSeriesKind

literal	value	description
forecast		Time series is forecast data. The values represent the result of scientific predictions based on historical time stamped data.
hindcast		Time series is hindcast data. The value represent probable past (historic) condition given by calculation done using actual values. For instance, determine the among of wind based on the energy produced by wind. However, hindcast is typical the result of a simulated forecasts for historical periods.
schedule		Time series is schedule data. The values represent the result of a committed and plan forecast data that has been through a quality control and could incur penalty when not followed.
actual		Time series is actual data. The values represent measured or calculated values that represent the actual behaviour.

3.28 BatteryStateKind enumeration

The state of the battery unit.
Table 5 shows all literals of BatteryStateKind.

Table 5 – Literals of SteadyStateHypothesisScheduleProfile::BatteryStateKind

literal	value	description
discharging		Stored energy is decreasing.
full		Unable to charge, and not discharging.
waiting		Neither charging nor discharging, but able to do so.
charging		Stored energy is increasing.
empty		Unable to discharge, and not charging.

3.29 CsOperatingModeKind enumeration

Operating mode for HVDC line operating as Current Source Converter.
Table 6 shows all literals of CsOperatingModeKind.

Table 6 – Literals of SteadyStateHypothesisScheduleProfile::CsOperatingModeKind

literal	value	description
inverter		Operating as inverter, which is the power receiving end.
rectifier		Operating as rectifier, which is the power sending end.

3.30 CsPpccControlKind enumeration

Active power control modes for HVDC line operating as Current Source Converter.
Table 7 shows all literals of CsPpccControlKind.

Table 7 – Literals of SteadyStateHypothesisScheduleProfile::CsPpccControlKind

literal	value	description
activePower		Control is active power control at AC side, at point of common coupling. Target is provided by ACDCConverter.targetPpcc.
dcVoltage		Control is DC voltage with target value provided by ACDCConverter.targetUdc.
dcCurrent		Control is DC current with target value provided by CsConverter.targetIdc.

3.31 (NC) DayOfWeekKind enumeration

The kind of day to be included in a regular schedule.
Table 8 shows all literals of DayOfWeekKind.

Table 8 – Literals of SteadyStateHypothesisScheduleProfile::DayOfWeekKind

literal	value	description
monday		Monday as the day of the week.
tuesday		Tuesday as the day of the week.
wednesday		Wednesday as the day of the week.
thursday		Thursday as the day of the week.
friday		Friday as the day of the week.
saturday		Saturday as the day of the week.
sunday		Sunday as the day of the week.
weekday		Day of the week other than Sunday or Saturday.
weekend		Day of the week which is Sunday or Saturday.
all		All days of the week.
publicHoliday		Public holiday is an officially designated day when most workplaces and schools are closed.
bridgeDay		A day that is a gap between two distinguished days e.g holiday and weekend that leads to an abnormal scheduling behavior. e.g. if Ascension day falls on a Thursday, then Friday would be a bridge day due to the schedule will not have a normal Friday consumption and production.

3.32 (NC) EnergyScenarioKind enumeration

Kind of energy scenario.
Table 9 shows all literals of EnergyScenarioKind.

Table 9 – Literals of SteadyStateHypothesisScheduleProfile::EnergyScenarioKind

literal	value	description
consumption		Scenario focus on consumption of energy.
production		Scenario focus on production of energy.
storage		Scenario focus on storage of energy.
export		Scenario focus on export of energy.
import		Scenario focus on import of energy.

3.33 (NC) PeakKind enumeration

Kind of time period with similar intensity.
Table 10 shows all literals of PeakKind.

Table 10 – Literals of SteadyStateHypothesisScheduleProfile::PeakKind

literal	value	description
offPeak		Off-peak refer to periods of lower demand for a particular service or commodity.
onPeak		Off-peak refer to periods of higher demand for a particular service or commodity.

3.34 SynchronousMachineOperatingMode enumeration

Synchronous machine operating mode.
Table 11 shows all literals of SynchronousMachineOperatingMode.

Table 11 – Literals of SteadyStateHypothesisScheduleProfile::SynchronousMachineOperatingMode

literal	value	description
generator		Operating as generator.
condenser		Operating as condenser.
motor		Operating as motor.

3.35 (NC) TimeSeriesInterpolationKind enumeration

Kinds of interpolation of values between two time point.
Table 12 shows all literals of TimeSeriesInterpolationKind.

Table 12 – Literals of SteadyStateHypothesisScheduleProfile::TimeSeriesInterpolationKind

literal	value	description
none		No interpolation is applied.
zero		The value between two time points is set to zero.
previous		The value between two time points is set to previous value.

literal	value	description
next		The value between two time points is set to next value.
linear		Linear interpolation is applied for values between two time points.

3.36 UnitMultiplier enumeration

The unit multipliers defined for the CIM. When applied to unit symbols, the unit symbol is treated as a derived unit. Regardless of the contents of the unit symbol text, the unit symbol shall be treated as if it were a single-character unit symbol. Unit symbols should not contain multipliers, and it should be left to the multiplier to define the multiple for an entire data type.
For example, if a unit symbol is "m2Pers" and the multiplier is "k", then the value is k(m**2/s), and the multiplier applies to the entire final value, not to any individual part of the value. This can be conceptualized by substituting a derived unit symbol for the unit type. If one imagines that the symbol "Þ" represents the derived unit "m2Pers", then applying the multiplier "k" can be conceptualized simply as "kÞ".
For example, the SI unit for mass is "kg" and not "g". If the unit symbol is defined as "kg", then the multiplier is applied to "kg" as a whole and does not replace the "k" in front of the "g". In this case, the multiplier of "m" would be used with the unit symbol of "kg" to represent one gram. As a text string, this violates the instructions in IEC 80000-1. However, because the unit symbol in CIM is treated as a derived unit instead of as an SI unit, it makes more sense to conceptualize the "kg" as if it were replaced by one of the proposed replacements for the SI mass symbol. If one imagines that the "kg" were replaced by a symbol "Þ", then it is easier to conceptualize the multiplier "m" as creating the proper unit "mÞ", and not the forbidden unit "mkg".
Table 13 shows all literals of UnitMultiplier.

Table 13 – Literals of SteadyStateHypothesisScheduleProfile::UnitMultiplier

literal	value	description
none		No multiplier or equivalently multiply by 1.
k		Kilo 10**3.
M		Mega 10**6.

3.37 UnitSymbol enumeration

The derived units defined for usage in the CIM. In some cases, the derived unit is equal to an SI unit. Whenever possible, the standard derived symbol is used instead of the formula for the derived unit. For example, the unit symbol Farad is defined as "F" instead of "CPerV". In cases where a standard symbol does not exist for a derived unit, the formula for the unit is used as the unit symbol. For example, density does not have a standard symbol and so it is represented as "kgPerm3". With the exception of the "kg", which is an SI unit, the unit symbols do not contain multipliers and therefore represent the base derived unit to which a multiplier can be applied as a whole.
Every unit symbol is treated as an unparseable text as if it were a single-letter symbol. The meaning of each unit symbol is defined by the accompanying descriptive text and not by the text contents of the unit symbol.
To allow the widest possible range of serializations without requiring special character handling, several substitutions are made which deviate from the format described in IEC 80000-1. The division symbol "/" is replaced by the letters "Per". Exponents are written in plain text after the unit as "m3" instead of being formatted as "m" with a superscript of 3 or introducing a symbol as in "m^3". The degree symbol "°" is replaced with the letters "deg". Any clarification of the meaning for a substitution is included in the description for the unit symbol.
Non-SI units are included in list of unit symbols to allow sources of data to be correctly labelled with their non-SI units (for example, a GPS sensor that is reporting numbers that represent feet

913 instead of meters). This allows software to use the unit symbol information correctly convert
914 and scale the raw data of those sources into SI-based units.
915 The integer values are used for harmonization with IEC 61850.
916 Table 14 shows all literals of UnitSymbol.

917 **Table 14 – Literals of SteadyStateHypothesisScheduleProfile::UnitSymbol**

literal	value	description
none		Dimension less quantity, e.g. count, per unit, etc.
A		Current in amperes.
deg		Plane angle in degrees.
V		Electric potential in volts (W/A).
ohm		Electric resistance in ohms (V/A).
W		Real power in watts (J/s). Electrical power may have real and reactive components. The real portion of electrical power (I^2R or VIcos(phi)), is expressed in Watts. See also apparent power and reactive power.
VA		Apparent power in volt amperes. See also real power and reactive power.
VAr		Reactive power in volt amperes reactive. The “reactive” or “imaginary” component of electrical power (VIsin(phi)). (See also real power and apparent power). Note: Different meter designs use different methods to arrive at their results. Some meters may compute reactive power as an arithmetic value, while others compute the value vectorially. The data consumer should determine the method in use and the suitability of the measurement for the intended purpose.
Wh		Real energy in watt hours.

918

919 ### 3.38 VsPpccControlKind enumeration

920 Types applicable to the control of real power and/or DC voltage by voltage source converter.
921 Table 15 shows all literals of VsPpccControlKind.

922 **Table 15 – Literals of SteadyStateHypothesisScheduleProfile::VsPpccControlKind**

literal	value	description
pPcc		Control is real power at point of common coupling. The target value is provided by ACDCConverter.targetPpcc.
udc		Control is DC voltage with target value provided by ACDCConverter.targetUdc.
pPccAndUdcDroop		Control is active power at point of common coupling and local DC voltage, with the droop. Target values are provided by ACDCConverter.targetPpcc, ACDCConverter.targetUdc and VsConverter.droop.
pPccAndUdcDroopWithCompensation		Control is active power at point of common coupling and compensated DC voltage, with the droop. Compensation factor is the resistance, as an approximation of the DC voltage of a common (real or virtual) node in the DC network. Targets are provided by ACDCConverter.targetPpcc,

literal	value	description
		ACDCConverter.targetUdc, VsConverter.droop and VsConverter.droopCompensation.
pPccAndUdcDroopPilot		Control is active power at point of common coupling and the pilot DC voltage, with the droop. The mode is used for Multi Terminal High Voltage DC (MTDC) systems where multiple HVDC Substations are connected to the HVDC transmission lines. The pilot voltage is then used to coordinate the control the DC voltage across the HVDC substations. Targets are provided by ACDCConverter.targetPpcc, ACDCConverter.targetUdc and VsConverter.droop.
phasePcc		Control is phase at point of common coupling. Target is provided by VsConverter.targetPhasePcc.

3.39 VsQpccControlKind enumeration

Kind of reactive power control at point of common coupling for a voltage source converter.
Table 16 shows all literals of VsQpccControlKind.

Table 16 – Literals of SteadyStateHypothesisScheduleProfile::VsQpccControlKind

literal	value	description
reactivePcc		Control is reactive power at point of common coupling. Target is provided by VsConverter.targetQpcc.
voltagePcc		Control is voltage at point of common coupling. Target is provided by VsConverter.targetUpcc.
powerFactorPcc		Control is power factor at point of common coupling. Target is provided by VsConverter.targetPowerFactorPcc.
pulseWidthModulation		No explicit control. Pulse-modulation factor is directly set in magnitude (VsConverter.targetPWMfactor) and phase (VsConverter.targetPhasePcc).

3.40 ActivePower datatype

Product of RMS value of the voltage and the RMS value of the in-phase component of the current.
Table 17 shows all attributes of ActivePower.

Table 17 – Attributes of SteadyStateHypothesisScheduleProfile::ActivePower

name	mult	type	description
value	0..1	Float	
multiplier	0..1	UnitMultiplier	
unit	0..1	UnitSymbol	

3.41 AngleDegrees datatype

Measurement of angle in degrees.
Table 18 shows all attributes of AngleDegrees.

Table 18 – Attributes of SteadyStateHypothesisScheduleProfile::AngleDegrees

name	mult	type	description
value	0..1	Float	
unit	0..1	UnitSymbol	
multiplier	0..1	UnitMultiplier	

3.42 ApparentPower datatype

Product of the RMS value of the voltage and the RMS value of the current.
Table 19 shows all attributes of ApparentPower.

Table 19 – Attributes of SteadyStateHypothesisScheduleProfile::ApparentPower

name	mult	type	description
value	0..1	Float	
multiplier	0..1	UnitMultiplier	
unit	0..1	UnitSymbol	

3.43 CurrentFlow datatype

Electrical current with sign convention: positive flow is out of the conducting equipment into the connectivity node. Can be both AC and DC.
Table 20 shows all attributes of CurrentFlow.

Table 20 – Attributes of SteadyStateHypothesisScheduleProfile::CurrentFlow

name	mult	type	description
value	0..1	Float	
multiplier	0..1	UnitMultiplier	
unit	0..1	UnitSymbol	

3.44 PerCent datatype

Percentage on a defined base. For example, specify as 100 to indicate at the defined base.
Table 21 shows all attributes of PerCent.

Table 21 – Attributes of SteadyStateHypothesisScheduleProfile::PerCent

name	mult	type	description
value	0..1	Float	Normally 0 to 100 on a defined base.
unit	0..1	UnitSymbol	
multiplier	0..1	UnitMultiplier	

3.45 PU datatype

Per Unit - a positive or negative value referred to a defined base. Values typically range from -10 to +10.
Table 22 shows all attributes of PU.

Table 22 – Attributes of SteadyStateHypothesisScheduleProfile::PU

name	mult	type	description
value	0..1	Float	

name	mult	type	description
unit	0..1	UnitSymbol	
multiplier	0..1	UnitMultiplier	

961

962 ### 3.46 ReactivePower datatype

963 Product of RMS value of the voltage and the RMS value of the quadrature component of the
964 current.
965 Table 23 shows all attributes of ReactivePower.

966 **Table 23 – Attributes of SteadyStateHypothesisScheduleProfile::ReactivePower**

name	mult	type	description
value	0..1	Float	
unit	0..1	UnitSymbol	
multiplier	0..1	UnitMultiplier	

967

968 ### 3.47 RealEnergy datatype

969 Real electrical energy.
970 Table 24 shows all attributes of RealEnergy.

971 **Table 24 – Attributes of SteadyStateHypothesisScheduleProfile::RealEnergy**

name	mult	type	description
multiplier	0..1	UnitMultiplier	
unit	0..1	UnitSymbol	
value	0..1	Float	

972

973 ### 3.48 Resistance datatype

974 Resistance (real part of impedance).
975 Table 25 shows all attributes of Resistance.

976 **Table 25 – Attributes of SteadyStateHypothesisScheduleProfile::Resistance**

name	mult	type	description
value	0..1	Float	
unit	0..1	UnitSymbol	
multiplier	0..1	UnitMultiplier	

977

978 ### 3.49 Voltage datatype

979 Electrical voltage, can be both AC and DC.
980 Table 26 shows all attributes of Voltage.

981 **Table 26 – Attributes of SteadyStateHypothesisScheduleProfile::Voltage**

name	mult	type	description
value	0..1	Float	
multiplier	0..1	UnitMultiplier	
unit	0..1	UnitSymbol	

982

3.50 Boolean primitive

A type with the value space "true" and "false".

3.51 DateTime primitive

Date and time as "yyyy-mm-ddThh:mm:ss.sss", which conforms with ISO 8601. UTC time zone is specified as "yyyy-mm-ddThh:mm:ss.sssZ". A local timezone relative UTC is specified as "yyyy-mm-ddThh:mm:ss.sss-hh:mm". The second component (shown here as "ss.sss") could have any number of digits in its fractional part to allow any kind of precision beyond seconds.

3.52 Float primitive

A floating point number. The range is unspecified and not limited.

3.53 Integer primitive

An integer number. The range is unspecified and not limited.

3.54 MonthDay primitive

MonthDay format as "--mm-dd", which conforms with XSD data type gMonthDay.

3.55 String primitive

A string consisting of a sequence of characters. The character encoding is UTF-8. The string length is unspecified and unlimited.

3.56 Time primitive

Time as "hh:mm:ss.sss", which conforms with ISO 8601. UTC time zone is specified as "hh:mm:ss.sssZ". A local timezone relative UTC is specified as "hh:mm:ss.sss±hh:mm". The second component (shown here as "ss.sss") could have any number of digits in its fractional part to allow any kind of precision beyond seconds.

3.57 (abstract,NC) ACDCConverterRegularSchedule

Inheritance path = BaseRegularIntervalSchedule : BaseTimeSeries : IdentifiedObject
Regular schedule for ACDC converter.
Table 27 shows all attributes of ACDCConverterRegularSchedule.

Table 27 – Attributes of SteadyStateHypothesisScheduleProfile::ACDCConverterRegularSchedule

name	mult	type	description
p	1..1	ActivePower	(NC) Active power at the point of common coupling. Load sign convention is used, i.e. positive sign means flow out from a node. Starting value for a steady state solution in the case a simplified power flow model is used.
q	1..1	ReactivePower	(NC) Reactive power at the point of common coupling. Load sign convention is used, i.e. positive sign means flow out from a node. Starting value for a steady state solution in the case a simplified power flow model is used.
targetPpcc	0..1	ActivePower	(NC) Real power injection target in AC grid, at point of common coupling. Load sign convention is used, i.e. positive sign means flow out from a node.
targetUdc	0..1	Voltage	(NC) Target value for DC voltage magnitude. The attribute shall be a positive value.
dayOfWeek	0..1	DayOfWeekKind	(NC) inherited from: BaseRegularIntervalSchedule
intervalStartTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule

name	mult	type	description
intervalEndTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 28 shows all association ends of ACDCConverterRegularSchedule with other classes.

Table 28 – Association ends of SteadyStateHypothesisScheduleProfile::ACDCConverterRegularSchedule with other classes

mult from	name	mult to	type	description
0..*	Season	0..1	Season	(NC) inherited from: BaseRegularIntervalSchedule
0..*	HourPattern	0..1	HourPattern	(NC) inherited from: BaseRegularIntervalSchedule

3.58 (abstract,NC) ACDCTimePoint root class

ACDC values for a given point in time.
Table 29 shows all attributes of ACDCTimePoint.

Table 29 – Attributes of SteadyStateHypothesisScheduleProfile::ACDCTimePoint

name	mult	type	description
atTime	1..1	DateTime	(NC) The time the data is valid for.
p	1..1	ActivePower	(NC) Active power at the point of common coupling. Load sign convention is used, i.e. positive sign means flow out from a node. Starting value for a steady state solution in the case a simplified power flow model is used.
q	1..1	ReactivePower	(NC) Reactive power at the point of common coupling. Load sign convention is used, i.e. positive sign means flow out from a node. Starting value for a steady state solution in the case a simplified power flow model is used.
targetPpcc	0..1	ActivePower	(NC) Real power injection target in AC grid, at point of common coupling. Load sign convention is used, i.e. positive sign means flow out from a node.
targetUdc	0..1	Voltage	(NC) Target value for DC voltage magnitude. The attribute shall be a positive value.

3.59 (NC) ActivePowerLimitSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Schedule for active power limit.

Table 30 shows all attributes of ActivePowerLimitSchedule.

Table 30 – Attributes of SteadyStateHypothesisScheduleProfile::ActivePowerLimitSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 31 shows all association ends of ActivePowerLimitSchedule with other classes.

Table 31 – Association ends of SteadyStateHypothesisScheduleProfile::ActivePowerLimitSchedule with other classes

mult from	name	mult to	type	description
0..*	ActivePowerLimit	1..1	ActivePowerLimit	(NC) Active power limit which has active power limit schedules.

3.60 (NC) ActivePowerLimitTimePoint root class

Active power limit for a given point in time.
Table 32 shows all attributes of ActivePowerLimitTimePoint.

Table 32 – Attributes of SteadyStateHypothesisScheduleProfile::ActivePowerLimitTimePoint

name	mult	type	description
atTime	1..1	DateTime	(NC) The time the data is valid for.
value	1..1	ActivePower	(NC) Value of active power limit. The attribute shall be a positive value or zero.

Table 33 shows all association ends of ActivePowerLimitTimePoint with other classes.

Table 33 – Association ends of SteadyStateHypothesisScheduleProfile::ActivePowerLimitTimePoint with other classes

mult from	name	mult to	type	description
1..*	ActivePowerLimitSchedule	1..1	ActivePowerLimitSchedule	(NC) The active power limit schedule that has this time point.

3.61 (NC) ApparentPowerLimitSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Schedule for apparent power limit.
Table 34 shows all attributes of ApparentPowerLimitSchedule.

Table 34 – Attributes of SteadyStateHypothesisScheduleProfile::ApparentPowerLimitSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 35 shows all association ends of ApparentPowerLimitSchedule with other classes.

Table 35 – Association ends of SteadyStateHypothesisScheduleProfile::ApparentPowerLimitSchedule with other classes

mult from	name	mult to	type	description
0..*	ApparentPowerLimit	1..1	ApparentPowerLimit	(NC) Apparent power limit which has apparent power limit schedules.

3.62 (NC) ApparentPowerLimitTimePoint root class

Apparent power limit for a given point in time.
Table 36 shows all attributes of ApparentPowerLimitTimePoint.

Table 36 – Attributes of SteadyStateHypothesisScheduleProfile::ApparentPowerLimitTimePoint

name	mult	type	description
atTime	1..1	DateTime	(NC) The time the data is valid for.
value	1..1	ApparentPower	(NC) The apparent power limit. The attribute shall be a positive value or zero.

Table 37 shows all association ends of ApparentPowerLimitTimePoint with other classes.

Table 37 – Association ends of SteadyStateHypothesisScheduleProfile::ApparentPowerLimitTimePoint with other classes

mult from	name	mult to	type	description
1..*	ApparentPowerLimitSchedule	1..1	ApparentPowerLimitSchedule	(NC) The apparent power limit schedule that has this time point.

3.63 (NC) AsynchronousMachineRegularSchedule

Inheritance path = BaseRegularIntervalSchedule : BaseTimeSeries : IdentifiedObject
Regular schedule for asynchronous machine.
Table 38 shows all attributes of AsynchronousMachineRegularSchedule.

Table 38 – Attributes of SteadyStateHypothesisScheduleProfile::AsynchronousMachineRegularSchedule

name	mult	type	description
asynchronousMachineType	1..1	AsynchronousMachineKind	(NC) Indicates the type of Asynchronous Machine (motor or generator).
dayOfWeek	0..1	DayOfWeekKind	(NC) inherited from: BaseRegularIntervalSchedule
intervalStartTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
intervalEndTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 39 shows all association ends of AsynchronousMachineRegularSchedule with other classes.

Table 39 – Association ends of SteadyStateHypothesisScheduleProfile::AsynchronousMachineRegularSchedule with other classes

mult from	name	mult to	type	description
0..*	AsynchronousMachine	1..1	AsynchronousMachine	(NC) AsynchronousMachine which has AsynchronousMachineRegularSchedule.
0..*	Season	0..1	Season	(NC) inherited from: BaseRegularIntervalSchedule
0..*	HourPattern	0..1	HourPattern	(NC) inherited from: BaseRegularIntervalSchedule

3.64 (NC) AsynchronousMachineSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Schedule for asynchronous machine.
Table 40 shows all attributes of AsynchronousMachineSchedule.

Table 40 – Attributes of SteadyStateHypothesisScheduleProfile::AsynchronousMachineSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject

name	mult	type	description
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 41 shows all association ends of AsynchronousMachineSchedule with other classes.

Table 41 – Association ends of SteadyStateHypothesisScheduleProfile::AsynchronousMachineSchedule with other classes

mult from	name	mult to	type	description
0..*	AsynchronousMachine	1..1	AsynchronousMachine	(NC) Asynchronous machine which has asynchronous machine schedules.

3.65 (NC) AsynchronousMachineTimePoint root class

Asynchronous machine values for a given point in time.
Table 42 shows all attributes of AsynchronousMachineTimePoint.

Table 42 – Attributes of SteadyStateHypothesisScheduleProfile::AsynchronousMachineTimePoint

name	mult	type	description
asynchronousMachineType	1..1	AsynchronousMachineKind	(NC) Indicates the type of Asynchronous Machine (motor or generator).
atTime	1..1	DateTime	(NC) The time the data is valid for.

Table 43 shows all association ends of AsynchronousMachineTimePoint with other classes.

Table 43 – Association ends of SteadyStateHypothesisScheduleProfile::AsynchronousMachineTimePoint with other classes

mult from	name	mult to	type	description
1..*	AsynchronousMachineSchedule	1..1	AsynchronousMachineSchedule	(NC) The asynchronous machine schedule that has this time point.

3.66 (abstract,NC) BaseIrregularTimeSeries

Inheritance path = BaseTimeSeries : IdentifiedObject
Time series that has irregular points in time.
Table 44 shows all attributes of BaseIrregularTimeSeries.

Table 44 – Attributes of SteadyStateHypothesisScheduleProfile::BaseIrregularTimeSeries

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject

name	mult	type	description
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

3.67 (abstract,NC) BaseRegularIntervalSchedule

Inheritance path = BaseTimeSeries : IdentifiedObject
Time series that has regular points in time.
Table 45 shows all attributes of BaseRegularIntervalSchedule.

Table 45 – Attributes of SteadyStateHypothesisScheduleProfile::BaseRegularIntervalSchedule

name	mult	type	description
dayOfWeek	0..1	DayOfWeekKind	(NC) Day of the week for which the schedule is valid for.
intervalStartTime	0..1	DateTime	(NC) Interval start time for which the schedule is valid for.
intervalEndTime	0..1	DateTime	(NC) Interval end time for which the schedule is valid for.
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 46 shows all association ends of BaseRegularIntervalSchedule with other classes.

Table 46 – Association ends of SteadyStateHypothesisScheduleProfile::BaseRegularIntervalSchedule with other classes

mult from	name	mult to	type	description
0..*	Season	0..1	Season	(NC) Season associated with a base regular interval schedule.
0..*	HourPattern	0..1	HourPattern	(NC) HourPattern that has base regular interval schedule.

3.68 (abstract,NC) BaseTimeSeries

Inheritance path = IdentifiedObject
Time series of values at points in time.
Table 47 shows all attributes of BaseTimeSeries.

Table 47 – Attributes of SteadyStateHypothesisScheduleProfile::BaseTimeSeries

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) Kind of interpolation done between time point.
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) Kind of base time series.

name	mult	type	description
generatedAtTime	0..1	DateTime	(NC) The time this time series (entity) come to existents and available for use.
percentile	0..1	Integer	(NC) The percentile is a number where a certain percentage of scores/ranking/values of a sample fall below that number. This is a way for expressing uncertainty in the number provided.
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

3.69 (NC) BatteryUnitSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Schedule for battery unit.
Table 48 shows all attributes of BatteryUnitSchedule.

Table 48 – Attributes of SteadyStateHypothesisScheduleProfile::BatteryUnitSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 49 shows all association ends of BatteryUnitSchedule with other classes.

Table 49 – Association ends of SteadyStateHypothesisScheduleProfile::BatteryUnitSchedule with other classes

mult from	name	mult to	type	description
0..*	BatteryUnit	1..1	BatteryUnit	(NC) Battery unit which has battery unit schedules.

3.70 (NC) BatteryUnitTimePoint root class

Battery unit values for a given point in time.
Table 50 shows all attributes of BatteryUnitTimePoint.

Table 50 – Attributes of SteadyStateHypothesisScheduleProfile::BatteryUnitTimePoint

name	mult	type	description
atTime	1..1	DateTime	(NC) The time the data is valid for.
batteryState	1..1	BatteryStateKind	(NC) The current state of the battery (charging, full, etc.).
storedE	1..1	RealEnergy	(NC) Amount of energy currently stored. The attribute shall be a positive value or zero and lower than BatteryUnit.ratedE.

Table 51 shows all association ends of BatteryUnitTimePoint with other classes.

Table 51 – Association ends of SteadyStateHypothesisScheduleProfile::BatteryUnitTimePoint with other classes

mult from	name	mult to	type	description
1..*	BatteryUnitSchedule	1..1	BatteryUnitSchedule	(NC) The battery unit schedule that has this time point.

3.71 (NC) ControlAreaRegularSchedule

Inheritance path = BaseRegularIntervalSchedule : BaseTimeSeries : IdentifiedObject
Regular schedule for control area.
Table 52 shows all attributes of ControlAreaRegularSchedule.

Table 52 – Attributes of SteadyStateHypothesisScheduleProfile::ControlAreaRegularSchedule

name	mult	type	description
netInterchange	1..1	ActivePower	(NC) The specified positive net interchange into the control area, i.e. positive sign means flow into the area.
pTolerance	0..1	ActivePower	(NC) Active power net interchange tolerance. The attribute shall be a positive value or zero.
dayOfWeek	0..1	DayOfWeekKind	(NC) inherited from: BaseRegularIntervalSchedule
intervalStartTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
intervalEndTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 53 shows all association ends of ControlAreaRegularSchedule with other classes.

Table 53 – Association ends of SteadyStateHypothesisScheduleProfile::ControlAreaRegularSchedule with other classes

mult from	name	mult to	type	description
0..*	ControlArea	1..1	ControlArea	(NC) ControlArea which has ControlAreaRegularSchedule.
0..*	Season	0..1	Season	(NC) inherited from: BaseRegularIntervalSchedule
0..*	HourPattern	0..1	HourPattern	(NC) inherited from: BaseRegularIntervalSchedule

3.72 (NC) ControlAreaSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Schedule for control area.
Table 54 shows all attributes of ControlAreaSchedule.

Table 54 – Attributes of SteadyStateHypothesisScheduleProfile::ControlAreaSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 55 shows all association ends of ControlAreaSchedule with other classes.

Table 55 – Association ends of SteadyStateHypothesisScheduleProfile::ControlAreaSchedule with other classes

mult from	name	mult to	type	description
0..*	ControlArea	1..1	ControlArea	(NC) Control area which has control area schedules.

3.73 (NC) ControlAreaTimePoint root class

Participation factor for a given point in time.
Table 56 shows all attributes of ControlAreaTimePoint.

Table 56 – Attributes of SteadyStateHypothesisScheduleProfile::ControlAreaTimePoint

name	mult	type	description
atTime	1..1	DateTime	(NC) The time the data is valid for.
netInterchange	1..1	ActivePower	(NC) The specified positive net interchange into the control area, i.e. positive sign means flow into the area.
pTolerance	0..1	ActivePower	(NC) Active power net interchange tolerance. The attribute shall be a positive value or zero.

Table 57 shows all association ends of ControlAreaTimePoint with other classes.

Table 57 – Association ends of SteadyStateHypothesisScheduleProfile::ControlAreaTimePoint with other classes

mult from	name	mult to	type	description
1..*	ControlAreaSchedule	1..1	ControlAreaSchedule	(NC) The control area schedule that has this time point.

3.74 (NC) CsConverterRegularSchedule

Inheritance path = ACDCConverterRegularSchedule : BaseRegularIntervalSchedule : BaseTimeSeries : IdentifiedObject

1179 Regular schedule for CS converter.
1180 Table 58 shows all attributes of CsConverterRegularSchedule.

1181 **Table 58 – Attributes of**
1182 **SteadyStateHypothesisScheduleProfile::CsConverterRegularSchedule**

name	mult	type	description
operatingMode	1..1	CsOperatingModeKind	(NC) Indicates whether the DC pole is operating as an inverter or as a rectifier. It is converter's control variable used in power flow.
pPccControl	1..1	CsPpccControlKind	(NC) Kind of active power control.
targetAlpha	0..1	AngleDegrees	(NC) Target firing angle. It is converter's control variable used in power flow. It is only applicable for rectifier if continuous tap changer control is used. Allowed values are within the range minAlpha<=targetAlpha<=maxAlpha. The attribute shall be a positive value.
targetGamma	0..1	AngleDegrees	(NC) Target extinction angle. It is converter's control variable used in power flow. It is only applicable for inverter if continuous tap changer control is used. Allowed values are within the range minGamma<=targetGamma<=maxGamma. The attribute shall be a positive value.
targetIdc	0..1	CurrentFlow	(NC) DC current target value. It is converter's control variable used in power flow. The attribute shall be a positive value.
p	1..1	ActivePower	(NC) inherited from: ACDCConverterRegularSchedule
q	1..1	ReactivePower	(NC) inherited from: ACDCConverterRegularSchedule
targetPpcc	0..1	ActivePower	(NC) inherited from: ACDCConverterRegularSchedule
targetUdc	0..1	Voltage	(NC) inherited from: ACDCConverterRegularSchedule
dayOfWeek	0..1	DayOfWeekKind	(NC) inherited from: BaseRegularIntervalSchedule
intervalStartTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
intervalEndTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

1183
1184 Table 59 shows all association ends of CsConverterRegularSchedule with other classes.

Table 59 – Association ends of SteadyStateHypothesisScheduleProfile::CsConverterRegularSchedule with other classes

mult from	name	mult to	type	description
0..*	CsConverter	1..1	CsConverter	(NC) CsConverter which has CsConverterRegularSchedule.
0..*	Season	0..1	Season	(NC) inherited from: BaseRegularIntervalSchedule
0..*	HourPattern	0..1	HourPattern	(NC) inherited from: BaseRegularIntervalSchedule

3.75 (NC) CsConverterSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Schedule for CS converter.
Table 60 shows all attributes of CsConverterSchedule.

Table 60 – Attributes of SteadyStateHypothesisScheduleProfile::CsConverterSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 61 shows all association ends of CsConverterSchedule with other classes.

Table 61 – Association ends of SteadyStateHypothesisScheduleProfile::CsConverterSchedule with other classes

mult from	name	mult to	type	description
0..*	CsConverter	1..1	CsConverter	(NC) Cs converter which has Cs converter schedules.

3.76 (NC) CsConverterTimePoint

Inheritance path = ACDCTimePoint
CSConverter values for a given point in time.
Table 62 shows all attributes of CsConverterTimePoint.

Table 62 – Attributes of SteadyStateHypothesisScheduleProfile::CsConverterTimePoint

name	mult	type	description
operatingMode	1..1	CsOperatingModeKind	(NC) Indicates whether the DC pole is operating as an inverter or as a rectifier. It is converter's control variable used in power flow.
pPccControl	1..1	CsPpccControlKind	(NC) Kind of active power control.

name	mult	type	description
targetAlpha	0..1	AngleDegrees	(NC) Target firing angle. It is converter's control variable used in power flow. It is only applicable for rectifier if continuous tap changer control is used. Allowed values are within the range minAlpha<=targetAlpha<=maxAlpha. The attribute shall be a positive value.
targetGamma	0..1	AngleDegrees	(NC) Target extinction angle. It is converter's control variable used in power flow. It is only applicable for inverter if continuous tap changer control is used. Allowed values are within the range minGamma<=targetGamma<=maxGamma. The attribute shall be a positive value.
targetIdc	0..1	CurrentFlow	(NC) DC current target value. It is converter's control variable used in power flow. The attribute shall be a positive value.
atTime	1..1	DateTime	(NC) inherited from: ACDCTimePoint
p	1..1	ActivePower	(NC) inherited from: ACDCTimePoint
q	1..1	ReactivePower	(NC) inherited from: ACDCTimePoint
targetPpcc	0..1	ActivePower	(NC) inherited from: ACDCTimePoint
targetUdc	0..1	Voltage	(NC) inherited from: ACDCTimePoint

Table 63 shows all association ends of CsConverterTimePoint with other classes.

Table 63 – Association ends of SteadyStateHypothesisScheduleProfile::CsConverterTimePoint with other classes

mult from	name	mult to	type	description
1..*	CsConverterSchedule	1..1	CsConverterSchedule	(NC) The CS converter schedule that has this time point.

3.77 (NC) CurrentLimitSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject

Schedule for current limit.

Table 64 shows all attributes of CurrentLimitSchedule.

Table 64 – Attributes of SteadyStateHypothesisScheduleProfile::CurrentLimitSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 65 shows all association ends of CurrentLimitSchedule with other classes.

Table 65 – Association ends of SteadyStateHypothesisScheduleProfile::CurrentLimitSchedule with other classes

mult from	name	mult to	type	description
0..*	CurrentLimit	1..1	CurrentLimit	(NC) Current limit which has current limit schedules.

3.78 (NC) CurrentLimitTimePoint root class

Current limit values for a given point in time.
Table 66 shows all attributes of CurrentLimitTimePoint.

Table 66 – Attributes of SteadyStateHypothesisScheduleProfile::CurrentLimitTimePoint

name	mult	type	description
atTime	1..1	DateTime	(NC) The time the data is valid for.
value	1..1	CurrentFlow	(NC) Limit on current flow. The attribute shall be a positive value or zero.

Table 67 shows all association ends of CurrentLimitTimePoint with other classes.

Table 67 – Association ends of SteadyStateHypothesisScheduleProfile::CurrentLimitTimePoint with other classes

mult from	name	mult to	type	description
1..*	CurrentLimitSchedule	1..1	CurrentLimitSchedule	(NC) The current limit schedule that has this time point.

3.79 (NC) EnergyConnectionRegularSchedule

Inheritance path = BaseRegularIntervalSchedule : BaseTimeSeries : IdentifiedObject
Regular schedule for energy connection.
Table 68 shows all attributes of EnergyConnectionRegularSchedule.

Table 68 – Attributes of SteadyStateHypothesisScheduleProfile::EnergyConnectionRegularSchedule

name	mult	type	description
p	1..1	ActivePower	(NC) Active power injection. Load sign convention is used, i.e. positive sign means flow out from a node. Starting value for a steady state solution.
q	1..1	ReactivePower	(NC) Reactive power injection. Load sign convention is used, i.e. positive sign means flow out from a node. Starting value for a steady state solution.
dayOfWeek	0..1	DayOfWeekKind	(NC) inherited from: BaseRegularIntervalSchedule
intervalStartTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
intervalEndTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries

name	mult	type	description
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 69 shows all association ends of EnergyConnectionRegularSchedule with other classes.

Table 69 – Association ends of SteadyStateHypothesisScheduleProfile::EnergyConnectionRegularSchedule with other classes

mult from	name	mult to	type	description
0..*	EnergyConnection	1..1	EnergyConnection	(NC) EnergyConnection which has EnergyConnectionRegularSchedule.
0..*	Season	0..1	Season	(NC) inherited from: BaseRegularIntervalSchedule
0..*	HourPattern	0..1	HourPattern	(NC) inherited from: BaseRegularIntervalSchedule

3.80 (NC) EnergyConnectionSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Schedule for energy connection.
Table 70 shows all attributes of EnergyConnectionSchedule.

Table 70 – Attributes of SteadyStateHypothesisScheduleProfile::EnergyConnectionSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 71 shows all association ends of EnergyConnectionSchedule with other classes.

Table 71 – Association ends of SteadyStateHypothesisScheduleProfile::EnergyConnectionSchedule with other classes

mult from	name	mult to	type	description
0..*	EnergyConnection	1..1	EnergyConnection	(NC) Energy connection which has energy connection schedules.

3.81 (NC) EnergyConnectionTimePoint root class

Energy connection values for a given point in time.

Table 72 shows all attributes of EnergyConnectionTimePoint.

Table 72 – Attributes of SteadyStateHypothesisScheduleProfile::EnergyConnectionTimePoint

name	mult	type	description
atTime	1..1	DateTime	(NC) The time the data is valid for.
p	1..1	ActivePower	(NC) Active power injection. Load sign convention is used, i.e. positive sign means flow out from a node. Starting value for a steady state solution.
q	1..1	ReactivePower	(NC) Reactive power injection. Load sign convention is used, i.e. positive sign means flow out from a node. Starting value for a steady state solution.

Table 73 shows all association ends of EnergyConnectionTimePoint with other classes.

Table 73 – Association ends of SteadyStateHypothesisScheduleProfile::EnergyConnectionTimePoint with other classes

mult from	name	mult to	type	description
1..*	EnergyConnectionSchedule	1..1	EnergyConnectionSchedule	(NC) The energy connection schedule that has this time point.

3.82 (NC) EquivalentInjectionRegularSchedule

Inheritance path = BaseRegularIntervalSchedule : BaseTimeSeries : IdentifiedObject
Regular schedule for equivalent injection.
Table 74 shows all attributes of EquivalentInjectionRegularSchedule.

Table 74 – Attributes of SteadyStateHypothesisScheduleProfile::EquivalentInjectionRegularSchedule

name	mult	type	description
regulationStatus	0..1	Boolean	(NC) Specifies the regulation status of the EquivalentInjection. True is regulating. False is not regulating.
regulationTarget	0..1	Voltage	(NC) The target voltage for voltage regulation. The attribute shall be a positive value.
p	1..1	ActivePower	(NC) Equivalent active power injection. Load sign convention is used, i.e. positive sign means flow out from a node. Starting value for steady state solutions.
q	1..1	ReactivePower	(NC) Equivalent reactive power injection. Load sign convention is used, i.e. positive sign means flow out from a node. Starting value for steady state solutions.
dayOfWeek	0..1	DayOfWeekKind	(NC) inherited from: BaseRegularIntervalSchedule
intervalStartTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
intervalEndTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries

name	mult	type	description
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 75 shows all association ends of EquivalentInjectionRegularSchedule with other classes.

Table 75 – Association ends of SteadyStateHypothesisScheduleProfile::EquivalentInjectionRegularSchedule with other classes

mult from	name	mult to	type	description
0..*	EquivalentInjection	1..1	EquivalentInjection	(NC) EquivalentInjection which has EquivalentInjectionRegularSchedule.
0..*	Season	0..1	Season	(NC) inherited from: BaseRegularIntervalSchedule
0..*	HourPattern	0..1	HourPattern	(NC) inherited from: BaseRegularIntervalSchedule

3.83 (NC) EquivalentInjectionSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Regular schedule for equivalent injection.
Table 76 shows all attributes of EquivalentInjectionSchedule.

Table 76 – Attributes of SteadyStateHypothesisScheduleProfile::EquivalentInjectionSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 77 shows all association ends of EquivalentInjectionSchedule with other classes.

Table 77 – Association ends of SteadyStateHypothesisScheduleProfile::EquivalentInjectionSchedule with other classes

mult from	name	mult to	type	description
0..*	EquivalentInjection	1..1	EquivalentInjection	(NC) Equivalent injection which has equivalent injection schedules.

3.84 (NC) EquivalentInjectionTimePoint root class

Equivalent injection values for a given point in time.
Table 78 shows all attributes of EquivalentInjectionTimePoint.

Table 78 – Attributes of SteadyStateHypothesisScheduleProfile::EquivalentInjectionTimePoint

name	mult	type	description
atTime	1..1	DateTime	(NC) The time the data is valid for.
regulationStatus	0..1	Boolean	(NC) Specifies the regulation status of the EquivalentInjection. True is regulating. False is not regulating.
regulationTarget	0..1	Voltage	(NC) The target voltage for voltage regulation. The attribute shall be a positive value.
p	1..1	ActivePower	(NC) Equivalent active power injection. Load sign convention is used, i.e. positive sign means flow out from a node. Starting value for steady state solutions.
q	1..1	ReactivePower	(NC) Equivalent reactive power injection. Load sign convention is used, i.e. positive sign means flow out from a node. Starting value for steady state solutions.

Table 79 shows all association ends of EquivalentInjectionTimePoint with other classes.

Table 79 – Association ends of SteadyStateHypothesisScheduleProfile::EquivalentInjectionTimePoint with other classes

mult from	name	mult to	type	description
1..*	EquivalentInjectionSchedule	1..1	EquivalentInjectionSchedule	(NC) The EquivalentInjection schedule that has this time point.

3.85 (NC) ExternalNetworkInjectionRegularSchedule

Inheritance path = BaseRegularIntervalSchedule : BaseTimeSeries : IdentifiedObject
Regular schedule for external network injection.
Table 80 shows all attributes of ExternalNetworkInjectionRegularSchedule.

Table 80 – Attributes of SteadyStateHypothesisScheduleProfile::ExternalNetworkInjectionRegularSchedule

name	mult	type	description
referencePriority	1..1	Integer	(NC) Priority of unit for use as powerflow voltage phase angle reference bus selection. 0 = don t care (default) 1 = highest priority. 2 is less than 1 and so on.
p	1..1	ActivePower	(NC) Active power injection. Load sign convention is used, i.e. positive sign means flow out from a node. Starting value for steady state solutions.
q	1..1	ReactivePower	(NC) Reactive power injection. Load sign convention is used, i.e. positive sign means flow out from a node. Starting value for steady state solutions.

name	mult	type	description
dayOfWeek	0..1	DayOfWeekKind	(NC) inherited from: BaseRegularIntervalSchedule
intervalStartTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
intervalEndTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 81 shows all association ends of ExternalNetworkInjectionRegularSchedule with other classes.

Table 81 – Association ends of SteadyStateHypothesisScheduleProfile::ExternalNetworkInjectionRegularSchedule with other classes

mult from	name	mult to	type	description
0..*	ExternalNetworkinjection	1..1	ExternalNetworkInjection	(NC) External network injection which has ExternalNetworkinjectionRegularSchedule
0..*	Season	0..1	Season	(NC) inherited from: BaseRegularIntervalSchedule
0..*	HourPattern	0..1	HourPattern	(NC) inherited from: BaseRegularIntervalSchedule

3.86 (NC) ExternalNetworkInjectionSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Schedule for external network injection.
Table 82 shows all attributes of ExternalNetworkInjectionSchedule.

Table 82 – Attributes of SteadyStateHypothesisScheduleProfile::ExternalNetworkInjectionSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 83 shows all association ends of ExternalNetworkInjectionSchedule with other classes.

Table 83 – Association ends of SteadyStateHypothesisScheduleProfile::ExternalNetworkInjectionSchedule with other classes

mult from	name	mult to	type	description
0..*	ExternalNetworkInjection	1..1	ExternalNetworkInjection	(NC) External Network Injection which has External Network Injection schedules.

3.87 (NC) ExternalNetworkInjectionTimePoint root class

External network injection values for a given point in time.
Table 84 shows all attributes of ExternalNetworkInjectionTimePoint.

Table 84 – Attributes of SteadyStateHypothesisScheduleProfile::ExternalNetworkInjectionTimePoint

name	mult	type	description
atTime	1..1	DateTime	(NC) The time the data is valid for.
referencePriority	1..1	Integer	(NC) Priority of unit for use as powerflow voltage phase angle reference bus selection. 0 = don t care (default) 1 = highest priority. 2 is less than 1 and so on.
p	1..1	ActivePower	(NC) Active power injection. Load sign convention is used, i.e. positive sign means flow out from a node. Starting value for steady state solutions.
q	1..1	ReactivePower	(NC) Reactive power injection. Load sign convention is used, i.e. positive sign means flow out from a node. Starting value for steady state solutions.

Table 85 shows all association ends of ExternalNetworkInjectionTimePoint with other classes.

Table 85 – Association ends of SteadyStateHypothesisScheduleProfile::ExternalNetworkInjectionTimePoint with other classes

mult from	name	mult to	type	description
1..*	ExternalNetworkInjectionSchedule	1..1	ExternalNetworkInjectionSchedule	(NC) The external network injection schedule that has this time point.

3.88 (abstract,NC) FuelStorage root class

Fuel storage. e.g. pile of coal that can be shared between multiple thermal generating units.

3.89 (NC) FuelStorageRegularSchedule

Inheritance path = BaseRegularIntervalSchedule : BaseTimeSeries : IdentifiedObject
Regular schedule for fuel storage.
Table 86 shows all attributes of FuelStorageRegularSchedule.

Table 86 – Attributes of SteadyStateHypothesisScheduleProfile::FuelStorageRegularSchedule

name	mult	type	description
energyStorage	1..1	RealEnergy	(NC) Amount of energy available in the storage.
dayOfWeek	0..1	DayOfWeekKind	(NC) inherited from: BaseRegularIntervalSchedule
intervalStartTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
intervalEndTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 87 shows all association ends of FuelStorageRegularSchedule with other classes.

Table 87 – Association ends of SteadyStateHypothesisScheduleProfile::FuelStorageRegularSchedule with other classes

mult from	name	mult to	type	description
0..*	FuelStorage	1..1	FuelStorage	(NC) the fule storage that has a regular schedule.
0..*	Season	0..1	Season	(NC) inherited from: BaseRegularIntervalSchedule
0..*	HourPattern	0..1	HourPattern	(NC) inherited from: BaseRegularIntervalSchedule

3.90 (NC) FuelStorageSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Fuel storage schedule for elements.
Table 88 shows all attributes of FuelStorageSchedule.

Table 88 – Attributes of SteadyStateHypothesisScheduleProfile::FuelStorageSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 89 shows all association ends of FuelStorageSchedule with other classes.

Table 89 – Association ends of SteadyStateHypothesisScheduleProfile::FuelStorageSchedule with other classes

mult from	name	mult to	type	description
0..*	FuelStorage	1..1	FuelStorage	(NC) Fuel storgae of this fuel storage schedule.

3.91 (NC) FuelStorageTimePoint root class

Fuel storage values for a given point in time.
Table 90 shows all attributes of FuelStorageTimePoint.

Table 90 – Attributes of SteadyStateHypothesisScheduleProfile::FuelStorageTimePoint

name	mult	type	description
atTime	1..1	DateTime	(NC) The time the data is valid for.
energyStorage	1..1	RealEnergy	(NC) Amount of energy available in the storage.

Table 91 shows all association ends of FuelStorageTimePoint with other classes.

Table 91 – Association ends of SteadyStateHypothesisScheduleProfile::FuelStorageTimePoint with other classes

mult from	name	mult to	type	description
1..*	FuelStorageSchedule	1..1	FuelStorageSchedule	(NC) The fuel storage schedule that has this time point.

3.92 (NC) GeneratingUnitSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Schedule for generating unit.
Table 92 shows all attributes of GeneratingUnitSchedule.

Table 92 – Attributes of SteadyStateHypothesisScheduleProfile::GeneratingUnitSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 93 shows all association ends of GeneratingUnitSchedule with other classes.

Table 93 – Association ends of SteadyStateHypothesisScheduleProfile::GeneratingUnitSchedule with other classes

mult from	name	mult to	type	description
0..*	GeneratingUnit	1..1	GeneratingUnit	(NC) Generating unit which has generating unit schedules.

3.93 (NC) GeneratingUnitTimePoint root class

Generating unit values for a given point in time.
Table 94 shows all attributes of GeneratingUnitTimePoint.

Table 94 – Attributes of SteadyStateHypothesisScheduleProfile::GeneratingUnitTimePoint

name	mult	type	description
atTime	1..1	DateTime	(NC) The time the data is valid for.
normalPF	1..1	Float	(NC) Generating unit economic participation factor. The sum of the participation factors across generating units does not have to sum to one. It is used for representing distributed slack participation factor. The attribute shall be a positive value or zero.

Table 95 shows all association ends of GeneratingUnitTimePoint with other classes.

Table 95 – Association ends of SteadyStateHypothesisScheduleProfile::GeneratingUnitTimePoint with other classes

mult from	name	mult to	type	description
1..*	GeneratingUnitSchedule	1..1	GeneratingUnitSchedule	The generating unit schedule that has this time point.

3.94 (NC) HourPattern

Inheritance path = IdentifiedObject
Pattern of hourly period in a day with the same kind of intensity.
Table 96 shows all attributes of HourPattern.

Table 96 – Attributes of SteadyStateHypothesisScheduleProfile::HourPattern

name	mult	type	description
peakKind	0..1	PeakKind	(NC) Type of peak or intensity that the pattern is valid for.
energyDemandKind	0..1	EnergyScenarioKind	(NC) Type of energy demand that the pattern is valid for.
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

3.95 (NC) HourPeriod root class

Period of hours in a day.
Table 97 shows all attributes of HourPeriod.

Table 97 – Attributes of SteadyStateHypothesisScheduleProfile::HourPeriod

name	mult	type	description
mRID	1..1	String	(NC) Master resource identifier issued by a model authority. The mRID is unique within an exchange context. Global uniqueness is easily achieved by using a UUID, as specified in RFC 4122, for the mRID. The use of UUID is strongly recommended. For CIMXML data files in RDF syntax conforming to IEC 61970-552, the mRID is mapped to rdf:ID or rdf:about attributes that identify CIM object elements.
startTime	1..1	Time	(NC) Time the period start and including, e.g. 12:00 which means it include the time of 12:00.
endTime	1..1	Time	(NC) Time the period end and not including, e.g. 13:00 which means it does not include the time of 13:00 but 12:59.

Table 98 shows all association ends of HourPeriod with other classes.

Table 98 – Association ends of SteadyStateHypothesisScheduleProfile::HourPeriod with other classes

mult from	name	mult to	type	description
1..*	HourPattern	1..1	HourPattern	(NC) HourPattern which has some hour periods.

3.96 (NC) InServiceRegularSchedule

Inheritance path = BaseRegularIntervalSchedule : BaseTimeSeries : IdentifiedObject

Regular schedule for elements having in service.

Table 99 shows all attributes of InServiceRegularSchedule.

Table 99 – Attributes of SteadyStateHypothesisScheduleProfile::InServiceRegularSchedule

name	mult	type	description
inService	1..1	Boolean	(NC) Specifies the availability of the equipment. True means the equipment is available for topology processing, which determines if the equipment is energized or not. False means that the equipment is treated by network applications as if it is not in the model.
dayOfWeek	0..1	DayOfWeekKind	(NC) inherited from: BaseRegularIntervalSchedule
intervalStartTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
intervalEndTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject

name	mult	type	description
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 100 shows all association ends of InServiceRegularSchedule with other classes.

Table 100 – Association ends of SteadyStateHypothesisScheduleProfile::InServiceRegularSchedule with other classes

mult from	name	mult to	type	description
0..*	Equipment	1..1	Equipment	(NC) Equipment which has InServiceRegularSchedule.
0..*	Season	0..1	Season	(NC) inherited from: BaseRegularIntervalSchedule
0..*	HourPattern	0..1	HourPattern	(NC) inherited from: BaseRegularIntervalSchedule

3.97 (NC) InServiceSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Schedule for elements having in service.
Table 101 shows all attributes of InServiceSchedule.

Table 101 – Attributes of SteadyStateHypothesisScheduleProfile::InServiceSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 102 shows all association ends of InServiceSchedule with other classes.

Table 102 – Association ends of SteadyStateHypothesisScheduleProfile::InServiceSchedule with other classes

mult from	name	mult to	type	description
0..*	Equipment	1..1	Equipment	(NC) Equipment which has equipment schedules.

3.98 (NC) InServiceTimePoint root class

In service values for a given point in time.
Table 103 shows all attributes of InServiceTimePoint.

Table 103 – Attributes of SteadyStateHypothesisScheduleProfile::InServiceTimePoint

name	mult	type	description
atTime	1..1	DateTime	(NC) The time the data is valid for.

name	mult	type	description
inService	1..1	Boolean	(NC) Specifies the availability of the equipment. True means the equipment is available for topology processing, which determines if the equipment is energized or not. False means that the equipment is treated by network applications as if it is not in the model.

Table 104 shows all association ends of InServiceTimePoint with other classes.

Table 104 – Association ends of SteadyStateHypothesisScheduleProfile::InServiceTimePoint with other classes

mult from	name	mult to	type	description
1..*	InServiceSchedule	1..1	InServiceSchedule	(NC) The in service schedule that has this time point.

3.99 (NC) RegulatingControlRegularSchedule

Inheritance path = BaseRegularIntervalSchedule : BaseTimeSeries : IdentifiedObject
Regular schedule for regulating control.
Table 105 shows all attributes of RegulatingControlRegularSchedule.

Table 105 – Attributes of SteadyStateHypothesisScheduleProfile::RegulatingControlRegularSchedule

name	mult	type	description
targetValue	1..1	Float	(NC) The target value specified for case input. This value can be used for the target value without the use of schedules. The value has the units appropriate to the mode attribute.
targetValueUnitMultiplier	1..1	UnitMultiplier	(NC) Specify the multiplier for used for the targetValue.
maxAllowedTargetValue	0..1	Float	(NC) Maximum allowed target value (RegulatingControl.targetValue).
minAllowedTargetValue	0..1	Float	(NC) Minimum allowed target value (RegulatingControl.targetValue).
dayOfWeek	0..1	DayOfWeekKind	(NC) inherited from: BaseRegularIntervalSchedule
intervalStartTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
intervalEndTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 106 shows all association ends of RegulatingControlRegularSchedule with other classes.

Table 106 – Association ends of SteadyStateHypothesisScheduleProfile::RegulatingControlRegularSchedule with other classes

mult from	name	mult to	type	description
0..*	RegulatingControl	1..1	RegulatingControl	(NC) Regulating control which has RegulatingControlRegularSchedule.
0..*	Season	0..1	Season	(NC) inherited from: BaseRegularIntervalSchedule
0..*	HourPattern	0..1	HourPattern	(NC) inherited from: BaseRegularIntervalSchedule

3.100 (NC) RegulatingControlSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Schedule for regulating control.
Table 107 shows all attributes of RegulatingControlSchedule.

Table 107 – Attributes of SteadyStateHypothesisScheduleProfile::RegulatingControlSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 108 shows all association ends of RegulatingControlSchedule with other classes.

Table 108 – Association ends of SteadyStateHypothesisScheduleProfile::RegulatingControlSchedule with other classes

mult from	name	mult to	type	description
0..*	RegulatingControl	1..1	RegulatingControl	(NC) Regulating control which has regulating control schedules.

3.101 (NC) RegulatingControlTimePoint root class

Regulating control values for a given point in time.
Table 109 shows all attributes of RegulatingControlTimePoint.

Table 109 – Attributes of SteadyStateHypothesisScheduleProfile::RegulatingControlTimePoint

name	mult	type	description
atTime	1..1	DateTime	(NC) The time the data is valid for.
targetValue	1..1	Float	(NC) The target value specified for case input. This value can be used for the target value

name	mult	type	description
			without the use of schedules. The value has the units appropriate to the mode attribute.
targetValueUnitMultiplier	1..1	UnitMultiplier	(NC) Specify the multiplier for used for the targetValue.
maxAllowedTargetValue	0..1	Float	(NC) Maximum allowed target value (RegulatingControl.targetValue).
minAllowedTargetValue	0..1	Float	(NC) Minimum allowed target value (RegulatingControl.targetValue).

Table 110 shows all association ends of RegulatingControlTimePoint with other classes.

Table 110 – Association ends of SteadyStateHypothesisScheduleProfile::RegulatingControlTimePoint with other classes

mult from	name	mult to	type	description
1..*	RegulatingControlSchedule	1..1	RegulatingControlSchedule	(NC) The regulating control schedule that has this time point.
1..*	TapChangerControlSchedule	1..1	TapChangerControlSchedule	(NC) The tap changer control schedule that has this time point.

3.102 (NC) ReservoirRegularSchedule

Inheritance path = BaseRegularIntervalSchedule : BaseTimeSeries : IdentifiedObject
Regular schedule for reservoir.
Table 111 shows all attributes of ReservoirRegularSchedule.

Table 111 – Attributes of SteadyStateHypothesisScheduleProfile::ReservoirRegularSchedule

name	mult	type	description
energyStorage	1..1	RealEnergy	(NC) Amount of energy available in the storage.
dayOfWeek	0..1	DayOfWeekKind	(NC) inherited from: BaseRegularIntervalSchedule
intervalStartTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
intervalEndTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 112 shows all association ends of ReservoirRegularSchedule with other classes.

Table 112 – Association ends of SteadyStateHypothesisScheduleProfile::ReservoirRegularSchedule with other classes

mult from	name	mult to	type	description
0..*	Reservoir	1..1	Reservoir	(NC) The reservoir that has this regular schedule.
0..*	Season	0..1	Season	(NC) inherited from: BaseRegularIntervalSchedule
0..*	HourPattern	0..1	HourPattern	(NC) inherited from: BaseRegularIntervalSchedule

3.103 (NC) ReservoirSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Reservoir schedule for elements.
Table 113 shows all attributes of ReservoirSchedule.

Table 113 – Attributes of SteadyStateHypothesisScheduleProfile::ReservoirSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 114 shows all association ends of ReservoirSchedule with other classes.

Table 114 – Association ends of SteadyStateHypothesisScheduleProfile::ReservoirSchedule with other classes

mult from	name	mult to	type	description
0..*	Reservoir	1..1	Reservoir	(NC) The reservoir that has a schedule.

3.104 (NC) ReservoirTimePoint root class

Reservoir values for a given point in time.
Table 115 shows all attributes of ReservoirTimePoint.

Table 115 – Attributes of SteadyStateHypothesisScheduleProfile::ReservoirTimePoint

name	mult	type	description
atTime	1..1	DateTime	(NC) The time the data is valid for.
energyStorage	1..1	RealEnergy	(NC) Amount of energy available in the storage.

Table 116 shows all association ends of ReservoirTimePoint with other classes.

Table 116 – Association ends of SteadyStateHypothesisScheduleProfile::ReservoirTimePoint with other classes

mult from	name	mult to	type	description
1..*	ReservoirSchedule	1..1	ReservoirSchedule	(NC) The reservoir schedule that has this time point.

3.105 (NC) ShuntCompensatorSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Schedule for shunt compensator.
Table 117 shows all attributes of ShuntCompensatorSchedule.

Table 117 – Attributes of SteadyStateHypothesisScheduleProfile::ShuntCompensatorSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 118 shows all association ends of ShuntCompensatorSchedule with other classes.

Table 118 – Association ends of SteadyStateHypothesisScheduleProfile::ShuntCompensatorSchedule with other classes

mult from	name	mult to	type	description
0..*	ShuntCompensator	1..1	ShuntCompensator	(NC) Shunt compensator which has shunt compensator schedules.

3.106 (NC) ShuntCompensatorTimePoint root class

Shunt compensator values for a given point in time.
Table 119 shows all attributes of ShuntCompensatorTimePoint.

Table 119 – Attributes of SteadyStateHypothesisScheduleProfile::ShuntCompensatorTimePoint

name	mult	type	description
atTime	1..1	DateTime	(NC) The time the data is valid for.
sections	1..1	Float	(NC) Shunt compensator sections in use. Starting value for steady state solution. The attribute shall be a positive value or zero. Non integer values are allowed to support continuous variables. The reasons for continuous value are to support study cases where no discrete shunt compensators has yet been designed, a solutions where a narrow voltage band force the sections to oscillate or accommodate for a continuous solution as input.

name	mult	type	description
			For LinearShuntConpensator the value shall be between zero and ShuntCompensator.maximumSections. At value zero the shunt compensator conductance and admittance is zero. Linear interpolation of conductance and admittance between the previous and next integer section is applied in case of non-integer values. For NonlinearShuntCompensator-s shall only be set to one of the NonlinearShuntCompenstorPoint.sectionNumber. There is no interpolation between NonlinearShuntCompenstorPoint-s.

Table 120 shows all association ends of ShuntCompensatorTimePoint with other classes.

Table 120 – Association ends of SteadyStateHypothesisScheduleProfile::ShuntCompensatorTimePoint with other classes

mult from	name	mult to	type	description
1..*	ShuntCompensatorSchedule	1..1	ShuntCompensatorSchedule	(NC) The shunt compensator schedule that has this time point.

3.107 (NC) StaticVarCompensatorSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Schedule for static var compensator.
Table 121 shows all attributes of StaticVarCompensatorSchedule.

Table 121 – Attributes of SteadyStateHypothesisScheduleProfile::StaticVarCompensatorSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 122 shows all association ends of StaticVarCompensatorSchedule with other classes.

Table 122 – Association ends of SteadyStateHypothesisScheduleProfile::StaticVarCompensatorSchedule with other classes

mult from	name	mult to	type	description
0..*	StaticVarCompensator	1..1	StaticVarCompensator	(NC) Static var compensator which has static var compensator schedules.

3.108 (NC) StaticVarCompensatorTimePoint root class

Static var compensator values for a given point in time.
Table 123 shows all attributes of StaticVarCompensatorTimePoint.

Table 123 – Attributes of SteadyStateHypothesisScheduleProfile::StaticVarCompensatorTimePoint

name	mult	type	description
atTime	1..1	DateTime	(NC) The time the data is valid for.
q	1..1	ReactivePower	(NC) Reactive power injection. Load sign convention is used, i.e. positive sign means flow out from a node. Starting value for a steady state solution.

Table 124 shows all association ends of StaticVarCompensatorTimePoint with other classes.

Table 124 – Association ends of SteadyStateHypothesisScheduleProfile::StaticVarCompensatorTimePoint with other classes

mult from	name	mult to	type	description
1..*	StaticVarCompensatorSchedule	1..1	StaticVarCompensatorSchedule	(NC) The StaticVarCompensator schedule that has this time point.

3.109 (NC) SwitchRegularSchedule

Inheritance path = BaseRegularIntervalSchedule : BaseTimeSeries : IdentifiedObject
Regular schedule for switch.
Table 125 shows all attributes of SwitchRegularSchedule.

Table 125 – Attributes of SteadyStateHypothesisScheduleProfile::SwitchRegularSchedule

name	mult	type	description
open	1..1	Boolean	(NC) The attribute tells if the switch is considered open when used as input to topology processing.
locked	1..1	Boolean	(NC) If true, the switch is locked. The resulting switch state is a combination of locked and Switch.open attributes as follows: - locked=true and Switch.open=true. The resulting state is open and locked; - locked=false and Switch.open=true. The resulting state is open; - locked=false and Switch.open=false. The resulting state is closed.
dayOfWeek	0..1	DayOfWeekKind	(NC) inherited from: BaseRegularIntervalSchedule
intervalStartTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
intervalEndTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries

name	mult	type	description
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 126 shows all association ends of SwitchRegularSchedule with other classes.

Table 126 – Association ends of SteadyStateHypothesisScheduleProfile::SwitchRegularSchedule with other classes

mult from	name	mult to	type	description
0..*	Switch	1..1	Switch	(NC) Switch which has SwitchRegularSchedule.
0..*	Season	0..1	Season	(NC) inherited from: BaseRegularIntervalSchedule
0..*	HourPattern	0..1	HourPattern	(NC) inherited from: BaseRegularIntervalSchedule

3.110 (NC) SwitchSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Schedule for switch.
Table 127 shows all attributes of SwitchSchedule.

Table 127 – Attributes of SteadyStateHypothesisScheduleProfile::SwitchSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 128 shows all association ends of SwitchSchedule with other classes.

Table 128 – Association ends of SteadyStateHypothesisScheduleProfile::SwitchSchedule with other classes

mult from	name	mult to	type	description
0..*	Switch	1..1	Switch	(NC) Switch which has switch schedules.

3.111 (NC) SwitchTimePoint root class

Switch values for a given point in time.
Table 129 shows all attributes of SwitchTimePoint.

1562 **Table 129 – Attributes of SteadyStateHypothesisScheduleProfile::SwitchTimePoint**

name	mult	type	description
atTime	1..1	DateTime	(NC) The time the data is valid for.
open	1..1	Boolean	(NC) The attribute tells if the switch is considered open when used as input to topology processing.
locked	1..1	Boolean	(NC) If true, the switch is locked. The resulting switch state is a combination of locked and Switch.open attributes as follows: - locked=true and Switch.open=true. The resulting state is open and locked; - locked=false and Switch.open=true. The resulting state is open; - locked=false and Switch.open=false. The resulting state is closed.

1563
1564 Table 130 shows all association ends of SwitchTimePoint with other classes.

1565 **Table 130 – Association ends of**
1566 **SteadyStateHypothesisScheduleProfile::SwitchTimePoint with other classes**

mult from	name	mult to	type	description
1..*	SwitchSchedule	1..1	SwitchSchedule	The switch schedule that has this time point.

1567

1568 ### 3.112 (NC) SynchronousMachineRegularSchedule

1569 Inheritance path = BaseRegularIntervalSchedule : BaseTimeSeries : IdentifiedObject
1570 Regular schedule for synchronous machine.
1571 Table 131 shows all attributes of SynchronousMachineRegularSchedule.

1572 **Table 131 – Attributes of**
1573 **SteadyStateHypothesisScheduleProfile::SynchronousMachineRegularSchedule**

name	mult	type	description
operatingMode	1..1	SynchronousMachineOperatingMode	(NC) Current mode of operation.
referencePriority	1..1	Integer	(NC) Priority of unit for use as powerflow voltage phase angle reference bus selection. 0 = don t care (default) 1 = highest priority. 2 is less than 1 and so on.
dayOfWeek	0..1	DayOfWeekKind	(NC) inherited from: BaseRegularIntervalSchedule
intervalStartTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
intervalEndTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject

name	mult	type	description
name	1..1	String	inherited from: IdentifiedObject

Table 132 shows all association ends of SynchronousMachineRegularSchedule with other classes.

Table 132 – Association ends of SteadyStateHypothesisScheduleProfile::SynchronousMachineRegularSchedule with other classes

mult from	name	mult to	type	description
0..*	SynchronousMachine	1..1	SynchronousMachine	(NC) SynchronousMachine which has SynchronousMachineRegularSchedule.
0..*	Season	0..1	Season	(NC) inherited from: BaseRegularIntervalSchedule
0..*	HourPattern	0..1	HourPattern	(NC) inherited from: BaseRegularIntervalSchedule

3.113 (NC) SynchronousMachineSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Schedule for synchronous machine.
Table 133 shows all attributes of SynchronousMachineSchedule.

Table 133 – Attributes of SteadyStateHypothesisScheduleProfile::SynchronousMachineSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 134 shows all association ends of SynchronousMachineSchedule with other classes.

Table 134 – Association ends of SteadyStateHypothesisScheduleProfile::SynchronousMachineSchedule with other classes

mult from	name	mult to	type	description
0..*	SynchronousMachine	1..1	SynchronousMachine	(NC) Synchronous machine which has synchronous machine schedules.

3.114 (NC) SynchronousMachineTimePoint root class

Synchronous machine values for a given point in time.
Table 135 shows all attributes of SynchronousMachineTimePoint.

Table 135 – Attributes of SteadyStateHypothesisScheduleProfile::SynchronousMachineTimePoint

name	mult	type	description
atTime	1..1	DateTime	(NC) The time the data is valid for.
operatingMode	1..1	SynchronousMachineOperatingMode	(NC) Current mode of operation.
referencePriority	1..1	Integer	(NC) Priority of unit for use as powerflow voltage phase angle reference bus selection. 0 = don t care (default) 1 = highest priority. 2 is less than 1 and so on.

Table 136 shows all association ends of SynchronousMachineTimePoint with other classes.

Table 136 – Association ends of SteadyStateHypothesisScheduleProfile::SynchronousMachineTimePoint with other classes

mult from	name	mult to	type	description
1..*	SynchronousMachineSchedule	1..1	SynchronousMachineSchedule	(NC) The synchronous machine schedule that has this time point.

3.115 (NC) TapChangerControlRegularSchedule

Inheritance path = RegulatingControlRegularSchedule : BaseRegularIntervalSchedule : BaseTimeSeries : IdentifiedObject

Regular schedule for tap changer control.

Table 137 shows all attributes of TapChangerControlRegularSchedule.

Table 137 – Attributes of SteadyStateHypothesisScheduleProfile::TapChangerControlRegularSchedule

name	mult	type	description
targetValue	1..1	Float	(NC) inherited from: RegulatingControlRegularSchedule
targetValueUnitMultiplier	1..1	UnitMultiplier	(NC) inherited from: RegulatingControlRegularSchedule
maxAllowedTargetValue	0..1	Float	(NC) inherited from: RegulatingControlRegularSchedule
minAllowedTargetValue	0..1	Float	(NC) inherited from: RegulatingControlRegularSchedule
dayOfWeek	0..1	DayOfWeekKind	(NC) inherited from: BaseRegularIntervalSchedule
intervalStartTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
intervalEndTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject

name	mult	type	description
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 138 shows all association ends of TapChangerControlRegularSchedule with other classes.

Table 138 – Association ends of SteadyStateHypothesisScheduleProfile::TapChangerControlRegularSchedule with other classes

mult from	name	mult to	type	description
0..*	TapChangerControl	1..1	TapChangerControl	(NC) Tap changer control which has TapChangerControlRegularSchedule.
0..*	RegulatingControl	1..1	RegulatingControl	(NC) inherited from: RegulatingControlRegularSchedule
0..*	Season	0..1	Season	(NC) inherited from: BaseRegularIntervalSchedule
0..*	HourPattern	0..1	HourPattern	(NC) inherited from: BaseRegularIntervalSchedule

3.116 (NC) TapChangerControlSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Schedule for tap changer control.
Table 139 shows all attributes of TapChangerControlSchedule.

Table 139 – Attributes of SteadyStateHypothesisScheduleProfile::TapChangerControlSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 140 shows all association ends of TapChangerControlSchedule with other classes.

Table 140 – Association ends of SteadyStateHypothesisScheduleProfile::TapChangerControlSchedule with other classes

mult from	name	mult to	type	description
0..*	TapChangerControl	1..1	TapChangerControl	(NC) Tap changer control which has TapChangerControlSchedule.

3.117 (NC) TapRegularSchedule

Inheritance path = BaseRegularIntervalSchedule : BaseTimeSeries : IdentifiedObject

Regular schedule for tap.
Table 141 shows all attributes of TapRegularSchedule.

Table 141 – Attributes of SteadyStateHypothesisScheduleProfile::TapRegularSchedule

name	mult	type	description
controlEnabled	1..1	Boolean	(NC) Specifies the regulation status of the equipment. True is regulating, false is not regulating.
step	1..1	Float	(NC) Tap changer position. Starting step for a steady state solution. Non integer values are allowed to support continuous tap variables. The reasons for continuous value are to support study cases where no discrete tap changer has yet been designed, a solution where a narrow voltage band forces the tap step to oscillate or to accommodate for a continuous solution as input. The attribute shall be equal to or greater than lowStep and equal to or less than highStep.
dayOfWeek	0..1	DayOfWeekKind	(NC) inherited from: BaseRegularIntervalSchedule
intervalStartTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
intervalEndTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 142 shows all association ends of TapRegularSchedule with other classes.

Table 142 – Association ends of SteadyStateHypothesisScheduleProfile::TapRegularSchedule with other classes

mult from	name	mult to	type	description
0..*	TapChanger	1..1	TapChanger	(NC) Tap changer which has TapRegularSchedule.
0..*	Season	0..1	Season	(NC) inherited from: BaseRegularIntervalSchedule
0..*	HourPattern	0..1	HourPattern	(NC) inherited from: BaseRegularIntervalSchedule

3.118 (NC) TapSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Schedule for tap.
Table 143 shows all attributes of TapSchedule.

Table 143 – Attributes of SteadyStateHypothesisScheduleProfile::TapSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 144 shows all association ends of TapSchedule with other classes.

Table 144 – Association ends of SteadyStateHypothesisScheduleProfile::TapSchedule with other classes

mult from	name	mult to	type	description
0..*	TapChanger	1..1	TapChanger	(NC) Tap changer which has tap schedules.

3.119 (NC) TapScheduleTimePoint root class

Tap schedule values for a given point in time.
Table 145 shows all attributes of TapScheduleTimePoint.

Table 145 – Attributes of SteadyStateHypothesisScheduleProfile::TapScheduleTimePoint

name	mult	type	description
atTime	1..1	DateTime	(NC) The time the data is valid for.
controlEnabled	1..1	Boolean	(NC) Specifies the regulation status of the equipment. True is regulating, false is not regulating.
step	1..1	Float	(NC) Tap changer position. Starting step for a steady state solution. Non integer values are allowed to support continuous tap variables. The reasons for continuous value are to support study cases where no discrete tap changer has yet been designed, a solution where a narrow voltage band forces the tap step to oscillate or to accommodate for a continuous solution as input. The attribute shall be equal to or greater than lowStep and equal to or less than highStep.

Table 146 shows all association ends of TapScheduleTimePoint with other classes.

Table 146 – Association ends of SteadyStateHypothesisScheduleProfile::TapScheduleTimePoint with other classes

mult from	name	mult to	type	description
1..*	TapSchedule	1..1	TapSchedule	(NC) The tap schedule that has this time point.

3.120 (NC) VoltageLimitSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Schedule for voltage limit.
Table 147 shows all attributes of VoltageLimitSchedule.

Table 147 – Attributes of SteadyStateHypothesisScheduleProfile::VoltageLimitSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 148 shows all association ends of VoltageLimitSchedule with other classes.

Table 148 – Association ends of SteadyStateHypothesisScheduleProfile::VoltageLimitSchedule with other classes

mult from	name	mult to	type	description
0..*	VoltageLimit	1..1	VoltageLimit	(NC) Voltage limit which has voltage limit schedules.

3.121 (NC) VoltageLimitTimePoint root class

Voltage limit values for a given point in time.
Table 149 shows all attributes of VoltageLimitTimePoint.

Table 149 – Attributes of SteadyStateHypothesisScheduleProfile::VoltageLimitTimePoint

name	mult	type	description
atTime	1..1	DateTime	(NC) The time the data is valid for.
value	1..1	Voltage	(NC) Limit on voltage. High or low limit nature of the limit depends upon the properties of the operational limit type. The attribute shall be a positive value or zero.

Table 150 shows all association ends of VoltageLimitTimePoint with other classes.

Table 150 – Association ends of SteadyStateHypothesisScheduleProfile::VoltageLimitTimePoint with other classes

mult from	name	mult to	type	description
1..*	VoltageLimitSchedule	1..1	VoltageLimitSchedule	(NC) The voltage limit schedule that has this time point.

3.122 (NC) VsConverterRegularSchedule

Inheritance path = ACDCConverterRegularSchedule : BaseRegularIntervalSchedule : BaseTimeSeries : IdentifiedObject

1683 Regular schedule for VS converter.
1684 Table 151 shows all attributes of VsConverterRegularSchedule.

1685 **Table 151 – Attributes of**
1686 **SteadyStateHypothesisScheduleProfile::VsConverterRegularSchedule**

name	mult	type	description
droop	0..1	PU	Droop constant. The pu value is obtained as D [kV/MW] x Sb / Ubdc. The attribute shall be a positive value.
droopCompensation	0..1	Resistance	Compensation constant. Used to compensate for voltage drop when controlling voltage at a distant bus. The attribute shall be a positive value.
pPccControl	1..1	VsPpccControlKind	Kind of control of real power and/or DC voltage.
qPccControl	1..1	VsQpccControlKind	Kind of reactive power control.
qShare	0..1	PerCent	Reactive power sharing factor among parallel converters on Uac control. The attribute shall be a positive value or zero.
targetQpcc	0..1	ReactivePower	Reactive power injection target in AC grid, at point of common coupling. Load sign convention is used, i.e. positive sign means flow out from a node.
targetUpcc	0..1	Voltage	Voltage target in AC grid, at point of common coupling. The attribute shall be a positive value.
targetPowerFactorPcc	0..1	Float	Power factor target at the AC side, at point of common coupling. The attribute shall be a positive value.
targetPhasePcc	0..1	AngleDegrees	Phase target at AC side, at point of common coupling. The attribute shall be a positive value.
targetPWMfactor	0..1	Float	Magnitude of pulse-modulation factor. The attribute shall be a positive value.
p	1..1	ActivePower	(NC) inherited from: ACDCConverterRegularSchedule
q	1..1	ReactivePower	(NC) inherited from: ACDCConverterRegularSchedule
targetPpcc	0..1	ActivePower	(NC) inherited from: ACDCConverterRegularSchedule
targetUdc	0..1	Voltage	(NC) inherited from: ACDCConverterRegularSchedule
dayOfWeek	0..1	DayOfWeekKind	(NC) inherited from: BaseRegularIntervalSchedule
intervalStartTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
intervalEndTime	0..1	DateTime	(NC) inherited from: BaseRegularIntervalSchedule
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject

name	mult	type	description
name	1..1	String	inherited from: IdentifiedObject

Table 152 shows all association ends of VsConverterRegularSchedule with other classes.

Table 152 – Association ends of SteadyStateHypothesisScheduleProfile::VsConverterRegularSchedule with other classes

mult from	name	mult to	type	description
0..*	VsConverter	1..1	VsConverter	(NC) VsConverter which has VsConverterRegularSchedule.
0..*	Season	0..1	Season	(NC) inherited from: BaseRegularIntervalSchedule
0..*	HourPattern	0..1	HourPattern	(NC) inherited from: BaseRegularIntervalSchedule

3.123 (NC) VsConverterSchedule

Inheritance path = BaseIrregularTimeSeries : BaseTimeSeries : IdentifiedObject
Schedule for VS converter.
Table 153 shows all attributes of VsConverterSchedule.

Table 153 – Attributes of SteadyStateHypothesisScheduleProfile::VsConverterSchedule

name	mult	type	description
interpolationKind	1..1	TimeSeriesInterpolationKind	(NC) inherited from: BaseTimeSeries
timeSeriesKind	0..1	BaseTimeSeriesKind	(NC) inherited from: BaseTimeSeries
generatedAtTime	0..1	DateTime	(NC) inherited from: BaseTimeSeries
percentile	0..1	Integer	(NC) inherited from: BaseTimeSeries
description	0..1	String	inherited from: IdentifiedObject
mRID	1..1	String	inherited from: IdentifiedObject
name	1..1	String	inherited from: IdentifiedObject

Table 154 shows all association ends of VsConverterSchedule with other classes.

Table 154 – Association ends of SteadyStateHypothesisScheduleProfile::VsConverterSchedule with other classes

mult from	name	mult to	type	description
0..*	VsConverter	1..1	VsConverter	(NC) Vs converter which has Vs converter schedules.

3.124 (NC) VsConverterTimePoint

Inheritance path = ACDCTimePoint
VS converter values for a given point in time.
Table 155 shows all attributes of VsConverterTimePoint.

Table 155 – Attributes of SteadyStateHypothesisScheduleProfile::VsConverterTimePoint

name	mult	type	description
droop	0..1	PU	(NC) Droop constant. The pu value is obtained as D [kV/MW] x Sb / Ubdc. The attribute shall be a positive value.
droopCompensation	0..1	Resistance	(NC) Compensation constant. Used to compensate for voltage drop when controlling voltage at a distant bus. The attribute shall be a positive value.
pPccControl	1..1	VsPpccControlKind	(NC) Kind of control of real power and/or DC voltage.
qPccControl	1..1	VsQpccControlKind	(NC) Kind of reactive power control.
qShare	0..1	PerCent	(NC) Reactive power sharing factor among parallel converters on Uac control. The attribute shall be a positive value or zero.
targetQpcc	0..1	ReactivePower	(NC) Reactive power injection target in AC grid, at point of common coupling. Load sign convention is used, i.e. positive sign means flow out from a node.
targetUpcc	0..1	Voltage	(NC) Voltage target in AC grid, at point of common coupling. The attribute shall be a positive value.
targetPowerFactorPcc	0..1	Float	(NC) Power factor target at the AC side, at point of common coupling. The attribute shall be a positive value.
targetPhasePcc	0..1	AngleDegrees	(NC) Phase target at AC side, at point of common coupling. The attribute shall be a positive value.
targetPWMfactor	0..1	Float	(NC) Magnitude of pulse-modulation factor. The attribute shall be a positive value.
atTime	1..1	DateTime	(NC) inherited from: ACDCTimePoint
p	1..1	ActivePower	(NC) inherited from: ACDCTimePoint
q	1..1	ReactivePower	(NC) inherited from: ACDCTimePoint
targetPpcc	0..1	ActivePower	(NC) inherited from: ACDCTimePoint
targetUdc	0..1	Voltage	(NC) inherited from: ACDCTimePoint

Table 156 shows all association ends of VsConverterTimePoint with other classes.

Table 156 – Association ends of SteadyStateHypothesisScheduleProfile::VsConverterTimePoint with other classes

mult from	name	mult to	type	description
1..*	VsConverterSchedule	1..1	VsConverterSchedule	(NC) The VS converter schedule that has this time point.

Annex A (informative): Sample data

A.1 General

This Annex is designed to illustrate the profile by using fragments of sample data. It is not meant to be a complete set of examples covering all possibilities of using the profile. Defining a complete set of test data is considered a separate activity to be performed for the purpose of setting up interoperability testing and conformity related to this profile.

A.2 Sample instance data

Test data files are available in the CIM EG SharePoint.