IEC TR 61850-90-10

Edition 1.0 2017-10

TECHNICAL REPORT

colour inside

Communication networks and systems for power utility automation – Part 90-10: Models for scheduling

IEC TR 61850-90-10:2017-10(en)

IEC Central Office
3, rue de Varembé
CH-1211 Geneva 20
Switzerland

Tel.: +41 22 919 02 11
Fax: +41 22 919 03 00
info@iec.ch
www.iec.ch

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

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

IEC Catalogue - webstore.iec.ch/catalogue
The stand-alone application for consulting the entire bibliographical information on IEC International Standards, Technical Specifications, Technical Reports and other documents. Available for PC, Mac OS, Android Tablets and iPad.

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 It also gives information on projects, replaced and withdrawn publications.

IEC Just Published - webstore.iec.ch/justpublished
Stay up to date on all new IEC publications. Just Published details all new publications released. Available online and also once a month by email.

Electropedia - www.electropedia.org
The world's leading online dictionary of electronic and electrical terms containing 20 000 terms and definitions in English and French, with equivalent terms in 16 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

IEC Glossary - std.iec.ch/glossary
65 000 electrotechnical terminology entries in English and French extracted from the Terms and Definitions clause of IEC publications issued since 2002. Some entries have been collected from earlier publications of IEC TC 37, 77, 86 and CISPR.

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

IEC TR 61850-90-10

Edition 1.0 2017-10

TECHNICAL REPORT

colour inside

Communication networks and systems for power utility automation – Part 90-10: Models for scheduling

INTERNATIONAL ELECTROTECHNICAL COMMISSION

ICS 33.200

ISBN 978-2-8322-4917-8

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

_____

COMMUNICATION NETWORKS AND SYSTEMS FOR POWER UTILITY AUTOMATION –

Part 90-10: Models for scheduling

FOREWORD

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

The main task of IEC technical committees is to prepare International Standards. However, a technical committee may propose the publication of a technical report when it has collected data of a different kind from that which is normally published as an International Standard, for example "state of the art".

IEC TR 61850-90-10, which is a technical report, has been prepared by IEC technical committee 57: Power systems management and associated information exchange.

The text of this technical report is based on the following documents:

Enquiry draft	Report on voting
57/1762/DTR	57/1902/RVDTR

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

This document has been drafted in accordance with the ISO/IEC Directives, Part 2.

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

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

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

A bilingual version of this publication may be issued at a later date.

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

COMMUNICATION NETWORKS AND SYSTEMS FOR POWER UTILITY AUTOMATION –

Part 90-10: Models for scheduling

1 Scope

This part of IEC 61850, which is a Technical Report, describes scheduling for devices using IEC 61850.

The parameters, which identify this new namespace, are:

- Namespace Version: 2017
- Namespace Revision: A
- UML model file which reflects this namespace edition: wg10uml02v17-wg18uml02v11b-wg17uml02v18-jwg25uml02v04c.eap, UML model version WG17UML02v18
- Namespace release date: 2017-06-12
- Namespace name: "(Tr)IEC61850-90-10:2017A"

The namespace "(Tr)IEC61850-90-10:2017A" is considered as "transitional" since the models are expected to be included in the next editions of IEC 61850-7-4xx and IEC 61850-7-3. Potential extensions/modifications may happen if/when the models are moved to International Standard status. Only the new data objects and CDCs that are not said inherited from existing LNs will be tagged with this namespace name. The others should still refer to the namespace where they are primarily defined.

2 Normative references

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

IEC TS 61850-2, *Communication networks and systems in substations – Part 2: Glossary*

IEC 61850-7-1:2011, *Communication networks and systems for power utility automation – Part 7-1: Basic communication structure – Principles and models*

IEC 61850-7-2:2010, *Communication networks and systems for power utility automation – Part 7-2: Basic information and communication structure – Abstract communication service interface (ACSI)*

IEC 61850-7-3:2010, *Communication networks and systems for power utility automation – Part 7-3: Basic communication structure – Common data classes*

IEC 61850-7-4:2010, *Communication networks and systems for power utility automation – Part 7-4: Basic communication structure – Compatible logical node classes and data object classes*

3 Terms, definitions and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC TS 61850-2 IEC 61850-7-2 and the following apply.

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

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

3.1.1 scheduled entity

data object of one of the following common data classes APC, ASG, INS, ING SPC, SPG, ENC or ENG where the control output or the value of the setting may be determined by the scheduling system

3.1.2 scheduling system

collection containing a schedule controller and the schedules to which the schedule controller refers

Note 1 to entry: The scheduling system is associated to a scheduled entity (by reference in the schedule controller) and determines the behaviour of the scheduled entity.

3.2 Abbreviated terms

Clause 4 of IEC 61850-7-4:2010 defines abbreviated terms for building concatenated data names.

For the purposes of this document, some new abbreviations have been introduced as presented in this table.

Table 1 shows normative terms that are combined to create data object names.

Table 1 – Normative abbreviations for data object names

Term	Description
Dsa	Disable, disabled
Entr	Entry, entries
Prio	Priority
Req	Requested
Reuse	Reuse
Schd	Schedule

NOTE Abbreviated terms used for the identification of the common data classes and as names of the attributes are specified in the relevant clauses of this document and are not repeated here.

4 Requirements for schedules

4.1 Schedules for establishing time-based behaviour

Schedules establish which behaviour (for example, tariff 1 or 2, mode 1 or 3) is applied during specified time periods. A schedule consists of a series of entries. Entries of a schedule represent the value of a setpoint, the activation value of a particular mode or the value of a parameter for a mode. All the entries of a single schedule refer to a same data object which is

a controllable object, a setpoint or a setting and which is called "scheduled entity" in this document.

One of the most common uses of schedules is for scheduling real power output. In that case, DER systems that directly manage their power output, such as reciprocating engines and energy storage systems, may use schedules to determine the time and level of power output.

However, many other controllable entities may also be scheduled. Larger inverter-based DER systems and large aggregations of small inverter-based DER systems have significant influence on the distribution system and have local Volt-VAr characteristics that may vary throughout the day. As a result, a single function or operational mode such as a specific Volt-VAr curve may not be suitable at all times. Yet sending many control commands every few hours to many different DER systems may impact bandwidth-limited communications systems or may not be received in a timely manner, leading to inadequate DER system responses. However, if schedules are established that the DER systems will follow autonomously, then these communication impacts will be minimized.

Schedules may be used to allow even more autonomous control of the behaviour of DER equipment. They are usually sent ahead of time, and then activated at the appropriate time or by an external trigger event. There are different applications for schedules:

- schedules for direct values (e.g. setpoints for P/Q/cosphi);
- schedules for constraints (e.g. do not exceed the maximum value of P/Q/cosphi at a certain time);
- schedules for pricing information (act on constraints);
- schedules for activation of modes (e.g. Volt-VAr curves);
- schedules for parameters of modes.

4.2 Relation between schedules and directly set values and parameters of the DER system

As has been mentioned before, schedules are an alternate way to determine the behaviour compared to direct control possibilities (i.e. using setpoints or activate and configure a selected mode of operation). Therefore, it is important to describe how schedules and direct control possibilities interact.

5 Principles for modelling schedules

5.1 Schedule controllers and their schedules

Within an electrical system, multiple entities may be scheduled in parallel. For example, it is possible to schedule active power and reactive power. For each of the scheduled entities (see 4.1 and 6.1), there will be, as depicted in Figure 1, at most one schedule controller and potentially several schedules referenced (used) by the schedule controller. One given schedule may be referenced by multiple schedule controllers.

An entity is a data object that exists in the IEC 61850 data model which is either a data object supporting control services (e.g. binary control or setpoint) or it is a setting (data object that includes attributes of FC=SP). The schedule controller has a data object which is a reference to the scheduled data object (see 6.3). For instance, a schedule controller could be associated with WMax (setting of the maximum real power output of a DER – namespace "(TR) IEC 61850-90-7:2012"). Another schedule controller could be associated with OpModVVAr which enables/disables the volt/var mode (namespace "(TR) IEC 61850-90-7:2012").

The description in this subclause applies to one scheduled entity (i.e. one schedule controller and multiple schedules).

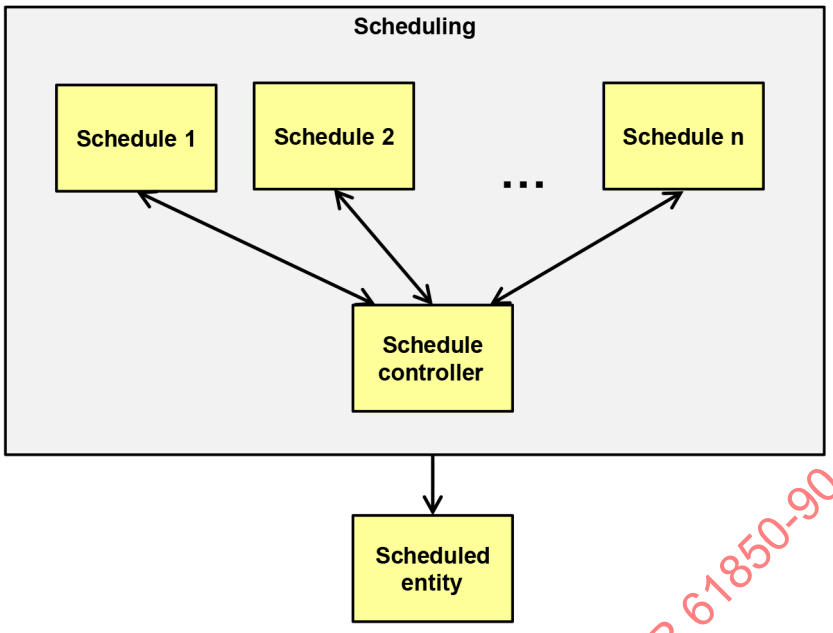


Figure 1 – Scheduling principle

5.2 Scheduling and direct control / setting

5.2.1 General

As mentioned in 5.1, scheduling applies to scheduled entities which are data objects in the IEC 61850 data model which may be controlled through control services (controllable data object) or which are settings where the value may be written.

5.2.2 Case when the scheduled entity is of type "controllable data object"

For controllable data objects a new data object will be added in IEC 61850-7-4[1] for each logical node which will indicate if direct settings control shall be used or if scheduling shall be used.

The behaviour, if scheduling is used and the schedule controller has no valid output (e.g. no schedule is active), shall be the same behaviour as when a setpoint has never been initialized, or when a controllable object has not received any control yet. It depends on the scheduled entity and shall be defined by user requirements or declared in the MICS of the device.

NOTE As an example, a photovoltaic system may feed in as much power as possible into the grid if no schedule is active. A pump may not operate anymore if no schedule is active. A valve may stay at the position it has last reached.

If the scheduled entity is of type "controllable object" and is used as a setpoint (data objects of the CDC APC that are not controlling equipment), the value applied shall be reflected in the data attribute mxVal. If the schedule controller has no valid output, the quality of mxVal of the scheduled entity shall be set to invalid.

5.2.3 Case when the scheduled entity is of type "setting"

For settings, the DO Mod of the schedule controller shall be set to "off" to indicate that the direct settings shall be used.

1 Extension under consideration, within the edition 2.1 of IEC 61850-7-4.

If scheduling is used, the setting value (data attribute with FC=SP) of the scheduled entity will be updated with the scheduled value. If the schedule controller has no valid output, the last value will remain active.

5.3 Schedule behaviour

Schedules may be started by one or multiple UTC times (e.g. 2016-06-10 06:00), one or multiple calendar times (e.g., every Monday at 08:00 h), or by an external trigger.

Implementation rules for the start time:

- A start time is *configured as "UTC time"* when setTime is defined and holds the specified UTC start time within the TSG CDC and not setCal.
- A start time is *configured as "periodic"* when setCal is defined and hosts the calendar-based period parameter within the TSG CDC and not setTime.
- A "periodic" start time may also be configured as *"periodic starting after a defined UTC time"* when both setTime (specifying the UTC time) and setCal (specifying the calendar-based period) are defined within the TSG CDC.

There are variants in the usage of a schedule:

- a schedule may run just once at a defined UTC time if only one start time is present and *configured as UTC time*;
- a schedule may run multiple times at multiple UTC time due to multiple start times that can be *configured as UTC time*;
- a schedule may be repeated periodically at a defined calendar-based period if start time is *configured as periodic*;
- a schedule may be repeated periodically at a multiple calendar-based periods if multiple start times are *configured as periodic*;
- a schedule may be repeated periodically after a defined UTC time if start time is *configured as periodic after a UTC time*. This can also apply with multiple start times defined;
- a schedule may start running based on an external trigger; in such a case, it is said to be "*event driven*";
- the configured values of a schedule may be reused with a new start time or a new external trigger.

Implementation rules for the schedule:

- "*reuse*" condition reflects the status of the "SchdReuse" setting, "not reuse" condition means the opposite of "reuse" condition;
- "*event driven*" condition reflects the status of the "EvTrg" setting, "not event driven" condition means the opposite of "event driven" condition;
- "*periodic*" condition means that at least one StrTm of the schedule is defined as periodic (refer to implementation rules defined in 5.3), "not periodic" condition means the opposite of "periodic" condition;
- "*all StrTm consumed*" condition means that, considering that all defined StrTm are not periodic, all of them are past, "*not all StrTm consumed*" condition means the opposite of "all StrTm consumed" condtion.

The following basic rules apply:

- A schedule can be configured to be *event driven* – in that case, start times are ignored. Once an event triggered schedule has started, it cannot be restarted by another event.

- If a start time is *configured as periodic*, the schedule will be executed periodically based on the configuration of start time. If the start time is *configured as UTC time*, the schedule will run once based on that start time.
- If multiple start times are available, a mixture of start times *configured as UTC time* and others *as periodic* is not allowed. If at least one start time is *configured as periodic*, all start times that are *configured as UTC time* will be ignored.
- If a schedule which is configured with multiple start times as UTC time is running and another start time becomes ready, the running one will have to end by anticipation and the schedule will be restarted with the new start time.
- If a schedule is configured to run periodically (i.e a start time is *configured as periodic)* and the period is shorter than the length of the schedule, **or if another start time is configured as periodic and its period start time occurs,** the running schedule will have to end by anticipation and it will restart. Thus it will strictly respect the given period(s).
- In case, for whatever reason, the schedule cannot start at the time given by the occurrence of a start time setting, but afterwards, then the schedule will run sharply as if it has started at the time given by the start time setting. Said in other words, the schedule won't shift the scheduled values based on the real-time start.

Schedules that are not *configured as periodic* will be automatically released (back to *Not ready* state) once an external trigger has happened or *all StrTm consumed* is true, unless they are configured to be *reuse*.

The overall expected behaviour is described in Table 2.

Table 2 – Expected behaviour of a schedule

<table>
<tr><th>External trigger</th><th colspan="2">Start times</th><th colspan="2">Reuse</th></tr>
<tr><td></td><td></td><td></td><td>not reuse</td><td>reuse</td></tr>
<tr><td>event driven</td><td>Start times are ignored</td><td></td><td>Once it has been enabled and integrity is OK, the schedule runs only once at the first occurrence of external trigger becoming true. Str Time (defined either as UTC time or periodic) is ignored.</td><td>Once it has been enabled and integrity is OK, the schedule runs each times external trigger becomes true (if the schedule has ended before the occurence of this trigger transition). Str Time (defined either as UTC time or periodic) is ignored.</td></tr>
<tr><td rowspan="3">not event driven</td><td>periodic</td><td></td><td colspan="2">Once it has been enabled and integrity is OK, the schedule runs periodically, starting at the first occurrence of the Calendar time and repeats with every re-occurrence of the Calendar time until it is disabled. If multiple start times are defined as periodic, a (re)-occurrence of any of the calendar times will restart the schedule</td></tr>
<tr><td rowspan="2">not periodic</td><td>Single start time configured as UTC time</td><td>Once it has been enabled and integrity is OK, the schedule runs only once at the first occurrence of the start time. At the end, the schedule becomes "not ready"</td><td>Once it has been enabled and integrity is OK, the schedule runs only once at the first occurrence of the start time. At the end, the schedule waits for a valid start time and schedule enabling</td></tr>
<tr><td>Multiple start times configured as UTC time</td><td>The schedule runs as many times as "valid" start times, at the occurrence of any of the start times. Once "all start times have been consumed", the schedule becomes not ready</td><td>The schedule runs as many times as "valid" start times, at the occurrence of any of the start times. Once all the start times have been consumed, the schedule waits for valid Start time(s) and schedule enabling</td></tr>
</table>

5.4 Schedule structure

The following is an overview on the principle information available in the schedules. The detailed modelling as IEC 61850 logical nodes and data objects is described later.

A schedule consists of an indication of the fixed duration of the schedule interval and a series of entries, representing the desired fixed attribute value of the scheduled entity at each consecutive schedule interval time, the first value being attached to the first time interval.

Schedules may have a priority. The default priority is priority 0. Schedules with higher priority take precedence over schedules with lower priority.

Table 3 contains the key information provided within a schedule.

- The first six entries provide the information that is required to control the execution of the schedule, including its configuration and status,
- The last three entries (framed) are the contents of the schedule of which the last entry is repeated for each interval time.

NOTE If the number of instances of typed values exceeds the defined number of schedule entries, the remaining last entries are ignored.

Table 3 – Structure of a schedule

	Priority of schedule
	State of schedule (*Not ready* | *Start Time required* | *Ready* | *Running*)
	Configuration for reusability
	Configuration for event driven execution of the schedule
	Current value determined by the schedule
	Time when the schedule started running (ActStrTm)
	Time when the schedule is planned/intended to start or re-start running (NxtStrTm)
	Configuration of (multiple) Start Time(s) of schedule (StrTm) including the configuration of periodic execution
	Schedule enable error report (SchdEnaErr)
	Number of schedule entries (NumEntr)
	Duration of the schedule interval (IntvEntr)
	Multiple instances of typed value representing the schedule entries

And 2 controls:

	Enable request (EnaReq)
	Disable request (DsaReq)

If concurrent schedules need to run, i.e multiple exclusive schedules with different settings, it is preferable to provide multiple instances of schedule (one per setting) and to use the schedule controller to activate them at the requested time, rather than to try to update the schedule settings after disabling, on the fly.

Schedule running depends very much on clock quality and validity. Device specifications or product profiles should state what a valid clock is in the context of scheduling and how a device should behave in case of invalid/bad quality clock.

5.5 Managing schedules

5.5.1 Overview

The management of the schedules is done on two levels:

- each schedule is managing its state based on various parameters and requests;
- the schedule controller selects the active schedule, among running ones, based on the schedule priority.

NOTE Priority of a schedule is assigned by the user.

5.5.2 The state machine of the schedule

The behaviour of schedules is based on a state machine as shown in Figure 2.

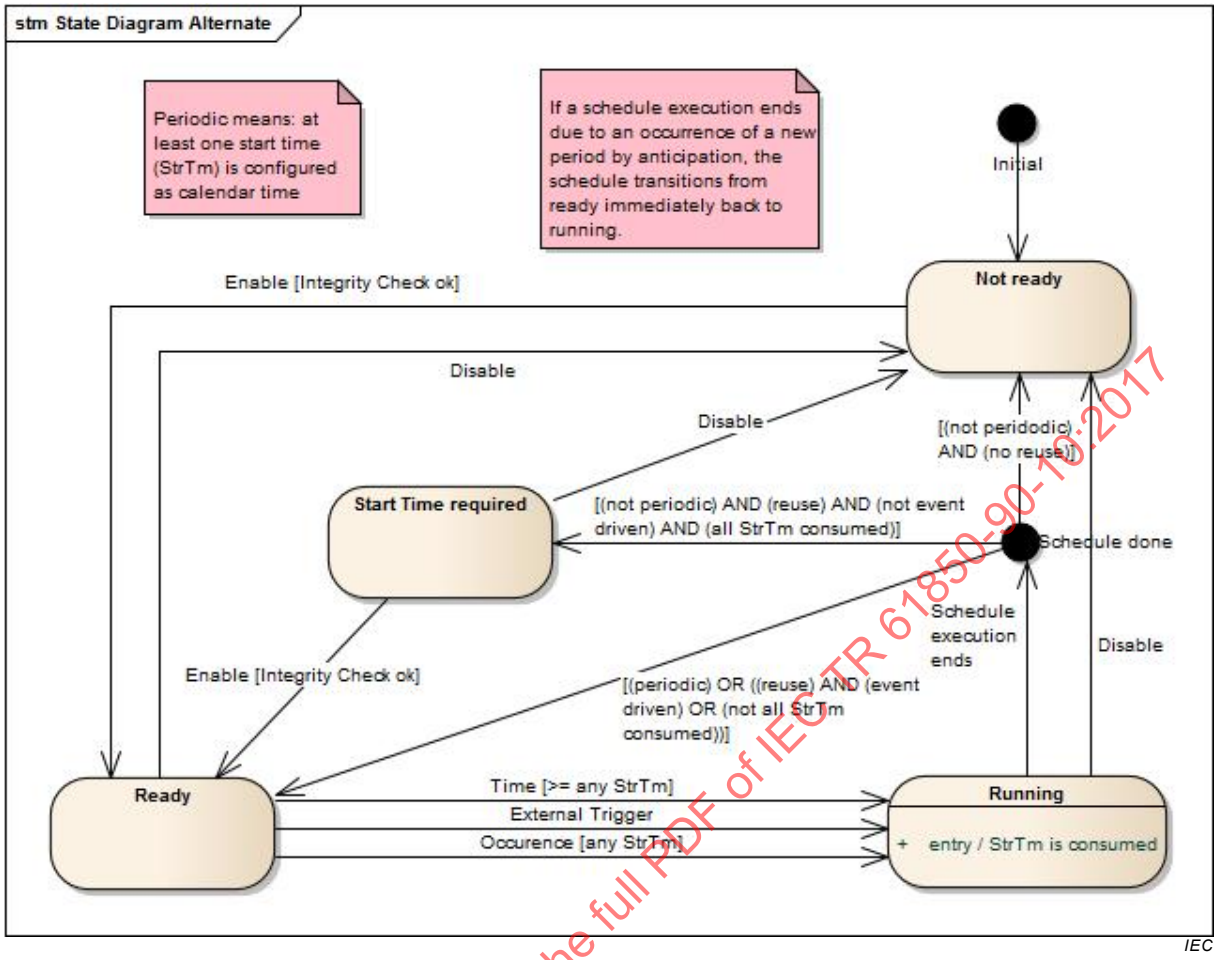


Figure 2 – State diagram for schedule

Schedules are in one of the following four states:

- *Not ready*: the schedule is not ready to be run. The schedule does not contain valid entries. If schedule is in the *Not ready* state, values may be changed.
- *Ready*: the schedule has been configured and enabled by the client with consistent entries and the values of the entries have been validated by the server. That means the values are in supported range. As soon as a schedule is *Ready*, it shall update its planned starting time (NxtStrTm). If *event driven*, this data will remain invalid.
- *Running*: the schedule was *Ready* and the start condition has been met, which is either the current time is later than or equal to one of the Start Time (StrTm) defined in the schedule or an "external event" occurs that causes the schedule to be activated (e.g. a temperature exceeds a specified limit). As soon as a schedule is *Running*, it may update its actual active starting time (ActStrTm). This data object always reflects the UTC time the schedule has theoretically started, i.e. the first entry of the schedule has happened or should have happened (such first entry may have never been effectively output by the schedule because of a late start).
- *Start Time required*: a schedule that is *reuse* is available again but needs a new valid start time to be defined.

The following transitions between states are defined:

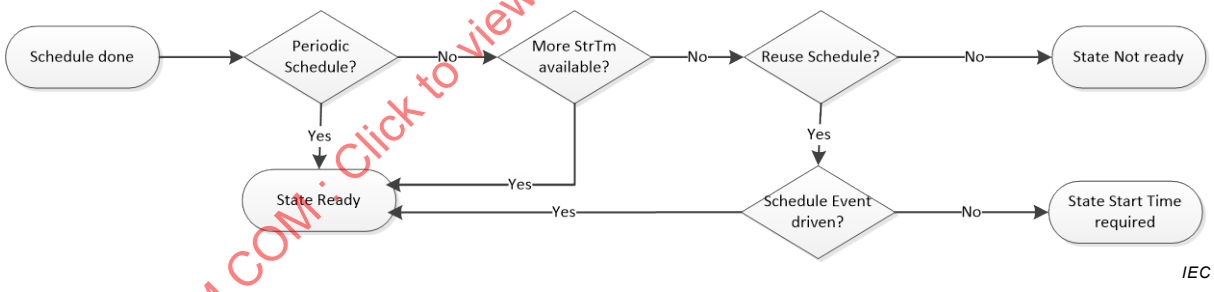
- From *Not ready* to *Ready*:
  - Following a "*Enable*" request issued by a client and under the condition that the schedule is valid, a Start Time is defined (StrTm), or external trigger option is set (EvTrg set to TRUE). Upon enabling, the server hosting the schedule may perform some internal integrity checks that are outside the scope of IEC 61850 (e.g. are the

values defined in the schedule compatible with the capabilities of the device to which the schedule should be applied?). If the validation of the schedule fails, the schedule will remain in the *Not ready* state and a negative response will be issued on the "*Enable*" request. In addition, the cause of the failure is reported into the dedicated Schedule Enable Error data object, to help the requester fixing the issue.

  - The start time or an external trigger have not been reached yet.

- From *Ready* to *Not ready*:
  - Following a "*Disable*" request issued by a client.
- From *Ready* to *Running*:
  - If one of the configured Start Times (StrTm) has been met or if – for a periodic one – an occurrence happened.
  - If a configured external trigger has happened.
- From any states to *Not ready*:

  Whatever its state, a schedule is disabled i.e is back to the *Not ready* state upon the reception of a "*Disable*" request. A "*Disable*" request is of highest priority and overrides all other commands or transitions. A "*Disable*" request is always successful.
- « *schedule execution ended* », i.e from *Running* to "*schedule done*" transition point:
  - "normally" at the end of the schedule duration, i.e. if the schedule ends by completing the interval of the last valid entry;
  - by anticipation, if a new start ime defined as UTC time occurs, i.e. if one of the configured and not yet consumed Start Times (StrTm) has been met;
  - by anticipation, if a new period as defined in the periodic start time(s), occurs i.e. if an occurrence of a periodic one happened.
- From "*schedule done*" transition point to either *Not ready*, or *Time Start required* or *Ready* states (the state that will be reached depends on the conditions described in Figure 3).



NOTE In the case of the periodic event with a schedule duration exactly equal to the period, an immediate return from *Schedule done* to *Ready* and from *Ready* to *Running* is expected, the main objective of the state changes being the reporting of a restart of the schedule.

Figure 3 – Flow chart for transition out of running state

A Start Time (StrTm) *configured as UTC time*, which initiated the running of the schedule will be automatically set to 00 by the schedule as soon as the schedule is running. Thus dual write access mecanism (by the client and by the schedule itself) for any StrTm DO should be carefully implemented to avoid unexpected loss of information.

During the *Ready*, *Running* states, the elements below shall not be changed:

- priority of schedule;
- configuration for reusability;
- configuration for event-driven execution of the schedule;

- number of schedule entries (NumEntr);
- duration of the schedule interval (IntvEntr);
- instances of typed value representing the schedule entries;
- configuration of (multiple) Start Time(s) of schedule (StrTm) including the configuration of periodic execution.

When the schedule is in the *Start Time required* state, the same rule as above applies, except for the Start time(s), which may be modified to configure new start times.

NOTE A schedule does not know whether it is active or not (activation is the result of a schedule controller). Only a given schedule controller (referencing this schedule) will determine which schedule is active, in its context, and exposes it through the value of its ActSchdRef data object. The scheduled entity may know which schedule is active at a given time by reading this data object (or by subscribing to it) .

This allows a schedule to be "used" by many schedule controllers at the same time. In case a schedule is shared by many schedule controllers, it may even be active as far as one controller is considered, but also be not active if seen from another controller.

5.5.3 The role of the schedule controller

A schedule controller is associated with an entity that is controlled by schedules (e.g. active power, state of a switch; see also 4.2 and 6.2). Several schedules may be associated to a schedule controller. The schedule controller class has a reference to the (single) entity it controls (e.g. a setpoint for active power or the activation of a Volt-VAr mode).

Each schedule controller is associated to a known set of schedules. Per schedule controller, only a schedule that is in the *Running* state may affect a scheduled entity. Out of all schedules that are in the *Running* state, at most one schedule is *Active* at a time, which then determines the value resulting from the scheduling system. (See in Annex A some specific case related to the "Mod" of the LNs in charge of supporting schedules and schedule controllers).

The schedule controller is responsible for determining the *Active* schedule at a given time. The following rules apply to determine the *Active* schedule:

- Between schedules in the *Running* state with different priorities, the schedule with the highest priority is the *Active* schedule.
- Between schedules with the same priority, the new schedule that transitions into the *Running* state becomes the *Active* schedule. If two schedules are configured with the same start time and the same priority: this is considered a misconfiguration and it is unpredictable which schedule will become *Running*. If for a schedule that is started by an event, the event happens at exactly the start time of another schedule with the same priority, the one triggered by the event shall become *Active*. If two schedules are triggered by events and the event happens exactly at the same time, it will be random which schedule will be activated. Such a configuration should be avoided.
- For periodic schedules, a new occurrence is considered as transition in the *Running* state.
- Between *Running* schedules with the same priority, if the *Active* schedule ends, a last-in-first-out (LIFO) rule shall apply to determine which schedule to consider for becoming the new *Active* one when this schedule ends.
- In case no *Running* schedule is available at a given time, the output of the controller is set to "quality:invalid".

An example is shown in Figure 4. The sequence and precedence of each of these schedules is shown at the bottom as follows:

- A first schedule with Prio 0 that is configured to be periodic and that is in *Running* state continuously.
- A schedule with Prio 1 that changes from *Ready* to *Running* at time t1.
- At time t2, another schedule with Prio 1 changes to the *Running* state and becomes de facto the *Active* one.
- At time t3, this schedule with Prio 1 ends; the still *Running* schedule with Prio 1 becomes *Active*.
- At time t4, this first schedule with Prio 1 ends; the still *Running* schedule with Prio 0 becomes *Active*.
- At time t5, another schedule with Prio 1 changes to the *Running* state.
- At time t6, a schedule with Prio 2 changes to the *Running* state.
- At time t7, the schedule with Prio 2 ends; the still *Running* schedule with Prio 1 becomes *Active*.
- At time t8, this schedule with Prio 1 ends; the still *Running* schedule with Prio 0 becomes *Active*.

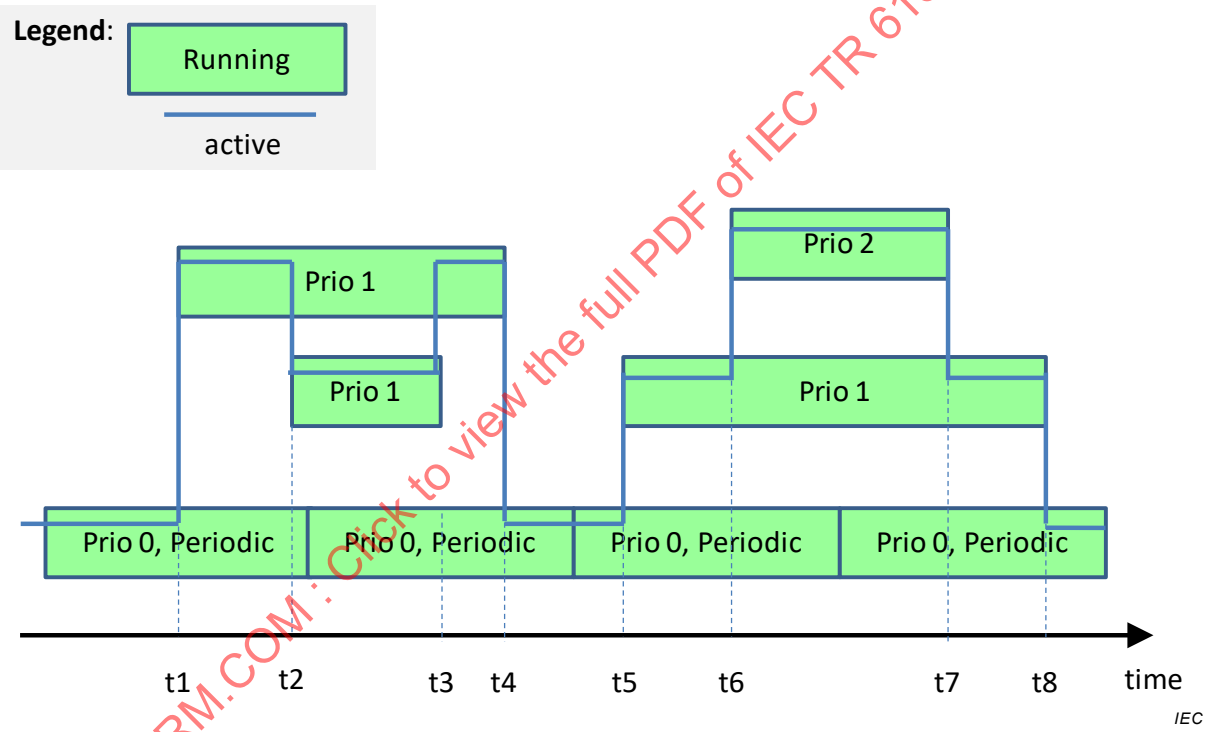


Figure 4 – Handling priorities of schedules to determine the *Active* one

5.5.4 Relations between schedule controller, schedule and scheduled entity

The relations between schedule controller, schedules and entity controlled by the schedule are shown in Figure 5. The schedule controller monitors state and priority of the associated schedules and informs the scheduled entity about the scheduled value.

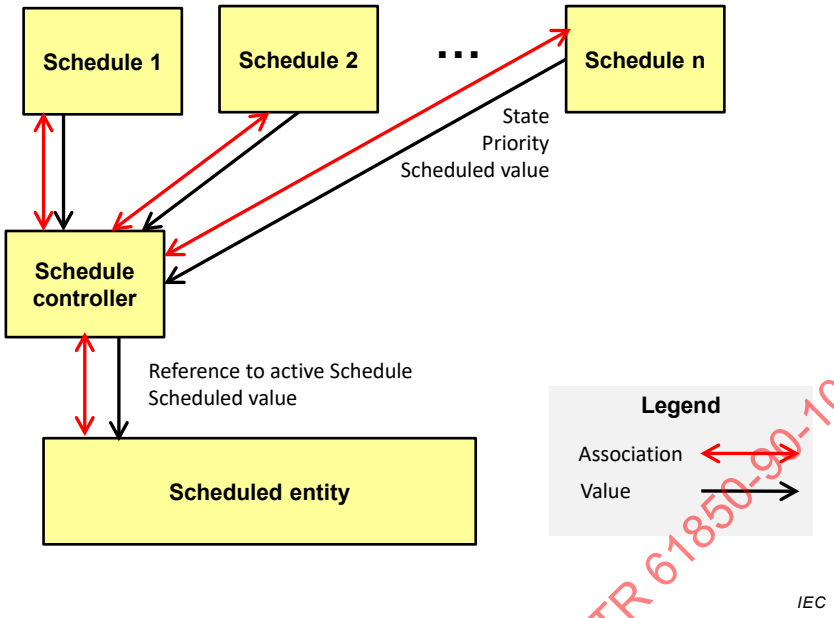


Figure 5 – Relation between schedule controller, schedules and entity controlled

NOTE Any scheduled entity will have in future (request for amending the 7-4 abstract class accordingly) the possibility to run with or without scheduled values, based on a setting defined at the level of the LN hosting the scheduled entity.

5.6 Configuration of schedules

Schedules are defined in IEC 61850 as logical nodes and data objects. A client may configure an already existing schedule in the IED by setting the individual values and then request enabling the schedule.

NOTE As an alternate variant, it could be required to download schedules (with or without individual values) as a file. For that purpose, an SCL structure of the schedule may be used with preconfigured values.

In reality, both methods could be required to work together; i.e. an initial schedule may be downloaded as a file and then later individual values for the entries may be changed through services.

6 IEC 61850 LNs and CDCs

6.1 CDCs of supported scheduled entities

The entities that may be scheduled are Data Objects and may have different types depending on the common data class (CDC) of the entity to be scheduled. These types are:

- Analogue value for CDC: APC and ASG.
- Integer value for CDC: INC and ING
- Boolean value for CDC: SPC and SPG
- Enumerated values for CDC: ENC and ENG

For each type, a different common data class is required to model the schedule as follows:

- ASG for analogue
- ING for integer
- SPG for Boolean
- ENG for enumerated

There may be further new controllable entities to consider for scheduling, but this will be treated in further editions.

Annex B (informative) introduces the new codes for depicting data attributes and objects presence conditions rules (copied from 5.6 of IEC 61850-7-2:–[2]).

Annex C (informative) describes the content of abstract classes used for the definition of the new LNs and CDCs (to help the understanding of this document).

Annex D provides an example of usage of such modelling.

6.2 Package CommonDataClasses_90_10

6.2.1 General

Subclause 6.2 defines the new CDCs associated to this namespace. They should ultimately move to IEC/IS 61850-7-3.

2 Under preparation. Stage at the time of publication: IEC/AFDIS 61850-7-2:2017.

6.2.2 Package CDCStatusInfo

6.2.2.1 General

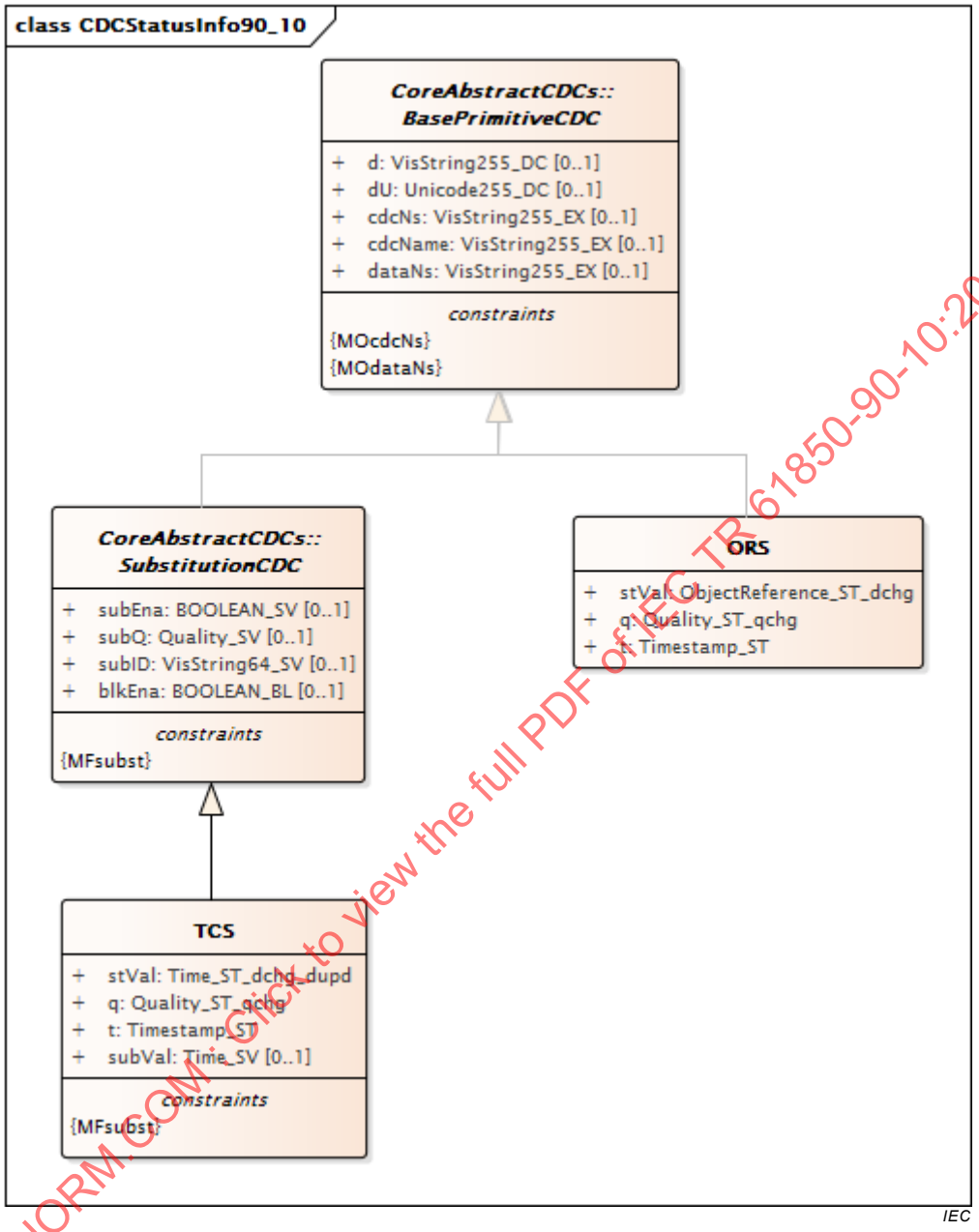


Figure 6 – Class diagram CDCStatusInfo::CDCStatusInfo90_10

Figure 6: this diagram shows all status information CDCs defined in this document with supertypes that factor their common attributes.

6.2.2.2 Time value status (TCS)

This common data class shall be used to represent calculated time values.

Table 4 shows all attributes of TCS.

Table 4 – Attributes of TCS

cdcId = TCS, UML class name = TCS					
Attribute name	**Attribute type**	**FC**	**TrgOp**	**(Value/Value range) Description**	**PresCond**
DataAttribute for status					
stVal	Timestamp	ST	dchg, dupd	Value of the calculated time.	M
q	Quality	ST	qchg	Quality of the value in 'stVal'.	M
t	Timestamp	ST		Timestamp of the last change of the value in any of 'stVal' or 'q'.	M
DataAttribute for substitution and blocked					
subEna	BOOLEAN	SV		inherited from: SubstitutionCDC	MFsubst
subVal	Timestamp	SV		Value used to substitute 'stVal'.	MFsubst
subQ	Quality	SV		inherited from: SubstitutionCDC	MFsubst
subID	VisString64	SV		inherited from: SubstitutionCDC	MFsubst
blkEna	BOOLEAN	BL		inherited from: SubstitutionCDC	O
DataAttribute for configuration, description and extension					
d	VisString255	DC		inherited from: BasePrimitiveCDC	O
dU	Unicode255	DC		inherited from: BasePrimitiveCDC	O
cdcNs	VisString255	EX		inherited from: BasePrimitiveCDC	MOcdcNs
cdcName	VisString255	EX		inherited from: BasePrimitiveCDC	MOcdcNs
dataNs	VisString255	EX		inherited from: BasePrimitiveCDC	MOdataNs

6.2.2.3 Object reference status (ORS)

This common data class shall be used to hold the reference to an element the data is referring to.

Table 5 shows all attributes of ORS.

Table 5 – Attributes of ORS

cdcId = ORS, UML class name = ORS					
Attribute name	**Attribute type**	**FC**	**TrgOp**	**(Value/Value range) Description**	**PresCond**
DataAttribute for status					
stVal	ObjectReference	ST	dchg	Reference to an element the data is referring to.	M
q	Quality	ST	qchg	Quality of the value in 'stVal'.	M
t	Timestamp	ST		Timestamp of the last event of 'stVal' or the last change of value in 'q'.	M
DataAttribute for configuration, description and extension					
d	VisString255	DC		inherited from: BasePrimitiveCDC	O
dU	Unicode255	DC		inherited from: BasePrimitiveCDC	O
cdcNs	VisString255	EX		inherited from: BasePrimitiveCDC	MOcdcNs
cdcName	VisString255	EX		inherited from: BasePrimitiveCDC	MOcdcNs
dataNs	VisString255	EX		inherited from: BasePrimitiveCDC	MOdataNs

6.3 Logical node classes

6.3.1 General

This subclause specifies logical nodes defined in IEC/TR 61850-90-10.

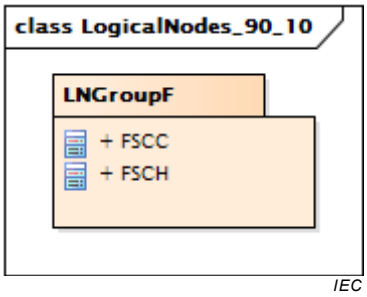


Figure 7 – Class diagram LogicalNodes_90_10::LogicalNodes_90_10

Figure 7: this diagram shows the LN classes included in this namespace.

6.3.2 Package LNGroupF

6.3.2.1 General

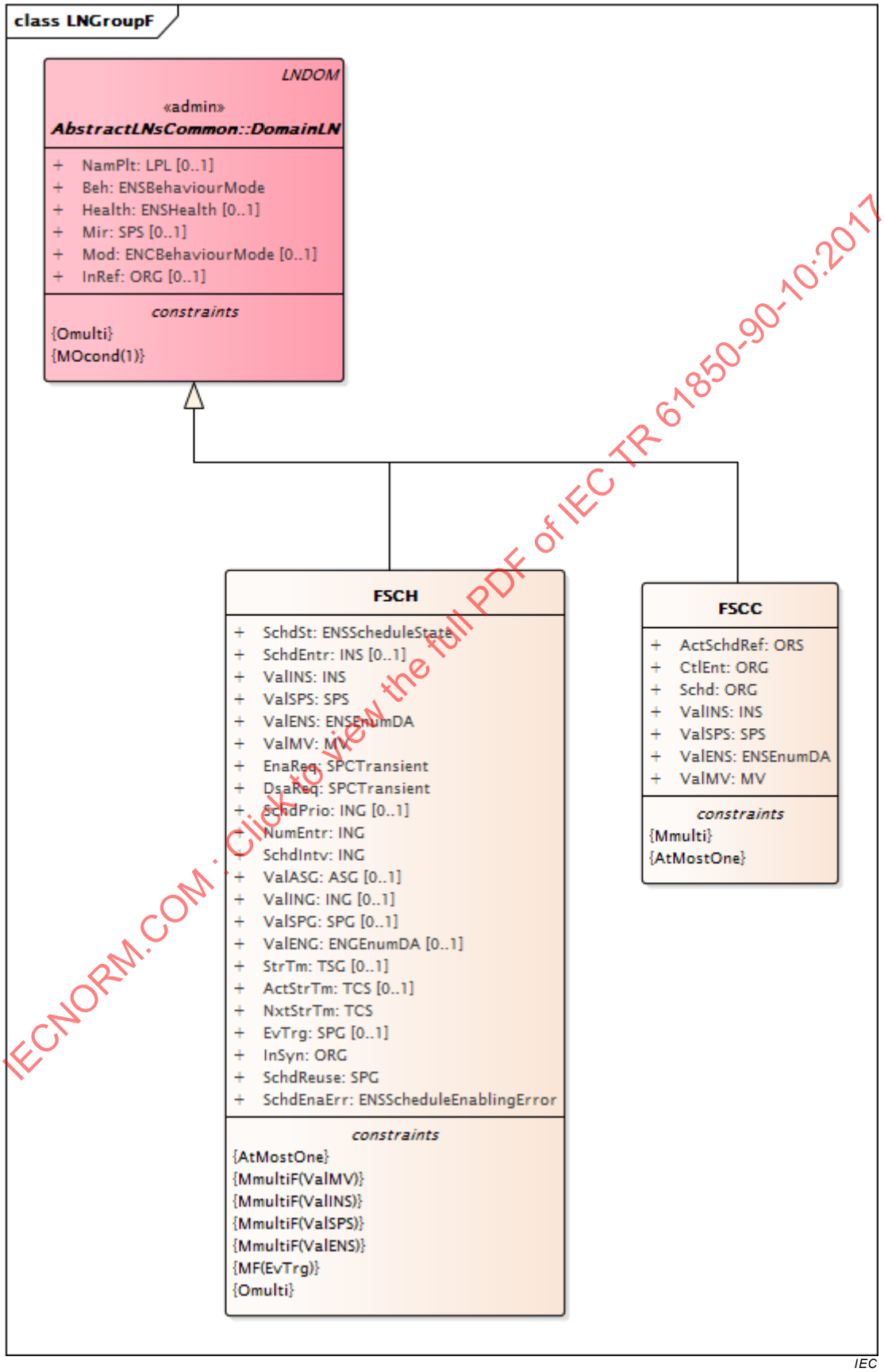


Figure 8 – Class diagram LNGroupF::LNGroupF

Figure 8: this digram shows the details of the LN classes of F Group included in this namespace, in addition to their relationship with their parent class.

6.3.2.2 LN: Schedule controller Name: FSCC

This logical node defines a schedule controller. It determines which schedule is currently active based on state and priority of the schedules and it exposes the actual scheduled value.

Table 6 shows all data objects of FSCC.

Table 6 – Data objects of FSCC

FSCC				
Data object name	**Common data class**	**T**	**Explanation**	**PresCond nds/ds**
Descriptions				
NamPlt	LPL		inherited from: DomainLN	O / na
Status information				
ActSchdRef	ORS		Indication of which schedule is active as an object reference. As long as no schedule is active the quality of the value shall be set to invalid. The active schedule is selected by the schedule controller within the running schedules based on the priority of the running schedules.	M / na
ValINS	INS		Current value determined by the scheduling system. As long as no schedule is active the quality of the value is invalid. The unit of this data shall be the same as the unit of the data object(s) ValING.	AtMostOne / na
ValSPS	SPS		Current value determined by the scheduling system. As long as no schedule is active the quality of the value is invalid.	AtMostOne / na
ValENS	ENS (EnumDA)		Current value determined by the scheduling system. As long as no schedule is active the quality of the value is invalid.	AtMostOne / na
Beh	ENS (BehaviourModeKind)		inherited from: DomainLN	M / na
Health	ENS (HealthKind)		inherited from: DomainLN	O / na
Mir	SPS		inherited from: DomainLN	MOcond(1) / na
Measured and metered values				
ValMV	MV			AtMostOne / na
Controls				
Mod	ENC (BehaviourModeKind)		inherited from: DomainLN	O / na
Settings				
CtlEnt	ORG		Object reference to the entity controlled by the schedule.	M / na
Schd	ORG		Object reference of schedule n (an LN reference to the LN of class FSCH).	Mmulti / na
InRef	ORG		inherited from: DomainLN	Omulti / na

6.3.2.3 LN: Schedule Name: FSCH

This logical node defines a schedule. Multiple schedules can be defined, using FSCC LN to control which ones are active.

Table 7 shows all data objects of FSCH.

Table 7 – Data objects of FSCH

FSCH				
Data object name	**Common data class**	**T**	**Explanation**	**PresCond nds/ds**
Descriptions				
NamPlt	LPL		inherited from: DomainLN	O / na
Status information				
SchdSt	ENS (ScheduleStateKind)		State of this schedule	M / na
SchdEntr	INS		The current schedule entry of a running schedule. This is the Data-Instance-ID of the data object ValXXX (e.g. ValASG). As long as the schedule is not running the value shall be 0.	O / na
ValINS	INS		Current value determined by the schedule. As long as the schedule is not running the quality of the value shall be set to invalid. The unit of this data shall be the same as the unit of the data object(s) ValING.	AtMostOne / na
ValSPS	SPS		Current value determined by the schedule. As long as the schedule is not running the quality of the value shall be set to invalid.	AtMostOne / na
ValENS	ENS (EnumDA)		Current value determined by the schedule. As long as the schedule is not running the quality of the value shall be set to invalid.	AtMostOne / na
ActStrTm	TCS		The time when the schedule started running. If the schedule is not running, the quality of the value is set to invalid.	O / na
NxtStrTm	TCS		The next time when the schedule is planned/intented to start or re-start running. If no schedules are planned running, the quality of the value is set to invalid.	M / na
SchdEnaErr	ENS (ScheduleEnablingErrorKind)		Enumeration reflecting the potential error which has happened at the latest Schedule Enabling request	M / na
Beh	ENS (BehaviourModeKind)		inherited from: DomainLN	M / na
Health	ENS (HealthKind)		inherited from: DomainLN	O / na
Mir	SPS		inherited from: DomainLN	MOcond(1) / na
Measured and metered values				
ValMV	MV		Current value determined by the schedule. As long as the schedule is not running the quality of the value shall be set to invalid. The unit of this data shall be the same as the unit of the data object(s) ValASG.	AtMostOne / na
Controls				
EnaReq	SPC	T	(controllable) Operating with value true initiates enable transition request according to the state diagram; operating with value false is ignored. The change of its status value is a local issue.	M / na
DsaReq	SPC	T	(controllable) Operating with value true initiates disable transition request according to the state	M / na

FSCH				
Data object name	**Common data class**	**T**	**Explanation**	**PresCond nds/ds**
			diagram; operating with value false is ignored. The change of its status value is a local issue.	
Mod	ENC (BehaviourModeKind)		inherited from: DomainLN	O / na
Settings				
SchdPrio	ING		The priority relation of this schedule (0..n) with higher numbers superseding lower numbers. Shall be a positive value. If missing the default is zero.	O / na
NumEntr	ING		The number of schedule entries that are valid out of the instantiated ValASG, ValING, ValSPG or ValENG. NumEntr shall be > 0. NumEntr is <= number of instantiated Val[ASG|ING|SPG|ENG]'s	M / na
SchdIntv	ING		The schedule interval duration in time entities as specified in the SchdIntv.units.SIUnit. The SIUnit is mandatory in this data object. The total duration of a schedule shall be (NumEntr x SchdIntv).	M / na
ValASG	ASG		The ASG scheduled values (current value output as MV).	MmultiF(ValMV) / na
ValING	ING		The ING scheduled values (current value output as INS).	MmultiF(ValINS) / na
ValSPG	SPG		The SPG scheduled values (current value as SPS).	MmultiF(ValSPS) / na
ValENG	ENG (EnumDA)		The ENG scheduled values (current value as ENS).	MmultiF(ValENS) / na
StrTm	TSG		Start time of the schedule in UTC time (setTm of TimeStamp type as defined in 7-3) or as Calendar time (setCal of CalendarTime type as defined in 7-3) or as Calendar time as defined by setCal, after the UTC time as defined by setTm, in case both are defined and valid. With multiple instances, it helps defining a list of time stamps or calendar entries. If from the multiple instances, a specific entry is not used, the time shall be set with setTm set to 00 UTC time.	Omulti / na
EvTrg	SPG		If true, the change of the schedule to the running state may be as well triggered by an external event. The external event shall be referenced through 'InSyn'.	O / na
InSyn	ORG		Object reference of the external trigger if the schedule is triggered by an event. Note that this shall be a reference to a Boolean and the change from FALSE to TRUE shall trigger.	MF(EvTrg) / na
SchdReuse	SPG		If true, once ended the schedule may be reused and transition as described in the state diagram, otherwise it will change to the not ready state.	M / na
InRef	ORG		inherited from: DomainLN	Omulti / na

6.4 Enumerated data attribute types

6.4.1 General

This subclause contains an explicit definition of enumerated types used in IEC/TR 61850-90-10.

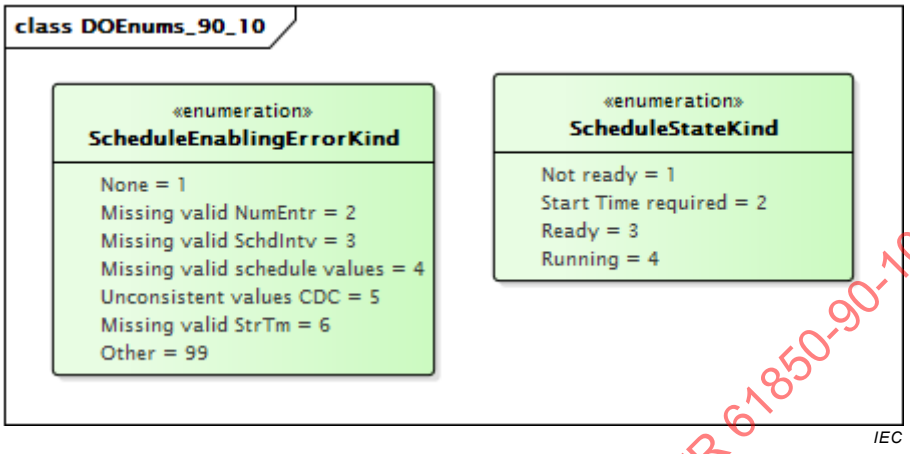


Figure 9 – Class diagram DOEnums_90_10::DOEnums_90_10

Figure 9: this diagram sums up the enumeration lists defined and used in this namespace

6.4.2 Schedule states (ScheduleStateKind enumeration)

This enumeration lists possible states of a schedule.

Table 8 shows all enumeration items of ScheduleStateKind.

Table 8 – Literals of ScheduleStateKind

ScheduleStateKind		
enumeration item	**value**	**description**
Not ready	1	
Start Time required	2	
Ready	3	
Running	4	

6.4.3 Schedule states (ScheduleEnablingErrorKind enumeration)

This enumeration lists possible states of a schedule.

Table 9 shows all enumeration items of ScheduleEnablingErrorKind.

Table 9 – Literals of ScheduleEnablingErrorKind

ScheduleEnablingErrorKind		
enumeration item	**value**	**description**
None	1	
Missing valid NumEntr	2	
Missing valid SchdIntv	3	
Missing valid schedule values	4	

ScheduleEnablingErrorKind		
enumeration item	**value**	**description**
Unconsistent values CDC	5	Schedule values with multiple CDCs have been set
Missing valid StrTm	6	At least one valid StrTm is needed
Other	99	

6.5 SCL enumerations (from DOEnums_90_10)

```
<EnumType id="ScheduleStateKind">
   <EnumVal ord="1">Not ready</EnumVal>
   <EnumVal ord="2">Start Time required</EnumVal>
   <EnumVal ord="3">Ready</EnumVal>
   <EnumVal ord="4">Running</EnumVal>
</EnumType>
<EnumType id="ScheduleEnablingErrorKind">
   <EnumVal ord="1">None</EnumVal>
   <EnumVal ord="2">Missing valid NumEntr</EnumVal>
   <EnumVal ord="3">Missing valid SchdIntv</EnumVal>
   <EnumVal ord="4">Missing valid schedule values</EnumVal>
   <EnumVal ord="5">Unconsistent values CDC</EnumVal>
   <EnumVal ord="6">Missing valid StrTm</EnumVal>
   <EnumVal ord="99">Other</EnumVal>
</EnumType>
```

6.6 Schedule implementation examples

Refer to Annex E for schedule implementation examples.

Annex A
(informative)

Impact of Mode (Mod) on the schedule status and on the controllable entity

Schedule	Schedule controller	Schedule status	Expected schedule controller behaviour	Controllable entity
on	on	normal situation	normal situation	normal situation
blocked	on	no change	no change	as if normally processed
x	blocked	no change	no change	as if normally processed
test or test/blocked	on	status value remains unchanged, but with quality test=true	The schedule controller will consider the concerned schedule as "invalid" and search the next *Running* (and not in test) schedule	as if the concerned schedule was operated as "not ready"
on	test or test/blocked	no impact of the schedule output	The status of the current *Active* schedule shall have the quality test=true	If not in test mode, the controlable entity shall consider the *Active* schedule reference as "invalid". If in test mode, the controllable entity shall process it as valid
test or test/blocked	test or test/blocked	status value remains unchanged, but with quality test=true	The status of the current *Active* schedule shall have the quality test=true	If not in test mode, the controlable entity shall consider the *Active* schedule reference as "invalid". If in test mode, the controllable entity shall process it as valid
off	on	status value is irrelevant, and quality invalid=true	as if the concerned schedule was operated as "not ready"	as if the concerned schedule was operated as "not ready"
x	off	x	status of the current *Active* schedule is irrelevant, and its quality invalid=true	as if no schedule value was forced, see clause 5.2

Annex B
(informative)

Conditions for element presence

This annex introduces conditions that specify presence of elements in a given context (one LN, or one CDC, or one data attribute type, or one data object for dataNs). It is copied from Subclause 5.6 of IEC 61850-7-2:2010.

Table B.1 shows presence conditions.

Table B.1 – Conditions for presence of elements within a context

Condition name	Definition
M	Element is mandatory.
O	Element is optional.
F	Element is forbidden.
na	Element is not applicable.
Mmulti	At least one element shall be present; all instances have an instance number >0.
Omulti	Zero or more elements may be present; all instances have an instance number >0.
AtLeastOne(n)	Parameter n: group number (>0). At least one of marked elements of a group n shall be present.
AtMostOne	At most one of marked elements shall be present.
AllOrNonePerGroup(n)	Parameter n: group number (>0). All or none of the elements of a group n shall be present.
AllOnlyOneGroup(n)	Parameter n: group number (>0). All elements of only one group n shall be present.
MF(sibling)	Parameter sibling: sibling element name. Mandatory if sibling element is present, otherwise forbidden.
MO(sibling)	Parameter sibling: sibling element name. Mandatory if sibling element is present, otherwise optional.
OM(sibling)	Parameter sibling: sibling element name. Optional if sibling element is present, otherwise mandatory.
FM(sibling)	Parameter sibling: sibling element name. Forbidden if sibling element is present, otherwise mandatory.
MOcond(condID)	Parameter condID: condition number (>0). Textual presence condition (non-machine processable) with reference condID to context specific text. If satisfied, the element is mandatory, otherwise optional.
MFcond(condID)	Parameter condID: condition number (>0). Textual presence condition (non-machine processable) with reference condID to context specific text. If satisfied, the element is mandatory, otherwise forbidden.
OFcond(condID)	Parameter condID: condition number (>0). Textual presence condition (non-machine processable) with reference condID to context specific text. If satisfied, the element is optional, otherwise forbidden.
MmultiRange(min, max)	Parameters min, max: limits for instance number (>0). One or more elements must be present; all instances have an instance number within range [min, max] (see IEC 61850-7-1).
OmultiRange(min, max)	Parameters min, max: limits for instance number (>0). Zero or more elements may be present; all instances have an instance number within range [min, max] (see IEC 61850-7-1).

Condition name	Definition
MFsubst	Element is mandatory if substitution is supported (for substitution, see IEC 61850-7-2), otherwise forbidden.
MOln0	Element is mandatory in the context of LLN0; otherwise optional.
MFln0	Element is mandatory in the context of LLN0; otherwise forbidden.
MOlnNs	Element is mandatory if the name space of its logical node deviates from the name space of the containing logical device, otherwise optional. See IEC 61850-7-1 for use of name space.
MOdataNs	Element is mandatory if the name space of its data object deviates from the name space of its logical node, otherwise optional. See IEC 61850-7-1 for use of name space.
MFscaledAV	Element is mandatory* if any sibling elements of type AnalogueValue include 'i' as a child, otherwise forbidden. *Even though devices without floating point capability cannot exchange floating point values through ACSI services, the description of scaling remains mandatory for their (SCL) configuration.
MFscaledMagV	Element is mandatory* if any sibling elements of type Vector include 'i' as a child of their 'mag' attribute, otherwise forbidden. *See MFscaledAV.
MFscaledAngV	Element is mandatory* if any sibling elements of type Vector include 'i' as a child of their 'ang' attribute, otherwise forbidden. *See MFscaledAV.
MOrms	Element is mandatory if the harmonic values in the context are calculated as a ratio to RMS value (value of data attribute 'hvRef' is 'rms'), optional otherwise.
MOrootLD	Element is mandatory in the context of a root logical device; otherwise it is optional.
MOoperTm	Element is mandatory if at least one controlled object on the IED supports time activation service; otherwise it is optional.
MmultiF(sibling)	Parameter sibling: sibling element name. One or more elements must be present if sibling element is present, otherwise forbidden.
MFsbo	Element is mandatory if declared control model supports 'sbo-with-normal-security' or 'sbo-with-enhanced-security', otherwise forbidden.
MFenhanced	Element is mandatory if declared control model supports 'direct-with-enhanced-security' or 'sbo-with-enhanced-security', otherwise forbidden.

Annex C
(informative)

Abstract classes used as a basis for new LNs/CDCs

C.1 General

This annex provides a copy of the content of abstract classes upon which new LNs and CDCs of this report have been set up (please refer to the original document to guarantee the exactness of the contents of this annex).

C.2 <<abstract>> Common attributes for primitive CDC (BasePrimitiveCDC)

This abstract type groups attributes common to all primitive common data classes.

Table C.1 shows all attributes of BasePrimitiveCDC.

Table C.1 – Attributes of BasePrimitiveCDC

UML class name = BasePrimitiveCDC					
Attribute name	**Attribute type**	**FC**	**TrgOp**	**(Value/Value range) Description**	**PresCond**
DataAttribute for configuration, description and extension					
d	VisString255	DC		Textual description of the data. In case it is used within the CDC LPL, the description refers to the logical node.	O
dU	Unicode255	DC		Textual description of the data using unicode characters. In case it is used within the CDC LPL, the description refers to the logical node.	O
cdcName	VisString255	EX		Name of the common data class. For details see IEC 61850-7-1.	O
dataNs	VisString255	EX		Data name space. For details, see IEC 61850-7-1.	MOdataNs

C.3 <<abstract,admin>> LN: Domain Name: DomainLN

Abstract type, holding attributes common to all of all the domain logical nodes.

Table C.2 shows all data objects of DomainLN.

Table C.2 – Data objects of DomainLN

DomainLN				
Data object name	**Common data class**	**T**	**Explanation**	**PresCond nds/ds**
Descriptions				
NamPlt	LPL		Name plate of the logical node.	O / O
Status information				
Beh	ENS (BehaviourModeKind)		Read-only value, describing the behaviour of a domain logical node. It depends on the current operating mode of the logical node ('DomainLN.Mod'), and the current operating mode of the logical device that contains it ('LLN0.Mod'). Processing of the quality status ('q') of the received data is the prerequisite for correct interpretation of 'DomainLN.Beh'.	M / M
Health	ENS (HealthKind)		Reflects the state of the logical node related hardware and software. More detailed information related to the source of the problem may be provided by specific attribute of the logical node.	O / O
Mir	SPS		If true, the LN is a mirror and represents a LN from another LD that does not reflect this physical device. If false, it is recommended to omit the data object Mirror.	MO(1) / MO(1)
Controls				
Mod	ENC (BehaviourModeKind)		(controllable) Operating mode of the domain logical node that may be changed by operator. Processing of the quality status ('q') of the received data is the prerequisite for correct interpretation of the operating mode.	O / O
Settings				
InRef	ORG		Object reference of data object bound to the input n.	Omulti / Omulti