IEC 62541-13

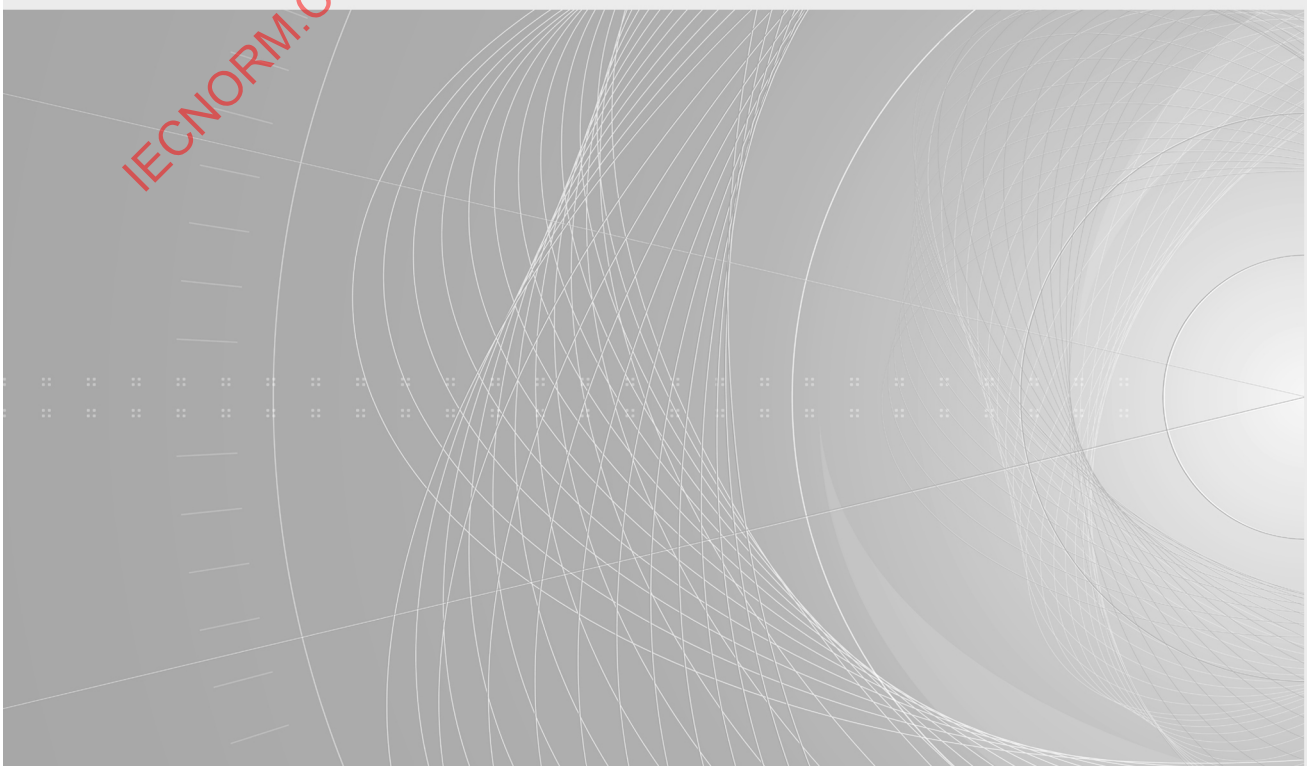
Edition 2.0 2020-06
REDLINE VERSION

INTERNATIONAL STANDARD

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

colour inside

OPC Unified Architecture – Part 13: Aggregates



IEC 62541-13:2020-06 RLV(en)

THIS PUBLICATION IS COPYRIGHT PROTECTED
Copyright © 2020 IEC, Geneva, Switzerland

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

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

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

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

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

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 It also gives information on projects, replaced and withdrawn publications.

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

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

Electropedia - www.electropedia.org
The world's leading online dictionary on electrotechnology, containing more than 22 000 terminological entries 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
67 000 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.

IECNORM.COM : Click to view the full PDF of IEC 60601-1-13:2020 RLV

IEC 62541-13

Edition 2.0 2020-06
REDLINE VERSION

INTERNATIONAL STANDARD

colour inside

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

OPC Unified Architecture – Part 13: Aggregates

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

ICS 25.040.40; 35.100.05

ISBN 978-2-8322-8530-5

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

® Registered trademark of the International Electrotechnical Commission

CONTENTS

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

INTERNATIONAL ELECTROTECHNICAL COMMISSION

_____

OPC UNIFIED ARCHITECTURE –

Part 13: Aggregates

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.

This redline version of the official IEC Standard allows the user to identify the changes made to the previous edition. A vertical bar appears in the margin wherever a change has been made. Additions are in green text, deletions are in strikethrough red text.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

IEC 62541-13 has been prepared by subcommittee 65E: Devices and integration in enterprise systems, of IEC technical committee 65: Industrial-process measurement, control and automation.

This second edition cancels and replaces the first edition of IEC 62541-13, published in 2015. This edition constitutes a technical revision.

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

a) no technical changes but numerous clarifications. Also some corrections to the examples.

The text of this standard is based on the following documents:

FDIS	Report on voting
65E/697/FDIS	65E/712/RVD

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

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

Throughout this document and the other Parts of the series, certain document conventions are used:

Italics are used to denote a defined term or definition that appears in the “Terms and definition” clause in one of the parts of the series.

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

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

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

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

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

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.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

OPC UNIFIED ARCHITECTURE –

Part 13: Aggregates

1 Scope

This part of IEC 62541 is part of the overall OPC Unified Architecture specification series and defines the information model associated with Aggregates.

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

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

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

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

IEC 62541-8, *OPC Unified Architecture – Part 8: Data Access*

IEC 62541-11, *OPC Unified Architecture – Part 11: Historical Access*

3 Terms, definitions, and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC TR 62541-1, IEC 62541-3, IEC 62541-4, and IEC 62541-11 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
ProcessingInterval
timespan for which derived values are produced based on a specified *Aggregate*

Note 1 to entry: The total time domain specified for ReadProcessed is divided by the *ProcessingInterval*. For example, performing a 10-minute Average over the time range 12:00 to 12:30 would result in a set of three intervals of *ProcessingInterval* length, with each interval having a start time of 12:00, 12:10 and 12:20 respectively. The rules used to determine the interval *Bounds* are discussed in 5.4.2.2.

3.1.2
interpolated data
data that is calculated from data samples

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Note 1 to entry: Data samples may be historical data or buffered real time data. An *interpolated* value is calculated from the data points on either side of the requested timestamp.

3.1.3
EffectiveEndTime
time immediately before *endTime*

Note 1 to entry: All *Aggregate* calculations include the *startTime* but exclude the *endTime*. However, it is sometimes necessary to return an *Interpolated* End Bound as the value for an *Interval* with a timestamp that is in the *interval*. *Servers* are expected to use the time immediately before *endTime* where the time resolution of the *Server* determines the exact value (do not confuse this with hardware or operating system time resolution). For example, if the *endTime* is 12:01:00, the time resolution is 1 second, then the *EffectiveEndTime* is 12:00:59. See 5.4.2.4.

If time is flowing backwards, *Servers* are expected to use the time immediately after *endTime* where the time resolution of the *Server* determines the exact value.

3.1.4
extrapolated data
data constructed from a discrete data set but is outside of the discrete data set

Note 1 to entry: It is similar to the process of interpolation, which constructs new points between known points, but its result is subject to greater uncertainty. *Extrapolated* data is used in cases where the requested time period falls farther into the future than the data available in the underlying system. See example in Table 1.

3.1.5
SlopedInterpolation
simple linear interpolation

Note 1 to entry: Compare to curve fitting using linear polynomials. See example in Table 1.

3.1.6
SteppedInterpolation
interpolation holding the last data point constant or interpolating the value based on a horizontal line fit

Note 1 to entry: Consider the following Table 1 of raw and *Interpolated*/*Extrapolated* values:

Table 1 – Interpolation examples

Timestamp	Raw Value	Sloped Interpolation	Stepped Interpolation
12:00:00	10		
12:00:05		15	10
12:00:08		18	10
12:00:10	20		
12:00:15		25	20
12:00:20	30		
		SlopedExtrapolation	SteppedExtrapolation
12:00:25		35	30
12:00:27		37	30

3.1.7
bounding values
values at the *startTime* and *endTime* needed for *Aggregates* to compute the result

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Note 1 to entry: If *Raw data* does not exist at the *startTime* and *endTime* a value shall be estimated. There are two ways to determine *Bounding Values* for an interval. One way (called *Interpolated Bounding Values*) uses the first non-Bad data points found before and after the timestamp to estimate the bound. The other (called *Simple Bounding Values*) uses the data points immediately before and after the boundary timestamps to estimate the bound even if these points are Bad. Subclauses 3.1.8 and 3.1.9 describe the two different approaches in more detail.

In all cases the *TreatUncertainAsBad* (see 4.2.1.2) flag is used to determine whether Uncertain values are Bad or non-Bad.

If a Raw value was not found and a non-Bad bounding value exists the *Aggregate* Bits (see 5.3.3) are set to 'Interpolated'.

When calculating *bounding values*, the value portion of *Raw data* that has Bad status is set to null. This means the value portion is not used in any calculation and a null is returned if the raw value is returned. The status portion is determined by the rules specified by the bound or *Aggregate*.

The *Interpolated Bounding Values* approach (see 3.1.8) is the same as what is used in Classic OPC Historical Data Access (HDA) and is important for applications such as advanced process control where having useful values at all times is important. The *Simple Bounding Values* approach (see 3.1.9) is new in this standard and is important for applications which shall produce regulatory reports and cannot use estimated values in place of Bad data.

3.1.8
interpolated bounding values
bounding values determined by a calculation using the nearest Good value

Note 1 to entry: *Interpolated Bounding Values* using *SlopedInterpolation* are calculated as follows:

- if a non-Bad Raw value exists at the timestamp then it is the bounding value;
- find the first non-Bad Raw value before the timestamp;
- find the first non-Bad Raw value after the timestamp;
- draw a line between before value and after value;
- use point where the line crosses the timestamp as an estimate of the bounding value.

The calculation can be expressed with the following formula:

$$V_{\text{bound}} = (T_{\text{bound}} - T_{\text{before}}) \times (V_{\text{after}} - V_{\text{before}})/(T_{\text{after}} - T_{\text{before}}) + V_{\text{before}}$$

where V_x is a value at 'x' and T_x is the timestamp associated with V_x.

If no non-Bad values exist before the timestamp the *StatusCode* is Bad_NoData. The *StatusCode* is *Uncertain_DataSubNormal* if any Bad values exist between the before value and after value. If either the before value or the after value are Uncertain the *StatusCode* is *Uncertain_DataSubNormal*. If the after value does not exist the before value shall be extrapolated using *SlopedExtrapolation* or *SteppedExtrapolation*.

The period of time that is searched to discover the Good values before and after the timestamp is *Server* dependent, but if a Good value is not found within some reasonable time range then the *Server* will assume it does not exist. The *Server* as a minimum should search a time range which is at least the size of the ProcessingInterval.

Interpolated Bounding Values using *SlopedExtrapolation* are calculated as follows:

- find the first non-Bad Raw value before timestamp;
- find the second non-Bad Raw value before timestamp;
- draw a line between these two values;
- extend the line to where it crosses the timestamp;
- use the point where the line crosses the timestamp as an estimate of the bounding value.

The formula is the same as the one used for *SlopedInterpolation*.

The *StatusCode* is always *Uncertain_DataSubNormal*. If only one non-Bad raw value can be found before the timestamp then *SteppedExtrapolation* is used to estimate the bounding value.

Interpolated Bounding Values using *SteppedInterpolation* are calculated as follows:

- if a non-Bad Raw value exists at the timestamp then it is the bounding value;
- find the first non-Bad Raw value before timestamp;
- use the value as an estimate of the bounding value.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

The *StatusCode* is *Uncertain_DataSubNormal* if any Bad values exist between the before value and the timestamp. If no non-Bad *Raw data* exists before the timestamp then the *StatusCode* is Bad_NoData. If the value before the timestamp is Uncertain the *StatusCode* is *Uncertain_DataSubNormal*. The value after the timestamp is not needed when using *SteppedInterpolation*; however, if the timestamp is after the end of the data then the bounding value is treated as extrapolated and the *StatusCode* is *Uncertain_DataSubNormal*.

SteppedExtrapolation is a term that describes *SteppedInterpolation* when a timestamp is after the last value in the history collection.

3.1.9
simple bounding values
bounding values determined by a calculation using the nearest value

Note 1 to entry: *Simple Bounding Values* using *SlopedInterpolation* are calculated as follows:

- if any Raw value exists at the timestamp then it is the bounding value;
- find the first Raw value before timestamp;
- find the first Raw value after timestamp;
- if the value after the timestamp is Bad then the before value is the bounding value;
- draw a line between before value and after value;
- use point where the line crosses the timestamp as an estimate of the bounding value.

The formula is the same as the one used for *SlopedInterpolation* in Clause 3.1.5.

If a Raw value at the timestamp is Bad the *StatusCode* is Bad_NoData. If the value before the timestamp is Bad the *StatusCode* is Bad_NoData. If the value before the timestamp is Uncertain the *StatusCode* is *Uncertain_DataSubNormal*. If the value after the timestamp is Bad or Uncertain the *StatusCode* is *Uncertain_DataSubNormal*.

Simple Bounding Values using *SteppedInterpolation* are calculated as follows:

- if any Raw value exists at the timestamp then it is the bounding value;
- find the first Raw value before timestamp;
- if the value before timestamp is non-Bad then it is the bounding value.

If a Raw value at the timestamp is Bad the *StatusCode* is Bad_NoData. If the value before the timestamp is Bad the *StatusCode* is Bad_NoData. If the value before the timestamp is Uncertain the *StatusCode* is *Uncertain_DataSubNormal*.

~~If either bounding time of an interval is beyond the last data point then extrapolation may be used if the *Server* feels it is appropriate for the data.~~

If either bounding time of an interval is beyond the last data point then the *Server* may use extrapolation or return an error. If extrapolation is used by the server the type [*SteppedExtrapolation* or *SloppedExtrapolation*] of extrapolation is server specific.

In some Historians, the last Raw value does not necessarily indicate the end of the data. Based on the Historian's knowledge of the data collection mechanism, i.e. frequency of data updates and latency, the Historian may extend the last value to a time known by the Historian to be covered. When calculating *Simple Bounding Values* the Historian will act as if there is another Raw value at this timestamp.

In the same way, if the earliest time of an interval starts before the first data point in history and the latest time is after the first data point in history, then the interval will be treated as if the interval extends from the first data point in history to the latest time of the interval and the *StatusCode* of the interval will have the Partial bit set (see 5.3.3.2).

The period of time that is searched to discover the values before and after the timestamp is *Server* dependent, but if a value is not found within some reasonable time range then the *Server* will assume it does not exist. The *Server* as a minimum should search a time range which is at least the size of the ProcessingInterval.

3.2 Abbreviated terms

DA Data Access

HA Historical Access (access to historical data or events)

HDA Historical Data Access

UA Unified Architecture

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

4 Aggregate information model

4.1 General

IEC 62541-3 and IEC 62541-5 standards define the representation of *Aggregate* historical or buffered real time data in the OPC Unified Architecture. This includes the definition of *Aggregates* used in processed data retrieval and in historical retrieval. This definition includes both standard *Reference* types and *Object* types.

4.2 Aggregate Objects

4.2.1 General

4.2.1.1 Overview

OPC UA *Servers* can support several different functionalities and capabilities. The following standard *Objects* are used to expose these capabilities in a common fashion, and there are several standard defined concepts that can be extended by vendors.

4.2.1.2 AggregateConfigurationType

The *AggregateConfigurationType* defines the general characteristics of a *Node* that defines the *Aggregate* configuration of any *Variable* or *Property*. *AggregateConfiguration Object* represents the browse entry point for information on how the *Server* treats *Aggregate* specific functionality such as handling Uncertain data. It is formally defined in Table 2.

Table 2 – AggregateConfigurationType Definition

Attribute	**Value**				
BrowseName	AggregateConfigurationType				
IsAbstract	False				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the *BaseObjectType* defined in IEC 62541-5					
HasProperty	Variable	TreatUncertainAsBad	Boolean	PropertyType	Mandatory
HasProperty	Variable	PercentDataBad	Byte	PropertyType	Mandatory
HasProperty	Variable	PercentDataGood	Byte	PropertyType	Mandatory
HasProperty	Variable	UseSlopedExtrapolation	Boolean	PropertyType	Mandatory

The *TreatUncertainAsBad Variable* indicates how the *Server* treats data returned with a *StatusCode* severity Uncertain with respect to *Aggregate* calculations. A value of True indicates the *Server* considers the severity equivalent to Bad, a value of False indicates the *Server* considers the severity equivalent to Good, unless the *Aggregate* definition says otherwise. The default value is True. Note that the value is still treated as Uncertain when the *StatusCode* for the result is calculated.

The *PercentDataBad Variable* indicates the minimum percentage of Bad data in a given interval required for the *StatusCode* for the given interval for processed data request to be set to Bad. (Uncertain is treated as defined above.) Refer to 5.4.3 for details on using this *Variable* when assigning *StatusCodes*. For details on which *Aggregates* use the *PercentDataBad Variable*, see the definition of each *Aggregate*. The default value is 100.

The *PercentDataGood Variable* indicates the minimum percentage of Good data in a given interval required for the *StatusCode* for the given interval for the processed data requests to be set to Good. Refer to 5.4.3 for details on using this *Variable* when assigning *StatusCodes*. For details on which *Aggregates* use the *PercentDataGood Variable*, see the definition of each *Aggregate*. The default value is 100.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

The *PercentDataGood* and *PercentDataBad* shall follow the following relationship *PercentDataGood* ≥ (100 – *PercentDataBad*). If they are equal the result of the *PercentDataGood* calculation is used. If the values entered for *PercentDataGood* and *PercentDataBad* do not result in a valid calculation (e.g. Bad = 80; Good = 0) the result will have a StatusCode of Bad_AggregateInvalidInputs The StatusCode Bad_AggregateInvalidInputs will be returned if the value of *PercentDataGood* or *PercentDataBad* exceed 100.

The *UseSlopedExtrapolation Variable* indicates how the *Server* interpolates data when no boundary value exists (i.e. extrapolating into the future from the last known value). A value of False indicates that the *Server* will use a *SteppedExtrapolation* format, and hold the last known value constant. A value of True indicates the *Server* will project the value using *UseSlopedExtrapolation* mode. The default value is False. For *SimpleBounds* this value is ignored.

4.2.2 AggregateFunction Object

4.2.2.1 General

This *Object* is used as the browse entry point for information about the *Aggregates* supported by a *Server*. The content of this *Object* is already defined by its type definition. All *Instances* of the *FolderType* use the standard *BrowseName* of 'AggregateFunctions'. The *HasComponent Reference* is used to relate a *ServerCapabilities Object* and/or any ~~*HistoricalServerCapabilities*~~ *HistoryServerCapabilitiesType Object* to an *AggregateFunction Object*. AggregateFunctions is formally defined in Table 3.

Table 3 – Aggregate Functions Definition

Attribute	**Value**				
BrowseName	AggregateFunctions				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Modelling Rule**
HasTypeDefinition	Object Type	*FolderType*	Defined in IEC 62541-5		

Each *ServerCapabilities* and ~~*HistoricalServerCapabilities*~~ *HistoryServerCapabilitiesType Object* shall reference an *AggregateFunction Object*. In addition, each *HistoricalConfiguration Object* belonging to a *HistoricalDataNode* may reference an *AggregateFunction Object* using the *HasComponent Reference*.

4.2.2.2 AggregateFunctionType

This *ObjectType* defines an *Aggregate* supported by a UA *Server*. This *Object* is formally defined in Table 4.

Table 4 – AggregateFunctionType Definition

Attribute	**Value**				
BrowseName	AggregateFunctionType				
IsAbstract	False				
References	**Node Class**	**BrowseName**	**DataType**	**Type Definition**	**Mod. Rule**
Subtype of the *BaseObjectType* defined in IEC 62541-5					

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

For the *AggregateFunctionType*, the *Description Attribute* (inherited from the *Base NodeClass*), is mandatory. The *Description Attribute* provides a localized description of the *Aggregate*.

Table 5 specifies the *BrowseName* and *Description Attributes* for the standard *Aggregate Objects*. The description is the localized "en" text. For other locales it shall be translated.

Table 5 – Standard AggregateType Nodes

BrowseName	Description
	Interpolation Aggregate
Interpolative	At the beginning of each interval, retrieve the calculated value from the data points on either side of the requested timestamp.
Average	Retrieve the average value of the data over the interval.
TimeAverage	Retrieve the time weighted average data over the interval using *Interpolated Bounding Values*.
TimeAverage2	Retrieve the time weighted average data over the interval using *Simple Bounding Values*.
Total	Retrieve the total (time integral) of the data over the interval using *Interpolated Bounding Values*.
Total2	Retrieve the total (time integral) of the data over the interval using *Simple Bounding Values*.
Minimum	Retrieve the minimum raw value in the interval with the timestamp of the start of the interval.
Maximum	Retrieve the maximum raw value in the interval with the timestamp of the start of the interval.
MinimumActualTime	Retrieve the minimum value in the interval and the timestamp of the minimum value.
MaximumActualTime	Retrieve the maximum value in the interval and the timestamp of the maximum value.
Range	Retrieve the difference between the minimum and maximum value over the interval.
Minimum2	Retrieve the minimum value in the interval including the *Simple Bounding Values*.
Maximum2	Retrieve the maximum value in the interval including the *Simple Bounding Values*.
MinimumActualTime2	Retrieve the minimum value with the actual timestamp including the *Simple Bounding Values*.
MaximumActualTime2	Retrieve the maximum value with the actual timestamp including the *Simple Bounding Values*.
Range2	Retrieve the difference between the Minimum2 and Maximum2 value over the interval.
Count	Retrieve the number of raw values over the interval.
DurationInStateZero	Retrieve the time a Boolean or numeric was in a zero state using *Simple Bounding Values*.
DurationInStateNonZero	Retrieve the time a Boolean or numeric was in a non-zero state using *Simple Bounding Values*.
NumberOfTransitions	Retrieve the number of changes between zero and non-zero that a Boolean or numeric value experienced in the interval.
Start	Retrieve the value at the beginning of the interval ~~using *Interpolated Bounding Values*~~.
End	Retrieve the value at the end of the interval ~~using *Interpolated Bounding Values*~~.
Delta	Retrieve the difference between the Start and End value in the interval.
StartBound	Retrieve the value at the beginning of the interval using *Simple Bounding Values*.
EndBound	Retrieve the value at the end of the interval using *Simple Bounding Values*.
DeltaBounds	Retrieve the difference between the StartBound and EndBound value in the interval using *Simple Bounding Values*.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

BrowseName	Description
	Interpolation Aggregate
DurationGood	Retrieve the total duration of time in the interval during which the data is Good.
DurationBad	Retrieve the total duration of time in the interval during which the data is Bad.
PercentGood	Retrieve the percentage of data (0 to 100) in the interval which has Good *StatusCode*.
PercentBad	Retrieve the percentage of data (0 to 100) in the interval which has Bad *StatusCode*.
WorstQuality	Retrieve the worst *StatusCode* of data in the interval.
WorstQuality2	Retrieve the worst *StatusCode* of data in the interval including the *Simple Bounding Values*.
AnnotationCount	Retrieve the number of *Annotation*s in the interval (applies to Historical *Aggregates* only).
StandardDeviationSample	Retrieve the standard deviation for the interval for a sample of the population (n-1).
VarianceSample	Retrieve the variance for the interval as calculated by the StandardDeviationSample.
StandardDeviation Population	Retrieve the standard deviation for the interval for a complete population (n) which includes *Simple Bounding Values*.
VariancePopulation	Retrieve the variance for the interval as calculated by the StandardDeviationPopulation which includes *Simple Bounding Values*.

4.3 MonitoredItem AggregateFilter

4.3.1 MonitoredItem AggregateFilter Defaults

The default values used for *MonitoredItem Aggregates* are the same as those used for historical *Aggregates*. They are defined in 4.2.1.2. For additional information on *MonitoredItem AggregateFilter* see IEC 62541-4.

4.3.2 MonitoredItem Aggregates and Bounding Values

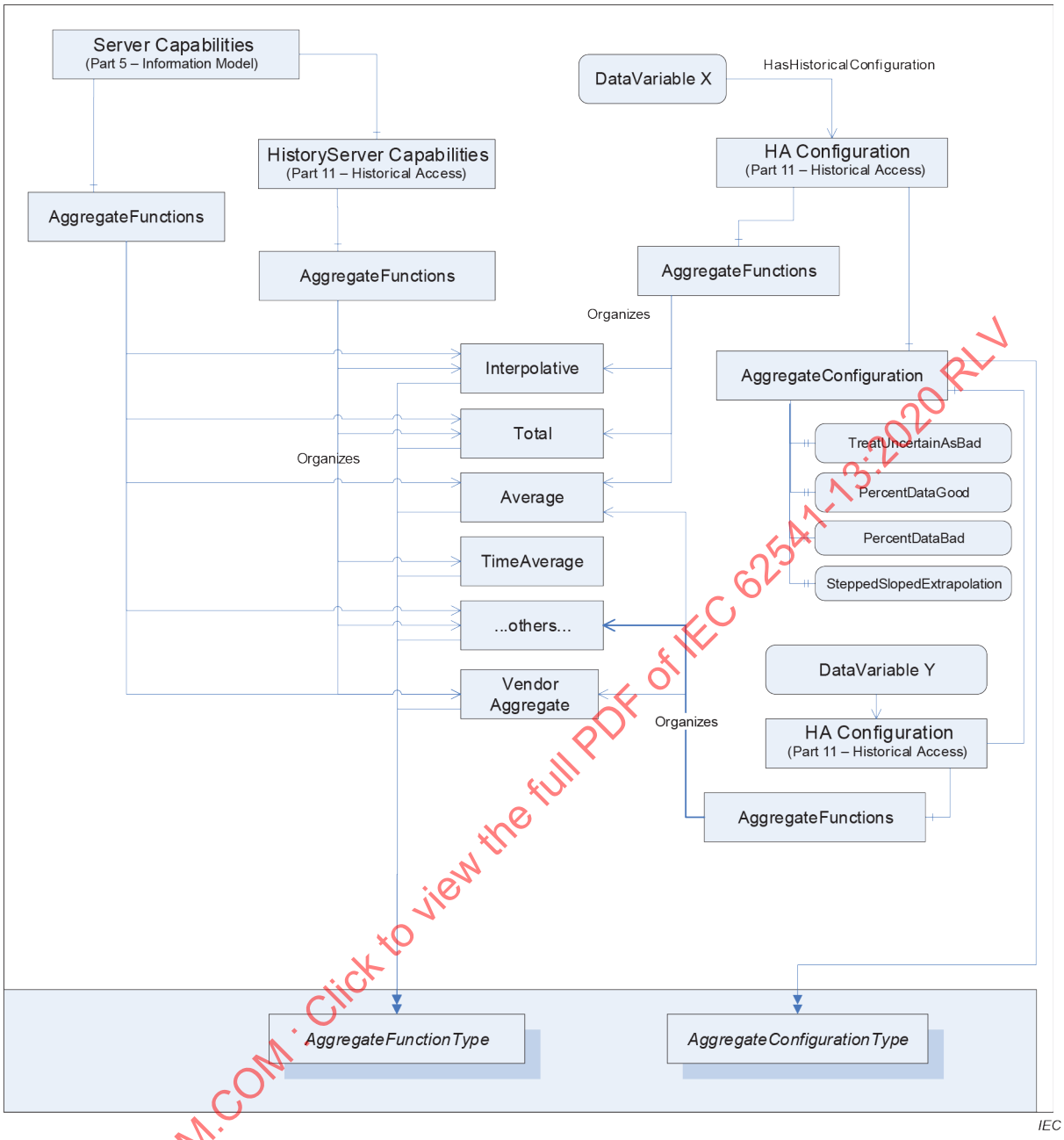
When calculating *MonitoredItem Aggregates* that require the use of *Bounding Values*, the bounds may not be known. The calculation is done in the same manner as a historical read with the Partial Bit set. The historian may wait some amount of time (normally no more than one processing interval) before calculating the interval to allow for any latency in data collection and reduce the use of the Partial Bit.

A historical read done after data collection and the data from the *MonitoredItem* over the same interval may not be the same.

4.4 Exposing Supported Functions and Capabilities

Figure 1 outlines a possible representation of *Aggregate* information in the *AddressSpace*. In this example, although the *Server* at the highest level may support *Aggregate* functionality for Interpolative, Total, Average, and others, *DataVariable* X only supports Interpolative, Total and Average, while *DataVariable* Y supports Average, a vendor defined *Aggregate* and other (unstated) *Aggregates.*

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV



IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Figure 1 – Representation of Aggregate Configuration information in the AddressSpace

5 Aggregate specific usage of Services

5.1 General

IEC 62541-4 specifies all Services needed for OPC UA *Aggregates*. In particular:

- The Browse Service Set or Query Service Set to detect *Aggregates* and their configuration.
- The HistoryRead Service of the Attribute Service Set to read the aggregated history of the HistoricalNodes.
- The CreateMonitoredItems Service allows specifying a filter for each MonitoredItem to read aggregated data.

5.2 Aggregate data handling

5.2.1 Overview

The *HistoryRead* service defined in IEC 62541-4 can perform several different functions. The *historyReadDetails* parameter is an *Extensible Parameter* that specifies which function to perform. The *ReadProcessedDetails* structure is used to read aggregated data for *HistoricalDataNodes*.

The *CreateMonitoredItems Service* allows specifying a filter for each *MonitoredItem*. The *MonitoringFilter* is an extensible parameter whose structure depends on the type of item being monitored. The *AggregateFilter* structure is used to obtain aggregated data for a subscription.

5.2.2 ReadProcessedDetails structure overview

ReadProcessedDetails structure is formally detailed in IEC 62541-11. Table 6 outlines the components of the ReadProcessedDetails structure for the purposes of discussion in this document.

Table 6 – ReadProcessedDetails

Name	Description
ReadProcessedDetails	Specifies the details used to perform a "processed" history read.
startTime	Beginning of period to read.
endTime	End of period to read.
processingInterval	Interval between returned *Aggregate* values.
aggregateType[]	The *NodeIds* of the *AggregateFunction Objects*. *AggregateFunction Objects* indicate the list of *Aggregates* to be used when retrieving processed history.
aggregateConfiguration	*Aggregate* configuration structure.
useServerDefaults	If True the *Server*'s default values are used and any values specified for the other parameters are ignored.
treatUncertainAsBad	See 4.2.1.2.
percentDataBad	See 4.2.1.2.
percentDataGood	See 4.2.1.2.
useSlopedExtrapolation	See 4.2.1.2.

5.2.3 AggregateFilter structure overview

The *AggregateFilter* defines the *Aggregate* function that should be used to calculate the values to be returned. The *AggregateFilter* is formally defined in IEC 62541-4. Table 7 outlines the components of the *AggregateFilter* structure for the purposes of discussion in this document.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 7 – AggregateFilter structure

Name	Description
AggregateFilter	
startTime	Beginning of period to calculate the *Aggregate* the first time.
aggregateType	The *NodeIds* of the *AggregateFunction Objects* that indicates the list of *Aggregates* to be used when retrieving processed data.
processingInterval	The period to be used to compute the *Aggregate*.
aggregateConfiguration	This parameter allows *Clients* to override the *Aggregate* configuration settings supplied by an *AggregateConfiguration Object* on a per monitored item basis.
useServerDefaults	If True the *Server*'s default values are used and any values specified for the other parameters are ignored.
treatUncertainAsBad	See 4.2.1.2.
percentDataBad	See 4.2.1.2.
percentDataGood	See 4.2.1.2.
useSlopedExtrapolation	See 4.2.1.2.

5.3 Aggregates StatusCodes

5.3.1 Overview

Subclause 5.3 defines additional codes and rules that apply to the *StatusCode* when used for *Aggregates*.

The general structure of the *StatusCode* is specified in IEC 62541-4. It includes a set of common operational result codes which also apply to *Aggregates.*

5.3.2 Operation level result codes

In OPC UA *Aggregates* the *StatusCode* is used to indicate the conditions under which a value or *Event* was stored, and thereby can be used as an indicator of its usability. Due to the nature of aggregated data, additional information beyond the basic quality and call result code needs to be conveyed to the client. For example, whether or not the result was *Interpolated*, were all data inputs to a calculation of Good quality, etc.

In the following, Table 8 contains codes with Bad severity indicating a failure; Table 9 contains codes with Uncertain severity indicating that the value has been retrieved under sub-normal conditions. It is important to note that these are the codes that are specific for OPC UA *Aggregates* and that they supplement the codes that apply to all types of data; they are therefore defined in IEC 62541-4, IEC 62541-8 and IEC 62541-11.

Table 8 – Bad operation level result codes

Symbolic Id	Description
Bad_AggregateListMismatch	The requested number of *Aggregates* does not match the requested number of *NodeIds*. When multiple *Aggregates* are requested, a corresponding *NodeId* is required for each *AggregateFunction*.
Bad_AggregateNotSupported	The requested *AggregateFunction* is not supported by the *Server* for the specified *Node*.
Bad_AggregateInvalidInputs	The *Aggregate* value could not be derived due to invalid data inputs, errors attempting to perform data conversions or similar situations.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 9 – Uncertain operation level result codes

Symbolic Id	Description
Uncertain_DataSubNormal	The value is derived from raw values and has less than the required number of Good values.

5.3.3 Aggregate Information Bits

5.3.3.1 General

These bits are set only when obtaining *Aggregate* data. They indicate where the data value came from and provide information that affects how the client uses the data value. Table 10 lists the bit settings which indicate the data location (i.e. is the value stored in the underlying data repository, or is the value the result of data aggregation). These bits are mutually exclusive.

Table 10 – Data location

StatusCode	Description
Raw	A *Raw data* value.
Calculated	A data value which was calculated.
Interpolated	A data value which was interpolated.

In the case where *Interpolated* data is requested, and there is an actual raw value for that timestamp, the *Server* should set the 'Raw' bit in the *StatusCode* of that value.

Table 11 lists the bit settings which indicate additional important information about the data values returned.

Table 11 – Additional information

StatusCode	Description
Partial	A calculated value that is not based on a complete interval. See 5.3.3.2.
Extra Data	If a *Server* chooses to set this bit, it indicates that a *Raw data* value supersedes other data at the same timestamp.
Multiple Values	Multiple values match the *Aggregate* criteria (i.e. multiple minimum values or multiple worst quality at different timestamps within the same *ProcessingInterval*).

The conditions under which these information bits are set depend on how the data has been requested and state of the underlying data repository.

5.3.3.2 Partial Information Bit

Partial bit is used to indicate that the interval is not a complete interval and that a client may receive a different value for the *Aggregate* if it re-fetches the interval with the same parameters.

The *Partial* bit will be set in the following examples:

Assume for these examples the first stored point in the collection is 1:01:10 and the last stored point in the collection is 1:31:20. Older data may exist but is unavailable or offline at the time of the query. Newer data may be available but has not yet been stored in the history collection.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

- The interval that overlaps the beginning of the history collection. If the start time is 1:00:00 and end time is 1:10:00 and the interval is 2 minutes then the first interval would have a partial bit set since it has no data for the first 70 seconds. The *Partial* bit will always be set for the first interval with data if the start time of the interval is before the first data value of the data collection. For intervals prior to the interval with a partial bit, these intervals will be flagged Bad_NoData.
- The interval that overlaps the latest point stored in the history collection. The last point in the collection is 1:31:20 and the historian was not shut down and is still running. A 6-minute interval that started at 1:30:00 would have the *Partial* bit set because the historian is expecting data, but just has not yet received anything. The *Partial* bit will always be set for the last interval with data if the end time of the interval is after the last data value stored in the data collection. Intervals entirely after the interval with a *Partial* bit will be flagged Bad_NoData. For those *Aggregates* with extrapolation, the *Partial* bit may be set. See the *Aggregate* specific characteristics for more details.
- If the start/end time does not result in an even interval and there is additional data beyond the end time then the last interval will have a *Partial* bit. If the start time is 1:00:00 and end time is 1:20:00 and the interval is 6 minutes then the last interval is just 2 minutes long and will have the partial bit set. Extrapolation does not apply in this case.

The *Partial* bit may be set with the Calculated bit when the Calculated bit is always set for the specific *Aggregate*.

5.4 Aggregate details

5.4.1 General

The purpose of Subclause 5.4 is to detail the requirements and behaviour for OPC UA *Servers* supporting *Aggregates*. The intent is to standardize the *Aggregates* so users can reliably predict the results of an *Aggregate* computation and understand its meaning. If users require custom functionality in the *Aggregates*, those *Aggregates* should be written as custom vendor defined *Aggregates*.

The standard *Aggregates* shall be as consistent as possible, meaning that each *Aggregate's* behaviour shall be similar to every other *Aggregate's* behaviour where input parameters, *Raw data*, and boundary conditions are similar. Where possible, the *Aggregates* should deal with input and preconditions in a similar manner.

Subclause 5.4 is divided up into two parts. Subclause 5.4.2 deals with *Aggregate* characteristics and behaviour that are common to all *Aggregates*. Subclause 5.4.3 deals with the characteristics and behaviour of *Aggregates* that are aggregate-specific.

5.4.2 Common characteristics

5.4.2.1 Description

Subclause 5.4.2 deals with *Aggregate* characteristics and behaviour that are common to all *Aggregates.*

5.4.2.2 Generating intervals

To read Historical *Aggregates*, OPC clients shall specify three time parameters:

– startTime (Start)
– endTime (End)
– *ProcessingInterval* (Int)

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

The OPC *Server* shall use these three parameters to generate a sequence of time intervals and then calculate an *Aggregate* for each interval. Subclause 5.4.2.2 specifies, given the three parameters, which time intervals are generated. Table 12 provides information on the intervals for each Start and End time combination. The range is defined to be |End – Start|.

All *Aggregates* return a timestamp of the start of the interval unless otherwise noted for the particular *Aggregate*.

Table 12 – History Aggregate interval information

Start/End Time	Interval	Resulting intervals
Start = *End*	*Int* = Anything	No intervals. Returns a Bad_InvalidArgument *StatusCode*, regardless of whether there is data at the specified time or not.
Start < *End*	*Int* = 0 or *Int* ≥ *Range*	One interval, starting at *Start* and ending at *End*. Includes *Start*, excludes *End*, i.e., *[Start, End)*.
Start < *End*	*Int* ≠ 0, *Int* < *Range*, *Int* divides Range evenly.	*Range/Int* intervals. Intervals are *[Start, Start + Int), [Start + Int, Start + 2 × Int),..., [End – Int, End)*.
Start < *End*	*Int* ≠ 0, *Int* < *Range*, *Int* does not divide *Range* evenly.	⌈*Range/Int*⌉ intervals. Intervals are *[Start, Start + Int), [Start + Int, Start + 2 × Int),..., [Start + (⌊Range/Int⌋– 1) × Int, Start + ⌊Range/Int⌋ × Int), [Start + ⌊Range/Int⌋ × Int, End)*. In other words, the last interval contains the "rest" that remains in the range after taking away ⌊*Range/Int*⌋ intervals of size *Int*.
Start > *End*	*Int* = 0 or *Int* ≥ *Range*	One interval, starting at *Start* and ending at *End*. Includes *Start*, excludes *End*, i.e.,[*Start*, *End*). [a]
Start > *End*	*Int* ≠ 0, *Int* < *Range*, *Int* divides *Range* evenly.	*Range/Int* intervals. Intervals are [*Start, Start- Int), [Start– Int, Start – 2 × Int),..., [End + Int, End)*. [a]
Start > *End*	*Int* ≠ 0, *Int* < *Range*, *Int* does not divide *Range* evenly.	⌈*Range/Int*⌉ intervals. Intervals are [*Start, Start – Int), [Start –Int,Start – 2 × Int),..., [Start – (⌊Range/Int⌋– 1) × Int , Start – ⌊Range/Int⌋ × Int), [Start – ⌊Range/Int⌋ × Int , End)*. In other words, the last interval contains the "rest" that remains in the range after taking away ⌊*Range/Int*⌋ intervals of size *Int* starting at Start. [a]
[a] In this case time is running backwards on the intervals.		

The calculation of all *Aggregates* when time flows backwards is the same as when time flows forwards with the exception that the 'early time' is excluded from the interval and the 'late time' is included. In most cases this means the value will be the same except the timestamps are shifted by one *ProcessingInterval*. E.g. when time flows forward the value at $T = n$ is the same as the value at $T = n + 1$ when time flows backward.

Note that when determining *Aggregates* with *MonitoredItem*, the interval is simply the *ProcessingInterval* parameter as defined in the *AggregateFilter* structure. See IEC 62541-4 for more details.

5.4.2.3 Data types

Table 13 outlines the valid *DataType* for each *Aggregate*. Some *Aggregates* are intended for numeric data types – i.e. integers or real/floating point numbers. Dates, strings, arrays, etc. are not supported. Other *Aggregates* are intended for digital data types – i.e. Boolean or enumerations. In addition some *Aggregates* may return results with a different DataType than those used to calculate the *Aggregate*. Table 13 also outlines the data type returned for each *Aggregate* .

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 13 – Standard History Aggregate Data Type information

BrowseName	Valid Data Type	Result Data Type
	Interpolation Aggregate	
Interpolative	Numeric	Raw Data Type
	Data Averaging Aggregates	
Average	Numeric	Double
TimeAverage	Numeric	Double
TimeAverage2	Numeric	Double
Total	Numeric	Double
Total2	Numeric	Double
	Data Variation Aggregates	
Minimum	Numeric	Raw data type
Maximum	Numeric	Raw data type
MinimumActualTime	Numeric	Raw data type
MaximumActualTime	Numeric	Raw data type
Range	Numeric	Raw data type
Minimum2	Numeric	Raw data type
Maximum2	Numeric	Raw data type
MinimumActualTime2	Numeric	Raw data type
MaximumActualTime2	Numeric	Raw data type
Range2	Numeric	Raw data type
	Counting Aggregates	
AnnotationCount	All	Integer
Count	All	Integer
DurationInStateZero	Numeric or Boolean	Duration
DurationInStateNonZero	Numeric or Boolean	Duration
NumberOfTransitions	Numeric or Boolean	Integer
	Time Aggregates	
Start	All	Raw data type
End	All	Raw data type
Delta	Numeric	Raw data type
StartBound	All	Raw data type
EndBound	All	Raw data type
DeltaBounds	Numeric	Raw data type
	Data Quality Aggregates	
DurationGood	All	Duration
DurationBad	All	Duration
PercentGood	All	Double
PercentBad	All	Double
WorstQuality	All	*StatusCode*
WorstQuality2	All	*StatusCode*
	Statistical Aggregates	
StandardDeviationSample	Numeric	Double
VarianceSample	Numeric	Double
StandardDeviationPopulation	Numeric	Double
VariancePopulation	Numeric	Double

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.2.4 Time calculation issues

The following issues may come up when calculating *Aggregates* that include time as part of the calculation.

- All *Aggregate* calculations include the *startTime* but exclude the *endTime*. However, it is sometimes necessary to return an *Interpolated* End Bound as the value for an *Interval* with a timestamp that is in the *Interval*. *Servers* are expected to use the time immediately before *endTime* where the time resolution of the *Server* determines the exact value (do not confuse this with hardware or operating system time resolution). For example, if the *endTime* is 12:01:00, the time resolution is 1 second, then the *EffectiveEndTime* is 12:00:59. If the *Server* time resolution is 1 millisecond the *EffectiveEndTime* is 12:00:59.999.

  If time is flowing backwards, *Servers* are expected to use the time immediately after *endTime* where the time resolution of the *Server* determines the exact value.
- If there is one data point in the *Interval* and it falls on the *StartTime* the time duration used in calculations is one unit of the time resolution of the *Server*.

5.4.3 Specific aggregated data handling

5.4.3.1 General

When accessing aggregated data using the *HistoryRead* or the *CreateMonitoredItems Service*, the following rules are used to handle specific *Aggregate* use cases.

If *ProcessingInterval* is 0, the *Server* shall create one *Aggregate* value for the entire time range. This allows *Aggregates* over large periods of time. A value with a timestamp equal to *endTime* will be excluded from that *Aggregate*, just as it would be excluded from an interval with that ending time. If the *ProcessingInterval* of 0 is passed in the *MonitoredItemFilter* it shall be revised to a suitable non-zero value.

The timestamp returned with the *Aggregate* shall be the time at the beginning of the interval, except where the *Aggregate* specifies a different timestamp.

If a requested timestamp is set to anything but the source timestamp the operation shall return the *Bad_TimestampToReturnInvalid StatusCode*. If a requested timestamp is not supported in any other way for a *HistoricalDataNode*, the operation shall return the *Bad_TimestampNotSupported StatusCode*. For *MonitoredItem* the *Server* shall not return past data if a requested timestamp is not supported by the history collection.

5.4.3.2 StatusCode calculation

5.4.3.2.1 General

StatusCodes for an *Aggregate* value shall take into account the values used to calculate them. In addition, the configuration parameters *PercentDataGood* and *PercentDataBad* allow the client to control how this calculation is done if supported by the *Server*.

If an *Aggregate* operates on raw values (e.g. Average) the calculation is done by counting values. If an *Aggregate* operates on raw values but can also return a *Bounding Value* then the *Bounding Values* are included in the count when computing the *StatusCode*. If an *Aggregate* does any sort of a time weighted calculation (e.g. TimeAverage or TimeAverage2) then the *StatusCode* calculation shall also be time weighted.

For purposes of calculating time weighted *StatusCodes* each interval shall be divided into regions of Good or Bad data. Creating these regions requires that the *bounding values* be calculated for each interval and the type of bounding value depends on the *Aggregate*.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

If *TreatUncertainAsBad* = False then Uncertain regions are included with the Good regions when calculating the above ratios, if the *TreatUncertainAsBad* = True then the Uncertain regions are included as Bad regions. The *StatusCode* of the value is still treated as Uncertain when the *StatusCode* for the result is calculated. If no Bad regions are in the interval then the *StatusCode* for the interval is Good. For any intervals containing regions where the *StatusCodes* are Bad, the total duration of all *Bad* regions is calculated and divided by the width of the interval. The resulting ratio is multiplied by 100 and compared to the *PercentDataBad* parameter. The *StatusCode* for the interval is Bad if the ratio is greater than or equal to the *PercentDataBad* parameter. For any interval which is not Bad, the total duration of all Good regions is then calculated and divided by the width of the interval. The resulting ratio is multiplied by 100 and compared to the *PercentDataGood* parameter. The *StatusCode* for the interval is Good if the ratio is greater than or equal to the *PercentDataGood* parameter. If for an interval neither ratio applies then that interval is *Uncertain_DataSubNormal*.

If there is no data in the interval and the interval is inside the range [~~StartOfData, EndOfData~~ start of data, end of data] and the *Aggregate* return data type is raw data type then the *StatusCodes* for the interval will be Bad_NoData unless ~~the *Aggregate* specific characteristics determine otherwise~~ an alternate status code is defined for a specific *Aggregate*.

The width of an interval is the *ProcessingInterval* unless it is a partial interval (i.e. has the Partial bit set). In these cases, the width is the time used when calculating the partial interval.

Subclauses 5.4.3.2.2 and 5.4.3.2.3 include diagrams that illustrate a request and data series. The colour of the time axis indicates the status for different regions. Red indicates Bad, green indicates Good and orange indicates Uncertain. These examples assume *TreatUncertainAsBad* = False.

5.4.3.2.2 Sloped Interpolation and Simple Bounding Values

Figure 2 illustrates a data series for *Variable* with Stepped = False and an *Aggregate* that uses *Simple Bounding Values*. The request being processed has a Start Time that falls before the first point in the series and an End Time that does not fall on an integer multiple of the *ProcessingInterval*.

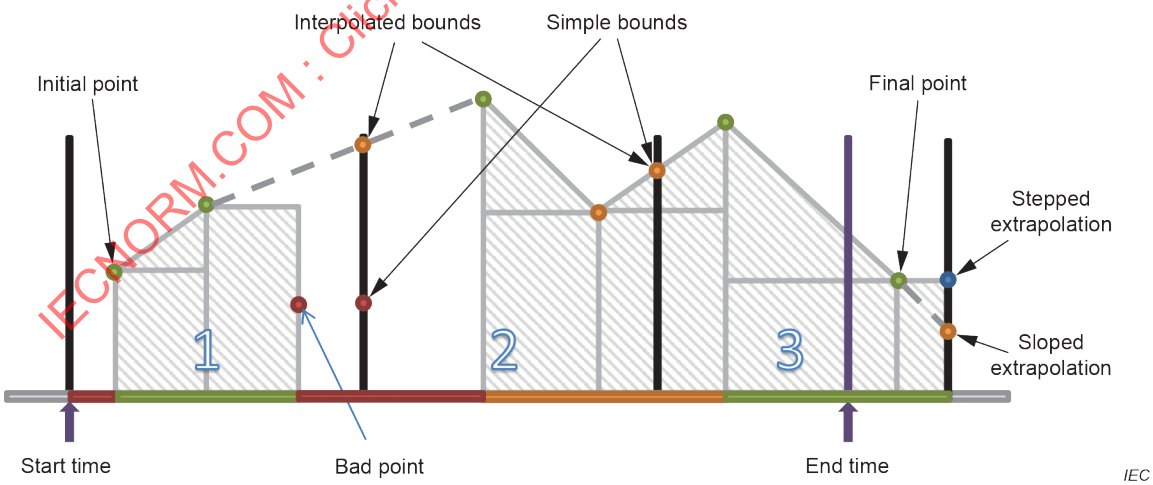


Figure 2 – Variable with Stepped = False and Simple Bounding Values

The first interval has four regions:

- the period before the first data point;
- the period between the first and second where *SlopedInterpolation* can be used;
- the period between the second and third point where *SteppedInterpolation* is used;

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

- the period after the Bad point where no data exists.

A region is Uncertain if a region ends in a Bad or Uncertain value and *SlopedInterpolation* is used. The end point has no effect on the region if *SteppedInterpolation* is used.

The second interval has three regions:

- the period before the first Good data point where no data exists;
- the period between the first and second where *SlopedInterpolation* can be used;
- the period between the second point and the bound calculated with *SlopedInterpolation*.

The third interval has three regions:

- the period between the simple bound and the first data point;
- the period between the first point and an interpolated point that falls on the end time;
- the period after the end time which is ignored.

This is a partial region and the data after the end time is not used, however, if sloped interpolation is used and the point after the endpoint is Uncertain then the region between the last point and the end time will be Uncertain.

5.4.3.2.3 Stepped Interpolation and Interpolated Bounding Values

Figure 3 illustrates a data series for *Variable* with Stepped = True and an *Aggregate* that uses *Interpolated Bounding Values*. The request being processed has a Start Time that falls before the first point in the series and an End Time that does not fall on an integer multiple of the *ProcessingInterval*.

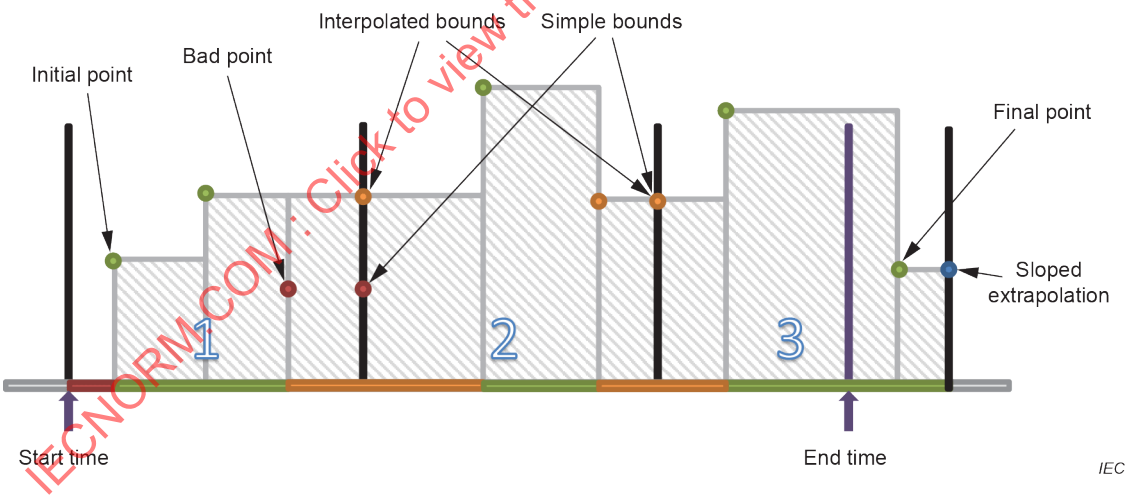


Figure 3 – Variable with Stepped = True and Interpolated Bounding Values

The first interval has three regions:

- the period before the first data point;
- the period between the first and second where *SteppedInterpolation* is used;
- the period between the second and the interpolated end bound.

The Bad point is ignored because of the interpolated end bound but this does create Uncertain regions. If *SlopedInterpolation* was used the Uncertain region would start at the second point. In this case, it only starts when the first Bad value is ignored.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

The second interval has three regions:

- the period between the start bound and the first data point;
- the period between the first and second where *SteppedInterpolation* is used;
- the period between the second and the interpolated end bound.

The third interval has three regions:

- the period between the interpolated bound and the first data point;
- the period between the first point and an interpolated point that falls on the end time;
- the period after the end time which is ignored.

This is a partial region and the data after the end time is not used.

5.4.3.3 Description

Subclause 5.4.3.3 deals with *Aggregate* specific characteristics and behaviour that is specific to a particular *Aggregate*.

Each subclause has a table which formally expresses the *Aggregate* behaviour (including any exceptions). The meaning of each of the fields in the table is described in Table 14.

Description of Table 14:

- The first column is the common name for the item.
- The second column includes a description of the item and a list of the valid selections with for the item including a description of each selection.
- The second part of the table describes how the status associated with the *Aggregate* calculation is computed.
- The last part of the table lists what behaviour is expected from the *Aggregate* for some common special cases. These behaviours require text descriptions so there is no list of valid selections.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 14 – Aggregate table description

Aggregate Characteristics	
Type	The type of *Aggregate*. <Interpolated | Calculated | Raw> Interpolated: See definition for Interpolated. Calculated: Computed from defined calculation. Raw: Selects a raw value from within an interval.
Data Type	The data type of the result. <Double | Int32 | Same as Source>
Use Bounds	How the *Aggregate* deals with bounds. <None | Interpolated | Simple> None: Bounds do not apply to the *Aggregate*. Interpolated: Uses Interpolated Bounds. Simple: Uses Simple Bounds.

Aggregate Characteristics	
Timestamp	What is the time stamp of the resulting *Aggregate* value: <*startTime* | *endTime* | Raw> *startTime*: The time at the start of the interval. *endTime*: The time at the end of the interval. Raw: The time associated with a value in the interval.
StatusCode Calculations	
Calculation Method	How the status code is calculated: <PercentValues | PercentTime | Custom> PercentValues: Based on percentage of value counts. PercentTime: Based on percentage of time interval. Custom: Specific to the *Aggregate* (description included).
Partial	For partial intervals does the *Aggregate* set this bit <Set Sometimes | Not Set> It may also describe any special cases for setting this bit
Calculated	Describes the usage of the calculated bit. <Set Always | Set Sometimes | Not Set> Set Always: The bit is always set. Set Sometimes: The bit is sometimes set (describes when). Not Set: The bit is never set.
Interpolated	Describes the usage of the interpolated bit. <Set Always | Set Sometimes | Not Set> Set Always: The bit is always set. Set Sometimes: The bit is sometimes set (describes when). Not Set: The bit is never set.
Raw	Describes the usage of the Raw bit. <Set Always | Set Sometimes | Not Set> Set Always: The bit is always set. Set Sometimes: The bit is sometimes set (describes when). Not Set: The bit is never set.
Multi Value	Describes the usage of the multi value bit. <Set Sometimes | Not Set> Set Sometimes: The bit is used (see IEC 62541-11). Not Set: The bit is never set.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Aggregate Characteristics	
StatusCode Common Special Cases	
Before Start of Data	If the entire interval is before the start of data.
After End of Data	If the entire interval is after the end of data (as determined by the Historian).
Start Bound Not Found	If the starting bound is not found for the earliest interval and it is not partial, then what, if any, special processing should be done.
End Bound Not Found	If the ending bound is not found for the latest interval and it is not partial, then what, if any, special processing should be done.
Bound Bad	If the Bounding value is Bad, then what, if any, special processing should be done.
Bound Uncertain	If the Bounding value is uncertain, then what, if any, special processing should be done.

5.4.3.4 Interpolative

The Interpolative *Aggregate* defined in Table 15 returns the *Interpolated Bounding Value* (see 3.1.8) for the *startTime* of each interval.

When searching for Good values before or after the bounding value, the time period searched is *Server* specific, but the *Server* should search a time range which is at least the size of the *ProcessingInterval*.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 15 – Interpolative Aggregate summary

Interpolated Aggregate Characteristics	
Type	Interpolated
Data Type	Same as Source
Use Bounds	Interpolated
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Good if no Bad values skipped and Good values are used, Uncertain if Bad values skipped or if Uncertain values are used. If no starting value then *Bad_NoData*. See description of Interpolated Bounds (see 3.1.8) for more details
Partial bit	Not Set
Calculated bit	Not Set
Interpolated bit	Set Sometimes Always set except for when the Raw bit is set
Raw bit	Set Sometimes If a value exists with the exact time of interval Start
Multi Value bit	Not Set
StatusCode Common Special Cases	
Before Start of Data	Return *Bad_NoData*
After End of Data	Return extrapolated value (see 3.1.8) (sloped or stepped according to settings) Status code is *Uncertain_DataSubNormal*.
Start Bound Not Found	*Bad_NoData*.
End Bound Not Found	*See "After End of Data"*
Bound Bad	Does not return a Bad bound except as noted above
Bound Uncertain	Returned *Uncertain_DataSubNormal* if any Bad value(s) was/were skipped to calculate the bounding value.

5.4.3.5 Average

The Average *Aggregate* defined in Table 16 adds up the values of all Good *Raw data* for each interval, and divides the sum by the number of Good values. If any non-Good values are ignored in the computation, the *Aggregate StatusCode* will be determined using the *StatusCode* Calculation (see 5.3). This *Aggregate* is not time based so the PercentGood/PercentBad applies to the number of values in the interval.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 16 – Average Aggregate summary

Average Aggregate Characteristics	
Type	Calculated
Data Type	Double
Use Bounds	None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	PercentValues
Partial	Not Set
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Bounds not used
No End Bound	Bounds not used
Bound Bad	Bounds not used
Bound Uncertain	Bounds not used

5.4.3.6 TimeAverage

The TimeAverage *Aggregate* defined in Table 17 uses *Interpolated Bounding Values* (see 3.1.8) to find the value of a point at the beginning and end of an interval. Starting at the starting bounding value a straight line is drawn between each value in the *interval* ending at the ending bounding value (see examples for illustrations). The area under the lines is divided by the length of the *ProcessingInterval* to yield the average. Note that this calculation always uses a sloped line between points; TimeAverage2 uses a stepped or sloped line depending on the value of the Stepped *Property* for the *Variable*.

If one or more Bad Values exist in the interval then they are omitted from the calculation and the *StatusCode* is set to *Uncertain_DataSubNormal*. Sloped lines are drawn between the Good values when calculating the area.

The time resolution used in this calculation is *Server* specific.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 17 – TimeAverage Aggregate summary

TimeAverage Aggregate Characteristics	
Type	Calculated
Data Type	Double
Use Bounds	Interpolated
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Good if no Bad values skipped and Good values are used, Uncertain if Bad values are skipped or if Uncertain values are used
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Value extrapolated, Uncertain status
No Start Bound	Calculate Partial Interval
No End Bound	Extrapolate data, Uncertain status
Bound Bad	NA
Bound Uncertain	NA

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.7 TimeAverage2

The TimeAverage2 *Aggregate* defined in Table 18 uses *Simple Bounding Values* (see 3.1.9) to find the value of a point at the beginning and end of an interval. Starting at the starting bounding value a straight line is drawn between each value in the interval ending at the ending bounding value (see examples for illustrations). The area under the lines is divided by the length of the *ProcessingInterval* to yield the average. Note that this calculation uses a stepped or sloped line depending on what the value of the Stepped *Property* for the *Variable*; TimeAverage always uses a sloped line between points.

The time resolution used in this calculation is *Server* specific.

If any non-Good data exists in the interval, this data is omitted from the calculation and the time interval is reduced by the duration of the non-Good data; i.e. if a value was Bad for 1 minute in a 5-minute interval then the TimeAverage2 would be the area under the 4-minute period of Good values divided by 4 minutes. If a sub-interval ends at a Bad value then only the Good starting value is used to calculate the area of sub-interval preceding the Bad value.

The *Aggregate StatusCode* will be determined using the *StatusCode* Calculation (see 5.3).

Table 18 – TimeAverage2 Aggregate summary

TimeAverage2 Aggregate Characteristics	
Type	Calculated
Data Type	Double
Use Bounds	Simple
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	PercentTime
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Bound is Bad_NoData and treated as any other Bad value in the interval
No End Bound	Bound is Bad_NoData and treated as any other Bad value in the interval
Bound Bad	Treated as any other Bad value in the interval
Bound Uncertain	Treated as any other Uncertain value in the interval

5.4.3.8 Total

The Total *Aggregate* defined in Table 19 performs the following calculation for each interval:

Total = TimeAverage × *ProcessingInterval* (seconds)

where: TimeAverage is the result from the TimeAverage *Aggregate*, using the *ProcessingInterval* supplied to the Total call.

The resulting units would be normalized to seconds, i.e. [TimeAverage Units] × seconds.

The *Aggregate StatusCode* will be determined using the *StatusCode* Calculation (see 5.3).

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 19 – Total Aggregate summary

Total Aggregate Characteristics	
Type	Calculated
Data Type	Double
Use Bounds	Interpolated
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Good if no Bad values are skipped and Good values are used, Uncertain if Bad values are skipped or if Uncertain values are used
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Value extrapolated, Uncertain status
No Start Bound	Calculate Partial Interval
No End Bound	Extrapolate data, Uncertain status
Bound Bad	NA
Bound Uncertain	NA

5.4.3.9 Total2

The Total2 *Aggregate* defined in Table 20 performs the following calculation for each interval:

Total2 = TimeAverage2 × *ProcessingInterval* of Good data (seconds)

where TimeAverage2 is the result from the TimeAverage2 *Aggregate*, using the *ProcessingInterval* supplied to the Total2 call.

The interval of Good data is the sum of all sub-intervals where non-Bad data exists; i.e. if a value was Bad for 1 minute in a 5-minute interval then the interval of Good data would be the 4-minute period.

The resulting units would be normalized to seconds, i.e. [TimeAverage2 Units] × seconds.

The *Aggregate StatusCode* will be determined using the *StatusCode* Calculation (see 5.3).

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 20 – Total2 Aggregate summary

Total2 Aggregate Characteristics	
Type	Calculated
Data Type	Double
Use Bounds	Simple
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	PercentTime
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Value for Bound is Bad_NoData and is treated like any other Bad quality value in the calculation (ignored)
No End Bound	Value for Bound is Bad_NoData and is treated like any other Bad quality value in the calculation (ignored)
Bound Bad	Value is treated like any other Bad quality value in the calculation (ignored)
Bound Uncertain	Value is treated like any other non-Good quality value in the calculation (ignored)

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.10 Minimum

The Minimum *Aggregate* defined in Table 21 retrieves the minimum Good raw value within the interval, and returns that value with the timestamp at ~~which that value occurs~~ the start of the interval. Note that if the same minimum exists at more than one timestamp~~, the oldest one is retrieved and~~ the MultipleValues bit is set.

Unless otherwise indicated, *StatusCodes* are Good, Calculated. If the minimum value is on the start time the status code will be Good, Raw. If only Bad quality values are available then the status is returned as *Bad_NoData*.

The timestamp of the *Aggregate* will always be the start of the interval for every *ProcessingInterval*.

Table 21 – Minimum Aggregate summary

Minimum Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom If no Bad values then the Status is Good. If Bad values exist then the Status is *Uncertain_SubNormal.* If an Uncertain value is less than the minimum Good value the Status is *Uncertain_SubNormal*.
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Sometimes If the Minimum value is not on the StartTime of the interval or if the Status was set to *Uncertain_SubNormal* because of non-Good values in the interval
Interpolated	Not Set
Raw	Set Sometimes If Minimum value is on the StartTime of the interval
Multi Value	Set Sometimes If multiple Good values exist with the Minimum value
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Not Applicable
No End Bound	Not Applicable
Bound Bad	Not Applicable
Bound Uncertain	Not Applicable

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.11 Maximum

The Maximum *Aggregate* defined in Table 22 retrieves the maximum Good raw value within the interval, and returns that value with the timestamp at ~~which that value occurs~~ the start of the interval. Note that if the same maximum exists at more than one timestamp~~, the oldest one is retrieved and~~ the MultipleValues bit is set.

Unless otherwise indicated, *StatusCodes* are Good, Calculated. If the minimum value is on the interval start time the status code will be Good, Raw. If only Bad quality values are available then the status is returned as *Bad_NoData*.

The timestamp of the *Aggregate* will always be the start of the interval for every *ProcessingInterval*.

Table 22 – Maximum Aggregate summary

Maximum Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom If no Bad values then the Status is Good. If Bad values exist then the Status is *Uncertain_SubNormal.* If an Uncertain value is greater than the maximum Good value the Status is *Uncertain_SubNormal*
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Sometimes If the Maximum value is not on the *startTime* of the interval or if the Status was set to *Uncertain_SubNormal* because of non-Good values in the interval
Interpolated	Not Set
Raw	Set Sometimes If Maximum value is on the *startTime* of the interval
Multi Value	Set Sometimes If multiple Good values exist with the Maximum value
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Not Applicable
No End Bound	Not Applicable
Bound Bad	Not Applicable
Bound Uncertain	Not Applicable

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.12 MinimumActualTime

The MinimumActualTime *Aggregate* defined in Table 23 retrieves the minimum Good raw value within the interval, and returns that value with the timestamp at which that value occurs. Note that if the same minimum exists at more than one timestamp, the oldest one is retrieved and the *Aggregate Bits are* set to MultipleValues.

Table 23 – MinimumActualTime Aggregate summary

MinimumActualTime Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	None
Timestamp	Time of Minimum
StatusCode Calculations	
Calculation Method	Custom If no Bad values then the Status is Good. If Bad values exist then the Status is *Uncertain_SubNormal.* If an Uncertain value is less than the minimum Good value the Status is *Uncertain_SubNormal*
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Sometimes If the Status was set to *Uncertain_SubNormal* because of non-Good values in the interval
Interpolated	Not Set
Raw	Set Sometimes If a Good minimum value is returned
Multi Value	Set Sometimes If multiple Good values exist with the Minimum value
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Not Applicable
No End Bound	Not Applicable
Bound Bad	Not Applicable
Bound Uncertain	Not Applicable

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.13 MaximumActualTime

The MaximumActualTime *Aggregate* defined in Table 24 is the same as the MinimumActualTime *Aggregate*, except that the value is the maximum raw value within the interval. Note that if the same maximum exists at more than one timestamp, the oldest one is retrieved and the *Aggregate Bits are* set to MultipleValues.

Table 24 – MaximumActualTime Aggregate summary

MaximumActualTime Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	None
Timestamp	Time of Maximum
StatusCode Calculations	
Calculation Method	Custom If no Bad values then the Status is Good. If Bad values exist then the Status is *Uncertain_SubNormal.* If an Uncertain value is greater than the maximum Good value the Status is *Uncertain_SubNormal*
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Sometimes If the Status was set to *Uncertain_SubNormal* because of non-Good values in the interval
Interpolated	Not Set
Raw	Set Sometimes If a Good maximum value is returned
Multi Value	Set Sometimes If multiple Good values exist with the maximum value
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Not Applicable
No End Bound	Not Applicable
Bound Bad	Not Applicable
Bound Uncertain	Not Applicable

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.14 Range

The Range *Aggregate* defined in Table 25 finds the difference between the maximum and minimum Good raw values in the interval. If only one *Good* value exists in the interval, the range is zero. Note that the range is always zero or positive. If non-Good values are ignored when finding the minimum or maximum values or if Bad values exist then the status is *Uncertain_DataSubNormal.*

Table 25 – Range Aggregate summary

Range Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom If no Bad values then the Status is Good. If Bad values exist then the Status is *Uncertain_SubNormal.* If an Uncertain value is greater than the maximum or less than the minimum Good value the Status is *Uncertain_SubNormal*
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Not Applicable
No End Bound	Not Applicable
Bound Bad	Not Applicable
Bound Uncertain	Not Applicable

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.15 Minimum2

The Minimum2 *Aggregate* defined in Table 26 retrieves the minimum Good value for each interval as defined for Minimum except that *Simple Bounding Values* are included. The *Simple Bounding Values* for the interval are found according to the definition of *Simple Bounding Values* (see 3.1.9). Any Bad values are ignored in the computation. The *Aggregate StatusCode* will be determined using the *StatusCode* Calculation (see 5.3) for time based *Aggregates*. If a bounding value is returned then the status will indicate, Raw, Calculated or Interpolated.

If *TreatUncertainAsBad* is false and an Uncertain raw value is the minimum then that Uncertain value is used. Uncertain values are ignored otherwise.

If sloped interpolation is used and the End bound is the minimum value then End bound is used as the Minimum with the timestamp set to the *startTime* of the interval. The End bound is ignored in all other cases.

Table 26 – Minimum2 Aggregate summary

Minimum2 Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	Simple
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	PercentTime
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Sometimes Set unless the StartBound is the Minimum
Interpolated	Set Sometimes If an Interpolated bound is the Minimum
Raw	Set Sometimes If a raw value is the Minimum.
Multi Value	Set Sometimes If more than one Good values exist with the same
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Treat the beginning value as Bad_NoData and compute the *Aggregate*
No End Bound	Treat the ending value as Bad_NoData and compute the *Aggregate*
Bound Bad	Use as value and compute the *Aggregate* as defined
Bound Uncertain	Use as value and compute the *Aggregate* as defined

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.16 Maximum2

The Maximum2 *Aggregate* defined in Table 27 retrieves the maximum Good value for each interval as defined for Maximum except that *Simple Bounding Values* are included. The *Simple Bounding Values* for the interval are found according to the definition of *Simple Bounding Values* (see 3.1.9). Any Bad values are ignored in the computation. The *Aggregate StatusCode* will be determined using the *StatusCode* Calculation (see 5.3) for time based *Aggregates*. If a bounding value is returned then the status will indicate, Raw, Calculated or Interpolated.

If *TreatUncertainAsBad* is false and an Uncertain raw value is the maximum then that Uncertain value is used. Uncertain values are ignored otherwise.

If sloped interpolation is used and the End bound is the maximum value then End bound is used as the maximum with the timestamp set to the *startTime* of the interval. The End bound is ignored in all other cases.

Table 27 – Maximum2 Aggregate summary

Maximum2 Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	Simple
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	PercentTime
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Sometimes Set unless the StartBound is the Maximum
Interpolated	Set Sometimes If an Interpolated bound is the Maximum
Raw	Set Sometimes If a raw value is the Maximum.
Multi Value	Set Sometimes If more than one Good values exist with the same
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Treat the beginning value as Bad_NoData and compute the *Aggregate*
No End Bound	Treat the ending value as Bad_NoData and compute the *Aggregate*
Bound Bad	Use as value and compute the *Aggregate* as defined
Bound Uncertain	Use as value and compute the *Aggregate* as defined

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.17 MinimumActualTime2

The MinimumActualTime2 *Aggregate* defined in Table 28 retrieves the minimum Good value for each interval as defined for MinimumActualTime except that *Simple Bounding Values* are included. The *Simple Bounding Values* for the interval are found according to the definition of *Simple Bounding Values* (see 3.1.9). Any Bad values are ignored in the computation. The *Aggregate StatusCode* will be determined using the *StatusCode* Calculation (see 5.3) for time based *Aggregates*. If a bounding value is returned then the status will indicate, Raw, Calculated or Interpolated.

If *TreatUncertainAsBad* is false and an Uncertain raw value is the minimum then that Uncertain value is used. Uncertain values are ignored otherwise.

If sloped interpolation is used and the End bound is the minimum value then End bound is used as the minimum with the timestamp set to the *EffectiveEndTime* of the interval. The End bound is ignored in all other cases.

Table 28 – MinimumActualTime2 Aggregate summary

MinumumActualTime2 Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	Simple
Timestamp	Time of minimum
StatusCode Calculations	
Calculation Method	PercentTime
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Not Set
Interpolated	Set Sometimes If an Interpolated bound is the Minimum
Raw	Set Sometimes If a raw value is the Minimum
Multi Value	Set Sometimes If more than one Good values exist with the same value
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Treat the beginning value as Bad_NoData and compute the *Aggregate*
No End Bound	Treat the ending value as Bad_NoData and compute the *Aggregate*
Bound Bad	Use as value and compute the *Aggregate* as defined
Bound Uncertain	Use as value and compute the *Aggregate* as defined

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.18 MaximumActualTime2

The MaximumActualTime2 *Aggregate* defined in Table 29 retrieves the maximum Good value for each interval as defined for MaximumActualTime except that *Simple Bounding Values* are included. The *Simple Bounding Values* for the interval are found according to the definition of *Simple Bounding Values* (see 3.1.9). Any Bad values are ignored in the computation. The *Aggregate StatusCode* will be determined using the *StatusCode* Calculation (see 5.3) for time based *Aggregates*. If a bounding value is returned then the status will indicate, Raw, Calculated or Interpolated.

If *TreatUncertainAsBad* is false and an Uncertain raw value is the maximum then that Uncertain value is used. Uncertain values are ignored otherwise.

If sloped interpolation is used and the End bound is the maximum value then End bound is used as the maximum with the timestamp set to the EffectiveEndTime of the interval. The End bound is ignored in all other cases.

Table 29 – MaximumActualTime2 Aggregate summary

MaximumActualTime2 Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	Simple
Timestamp	Time of maximum
StatusCode Calculations	
Calculation Method	PercentTime
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Not Set
Interpolated	Set Sometimes If an Interpolated bound is the Maximum
Raw	Set Sometimes If a raw value is the Maximum
Multi Value	Set Sometimes If more than one value is equal to the Maximum
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Treat the beginning value as Bad_NoData and compute the *Aggregate*
No End Bound	Treat the ending value as Bad_NoData and compute the *Aggregate*
Bound Bad	Use as value and compute the *Aggregate* as defined
Bound Uncertain	Use as value and compute the *Aggregate* as defined

5.4.3.19 Range2

The Range2 *Aggregate* defined in Table 30 finds the difference between the maximum and minimum values in the interval as returned by the Minimum2 and Maximum2 *Aggregates*. Note that the range is always zero or positive.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 30 – Range2 Aggregate summary

Range2 Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	Simple (used in Minimum2 and Maximum2 calculations)
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom If Minimum2 or Maximum2 are Bad then the status is Bad_NoData. If Minimum2 or Maximum2 are Uncertain then the status is *Uncertain_DataSubNormal*. Good otherwise
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Handled by Minimum2 and Maximum2
No End Bound	Handled by Minimum2 and Maximum2
Bound Bad	Handled by Minimum2 and Maximum2
Bound Uncertain	Handled by Minimum2 and Maximum2

5.4.3.20 AnnotationCount

The AnnotationCount *Aggregate* defined in Table 31 returns a count of all *Annotations* in the interval.

The *StatusCodes* are Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 31 – AnnotationCount Aggregate summary

AnnotationCount Aggregate Characteristics	
Type	Calculated
Data Type	Int32 (negative values are not allowed)
Use Bounds	None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Good unless the interval is before the start of data or after the end of data
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Does not apply
No End Bound	Does not apply
Bound Bad	Does not apply
Bound Uncertain	Does not apply

5.4.3.21 Count

The Count *Aggregate* defined in Table 32 retrieves a count of all the raw values within an interval. If one or more raw values are non-Good, they are not included in the count, and the *Aggregate StatusCode* is determined using the *StatusCode* Calculation (see 5.4.3) for non-time based *Aggregates*. If no Good data exists for an interval, the count is zero.

Unless otherwise indicated, *StatusCodes* are Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 32 – Count Aggregate summary

Count Aggregate Characteristics	
Type	Calculated
Data Type	Int32 (negative values are not allowed)
Use Bounds	None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	PercentValues
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Does not apply
No End Bound	Does not apply
Bound Bad	Does not apply
Bound Uncertain	Does not apply

5.4.3.22 DurationInStateZero

The DurationInStateZero *Aggregate* defined in Table 33 returns the time *Duration* during the interval that the *Variable* was in the zero state. The *Simple Bounding Values* for the interval are used to determine initial value (start time < end time) or ending value (if start time > end time). If one or more raw values are non-Good, they are not included in the *Duration*, and the *Aggregate StatusCode* is determined using the *StatusCode* Calculation (see 5.3) for time based *Aggregates*. *Duration* is in milliseconds. Unless otherwise indicated, *StatusCodes* are Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 33 – DurationInStateZero Aggregate summary

DurationInStateZero Aggregate Characteristics	
Type	Calculated
Data Type	Duration
Use Bounds	Simple
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	PercentTime
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Treat the beginning value as Bad_NoData and compute the *Aggregate*
No End Bound	Treat the ending value as Bad_NoData and compute the *Aggregate*
Bound Bad	Use as value and compute the *Aggregate* as defined
Bound Uncertain	Use as value and compute the *Aggregate* as defined

5.4.3.23 DurationInStateNonZero

The DurationInStateNonZero *Aggregate* defined in Table 34 returns the time *Duration* during the interval that the *Variable* was in the one state. The *Simple Bounding Values* for the interval are used to determine initial value (start time < end time) or ending value (if start time > end time). If one or more raw values are non-Good, they are not included in the *Duration*, and the *Aggregate StatusCode* is determined using the *StatusCode* Calculation (see 5.3) for time based *Aggregates*.

Duration is in milliseconds. Unless otherwise indicated, *StatusCodes* are Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 34 – DurationInStateNonZero Aggregate Summary

DurationInStateNonZero Aggregate Characteristics	
Type	Calculated
Data Type	Duration
Use Bounds	Simple
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	PercentTime
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Treat the beginning value as Bad_NoData and compute the *Aggregate*
No End Bound	Treat the ending value as Bad_NoData and compute the *Aggregate*
Bound Bad	Use as value and compute the *Aggregate* as defined
Bound Uncertain	Use as value and compute the *Aggregate* as defined

5.4.3.24 NumberOfTransitions

The NumberOfTransitions *Aggregate* defined in Table 35 returns a count of the number of transition the *Variable* had during the interval. If one or more raw values are Bad, they are not included in the count, and the *Aggregate StatusCode* is determined using the *StatusCode* Calculation (see 5.3) for non-time based *Aggregates*.

The earliest transition shall be calculated by comparing the earliest non-Bad value in the interval to the previous non-Bad value. A transition occurred if no previous non-Bad value exists or if the earliest non-Bad value is different. The *endTime* is not considered part of the interval, so a transition occurring at the *endTime* is not included.

Unless otherwise indicated, *StatusCodes* are Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 35 – NumberOfTransitions Aggregate summary

NumberOfTransitions Aggregate Characteristics	
Type	Calculated
Data Type	Int32 (negative values are not allowed)
Use Bounds	Custom, a non-Bad value prior to the interval is used
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	PercentValues
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Treat the beginning value as Bad_NoData and compute the *Aggregate*
No End Bound	Treat the ending value as Bad_NoData and compute the *Aggregate*
Bound Bad	Use as value and compute the *Aggregate* as defined
Bound Uncertain	Use as value and compute the *Aggregate* as defined

5.4.3.25 Start

The Start *Aggregate* defined in Table 36 retrieves the earliest raw value within the interval, and returns that value and status with the timestamp at which that value occurs. If no values are in the interval then the *StatusCode* is *Bad_NoData*.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 36 – Start Aggregate summary

Start Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	None
Timestamp	Time of Raw Value
StatusCode Calculations	
Calculation Method	Custom The raw value status is returned
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Not Set
Interpolated	Not Set
Raw	Always
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Does not apply
No End Bound	Does not apply
Bound Bad	Does not apply
Bound Uncertain	Does not apply

5.4.3.26 End

The End *Aggregate* defined in Table 37 retrieves the latest raw value within the interval, and returns that value and status with the timestamp at which that value occurs. If no values are in the interval then the *StatusCode* is *Bad_NoData*.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 37 – End Aggregate summary

End Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	None
Timestamp	Time of Raw Value
StatusCode Calculations	
Calculation Method	Custom The raw value status is returned
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Not Set
Interpolated	Not Set
Raw	Always
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Does not apply
No End Bound	Does not apply
Bound Bad	Does not apply
Bound Uncertain	Does not apply

5.4.3.27 Delta

The Delta *Aggregate* defined in Table 38 retrieves the difference between the earliest and latest Good raw values in the interval. The *Aggregate* is negative if the latest value is less than the earliest value. The status is *Uncertain_DataSubNormal* if non-Good values are skipped while looking for the first or last values. The status is *Good* otherwise. The status is *Bad_NoData* if no Good raw values exist.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 38 – Delta Aggregate summary

Delta Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom *Uncertain_DataSubNormal* if non-Good values are skipped while looking for the first or last values
Partial	Set Sometimes If an interval is not a complete interval
Calculated	~~Not~~ Set Always
Interpolated	Not Set
Raw	Always
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Does not apply
No End Bound	Does not apply
Bound Bad	Does not apply
Bound Uncertain	Does not apply

5.4.3.28 StartBound

The StartBound *Aggregate* defined in Table 39 returns the value and status at the *StartTime* for the interval by calculating the *Simple Bounding Values* for the interval (see 3.1.9).

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 39 – StartBound Aggregate summary

StartBound Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	Simple
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom The status of the start bound.
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Not Set
Interpolated	Set Sometimes If the bound is interpolated
Raw	Set Sometimes If a value exists at the start time
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Bad_NoData
No End Bound	Does not apply
Bound Bad	Same as bound
Bound Uncertain	Same as bound

5.4.3.29 EndBound

The EndBound *Aggregate* defined in Table 40 returns the value and status at the *EndTime* for the interval by calculating the *Simple Bounding Values* for the interval (see 3.1.9).

The timestamp returned is always the start of the interval and Calculated bit is set.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 40 – EndBound Aggregate summary

EndBound Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	Simple
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom The status of the end bound.
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Does not apply
No End Bound	Bad_NoData
Bound Bad	Same as bound
Bound Uncertain	Same as bound

5.4.3.30 DeltaBounds

The DeltaBounds *Aggregate* defined in Table 41 returns the difference between the *StartBound* and the *EndBound Aggregates* with the exception that both the start and end shall be Good. If the end value is less than the start value, the result will be negative. If the end value is the same as the start value the result will be zero. If the end value is greater than the start value, the result will be positive. If one or both values are Bad the return status will be *Bad_NoData*. If one or both values are Uncertain the status will be *Uncertain_DataSubNormal*.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 41 – DeltaBounds Aggregate summary

DeltaBounds Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	Simple
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Good if both bounds are Good *Uncertain_DataSubNormal* if either bound is uncertain Bad_NoData if either bound is Bad
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Bad_NoData
No End Bound	Bad_NoData
Bound Bad	Bad_NoData
Bound Uncertain	Uncertain_DataSubNormal

5.4.3.31 DurationGood

The DurationGood *Aggregate* defined in Table 42 divides the interval into regions of Good and non-Good data. Each region starts with a data point in the interval. If that data point is Good the region is Good. The *Aggregate* is the sum of the duration of all Good regions expressed in milliseconds.

The status of the first region is determined by finding the first data point at or before the start of the interval. If no value exists, the first region is Bad.

Each *Aggregate* is returned with timestamp of the start of the interval. *StatusCodes* are Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 42 – DurationGood Aggregate summary

DurationGood Aggregate Characteristics	
Type	Calculated
Data Type	Duration
Use Bounds	Uses status of bounding value
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom *StatusCode* is always *Good*, *Calculated*
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	No special handing required
No End Bound	No special handing required
Bound Bad	No special handing required
Bound Uncertain	No special handing required

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.32 DurationBad

The DurationBad *Aggregate* defined in Table 43 divides the interval into regions of Bad and non-Bad data. Each region starts with a data point in the interval. If that data point is Bad the region is Bad. The *Aggregate* is the sum of the duration of all Bad regions expressed in milliseconds.

The status of the first region is determined by finding the first data point at or before the start of the interval. If no value exists, the first region is Bad.

Each *Aggregate* is returned with timestamp of the start of the interval. *StatusCodes* are Good, Calculated.

Table 43 – DurationBad Aggregate summary

DurationBad Aggregate Characteristics	
Type	Calculated
Data Type	Duration
Use Bounds	Uses status of bounding value
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom *StatusCode* is always *Good, Calculated*
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	No special handing required
No End Bound	No special handing required
Bound Bad	No special handing required
Bound Uncertain	No special handing required

5.4.3.33 PercentGood

The PercentGood *Aggregate* defined in Table 44 performs the following calculation:

PercentGood = DurationGood / *ProcessingInterval* × 100

where:

DurationGood is the result from the DurationGood *Aggregate*, calculated using the *ProcessingInterval* supplied to *PercentGood* call.

ProcessingInterval is the duration of interval.

If the last interval is a partial interval then the duration of the partial interval is used in the calculation. Each *Aggregate* is returned with timestamp of the start of the interval. *StatusCodes* are Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 44 – PercentGood Aggregate summary

PercentGood Aggregate Characteristics	
Type	Calculated
Data Type	Double (percent)
Use Bounds	Simple (used in DurationGood calculation)
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Always Good
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	No special handing required
No End Bound	No special handing required
Bound Bad	No special handing required
Bound Uncertain	No special handing required

5.4.3.34 PercentBad

The PercentBad *Aggregate* defined in Table 45 performs the following calculation:

PercentBad = DurationBad / *ProcessingInterval* × 100

where:

DurationBad is the result from the DurationBad *Aggregate*, calculated using the *ProcessingInterval* supplied to *PercentBad* call.

ProcessingInterval is the duration of interval.

If the last interval is a partial interval then the duration of the partial interval is used in the calculation. Each *Aggregate* is returned with timestamp of the start of the interval. *StatusCodes* are Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 45 – PercentBad Aggregate summary

PercentBad Aggregate Characteristics	
Type	Calculated
Data Type	Double (percent)
Use Bounds	Simple (used in DurationBad calculation)
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Always Good.
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	No special handing required
No End Bound	No special handing required
Bound Bad	No special handing required
Bound Uncertain	No special handing required

5.4.3.35 WorstQuality

The WorstQuality *Aggregate* defined in Table 46 returns the worst status of the raw values in the interval where a Bad status is worse than Uncertain, which is worse than Good. No distinction is made between the specific reasons for the status.

If multiple values exist with the worst quality but different *StatusCodes* then the *StatusCode* of the first value is returned and the *MultipleValues* bit is set.

This *Aggregate* returns the worst *StatusCode* as the value of the *Aggregate*.

The timestamp is always the start of the interval. The *StatusCodes* are Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 46 – WorstQuality Aggregate summary

WorstQuality Aggregate Characteristics	
Type	Calculated
Data Type	StatusCode
Use Bounds	None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Always Good
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Used
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	No special handing required
No End Bound	No special handing required
Bound Bad	No special handing required
Bound Uncertain	No special handing required

5.4.3.36 WorstQuality2

The *WorstQuality2 Aggregate* defined in Table 47 returns the worst status of the raw values in the interval where a Bad status is worse than Uncertain, which is worse than Good. No distinction is made between the specific reasons for the status.

The start bound calculated using *Simple Bounding Values* (see 3.1.9) is always included when determining the worst quality.

If multiple values exist with the worst quality but different *StatusCodes* then the *StatusCode* of the first value is returned and the *MultipleValues* bit is set.

This *Aggregate* returns the worst *StatusCode* as the value of the *Aggregate*.

The timestamp is always the start of the interval. The *StatusCodes* are Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 47 – WorstQuality2 Aggregate summary

WorstQuality2 Aggregate Characteristics	
Type	Calculated
Data Type	StatusCode
Use Bounds	Simple
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Always Good
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Used
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	No special handing required
No End Bound	No special handing required
Bound Bad	No special handing required
Bound Uncertain	No special handing required

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.37 StandardDeviationSample

The *StandardDeviationSample Aggregate* defined in Table 48 uses the formula:

$$\sqrt{\frac{\sum_{1}^{n}(X - \text{Avg}(X))^2}{n-1}}$$

where X is each Good raw value in the interval, Avg(X) is the average of the Good raw values, and n is the number of Good raw values in the interval.

For every interval where n = 1, a value of 0 is returned.

If any non-Good values were ignored, the *Aggregate* quality is uncertain/subnormal.

All interval *Aggregates* return timestamp of the start of the interval. Unless otherwise indicated, qualities are *Good, Calculated*.

This calculation is for a sample population where the calculation is done on a subset of the full set of data. Use *StandardDeviationPopulation* to calculate the standard deviation of a full set of data (see 5.4.3.39). An example would be when the underlying data is sampled from the data source versus stored on an exception basis.

Table 48 – StandardDeviationSample Aggregate summary

StandardDeviationSample Aggregate Characteristics	
Type	Calculated
Data Type	StatusCode
Use Bounds	~~Simple~~ None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Always Good
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	No special handing required
No End Bound	No special handing required
Bound Bad	No special handing required
Bound Uncertain	No special handing required

5.4.3.38 VarianceSample

The *VarianceSample Aggregate* defined in Table 49 retrieves the square of the standard deviation. Its behaviour is the same as the *StandardDeviationSample Aggregate*. Unless otherwise indicated, qualities are *Good, Calculated*.

This calculation is for a sample population where the calculation is done on a subset of the full population. Use *VariancePopulation* to calculate the variance of a full set of data (5.4.3.40).

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 49 – VarianceSample Aggregate summary

VarianceSample Aggregate Characteristics	
Type	Calculated
Data Type	StatusCode
Use Bounds	~~Simple~~ None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Always Good
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	No special handing required
No End Bound	No special handing required
Bound Bad	No special handing required
Bound Uncertain	No special handing required

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.39 StandardDeviationPopulation

The *StandardDeviation Population Aggregate* defined in Table 50 uses the formula:

$$\sqrt{\frac{\sum_{1}^{n}(X - \text{Avg}(X))^2}{n}}$$

where X is each Good raw value in the interval, Avg(X) is the average of the Good raw values, and n is the number of Good raw values in the interval.

For every interval where n = 1, a value of 0 is returned.

If any non-Good values were ignored, the *Aggregate* quality is uncertain/subnormal.

All interval *Aggregates* return timestamp of the start of the interval. Unless otherwise indicated, qualities are *Good, Calculated*.

This calculation is for a full population where the calculation is done on the full set of data. Use *StandardDeviationSample* to calculate the standard deviation of a subset of the full population (5.4.3.37). An example would be when the underlying data is collected on an exception basis versus sampled from the data source.

Table 50 – StandardDeviationPopulation Aggregate summary

StandardDeviationPopulation Aggregate Characteristics	
Type	Calculated
Data Type	StatusCode
Use Bounds	~~Simple~~ None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Always Good
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	No special handing required
No End Bound	No special handing required
Bound Bad	No special handing required
Bound Uncertain	No special handing required

5.4.3.40 VariancePopulation

The *VariancePopulation Aggregate* defined in Table 51 retrieves the square of the standard deviation. Its behaviour is the same as the *StandardDeviationPopulation Aggregate*. Unless otherwise indicated, qualities are *Good, Calculated*.

This calculation is for a full population where the calculation is done on the full set of data. Use *VarianceSample* to calculate the variance of a subset of the full population (5.4.3.38).

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 51 – VariancePopulation Aggregate summary

VariancePopulation Aggregate Characteristics	
Type	Calculated
Data Type	StatusCode
Use Bounds	~~Simple~~ None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Always Good
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	No special handing required
No End Bound	No special handing required
Bound Bad	No special handing required
Bound Uncertain	No special handing required

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Annex A
(informative)

Aggregate Specific examples – Historical Access

A.1 Historical Aggregate specific characteristics

A.1.1 Example Aggregate data – Historian 1

For the purposes of Historian 1 examples consider a source historian with the ~~following~~ data given in Figure A.1:

Timestamp	Value	StatusCode	Notes
12:00:00	-	Bad_NoData	First archive entry, Point created
12:00:10	10	Raw, Good	
12:00:20	20	Raw, Good	
12:00:30	30	Raw, Good	
12:00:40	40	Raw, Bad	**ANNOTATION**: Operator 1 Jan-02-2012 8:00:00 Scan failed, Bad data entered **ANNOTATION:** Jan-04-2012 7:10:00 Value cannot be verified
12:00:50	50	Raw, Good	**ANNOTATION**: Engineer1 Jan-04-2012 7:00:00 Scanner fixed
12:01:00	60	Raw, Good	
12:01:10	70	Raw, Uncertain	**ANNOTATION: Technician_1** **Jan-02-2012 8:00:00 Value flagged as questionable**
12:01:20	80	Raw, Good	
12:01:30	90	Raw, Good	
	null	No Data	No more entries, awaiting next scan

The example historian also has *Annotations* associated with three data points.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

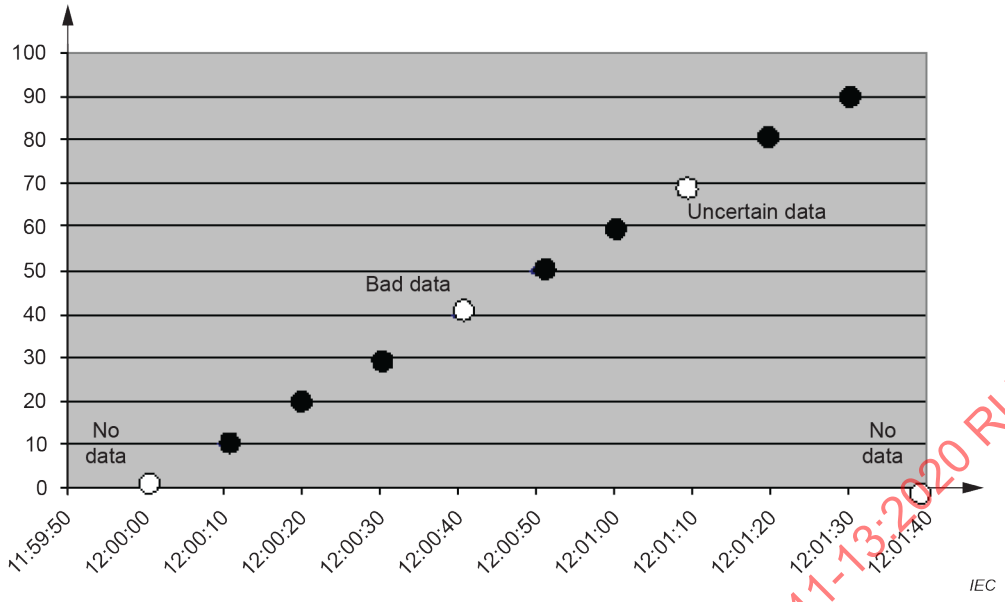


Figure A.1 – Historian 1

For the purposes of all Historian 1 examples:

1) *TreatUncertainAsBad* = False. Therefore Uncertain values are included in *Aggregate* calls.
2) *Stepped Attribute* = False. Therefore *SlopedInterpolation* is used between data points.
3) *UseSlopedExtrapolation* = False. Therefore *SteppedExtrapolation* is used at end boundary conditions.
4) *PercentBad* = 100, *PercentGood* = 100. Therefore if all values are Good then the quality will be Good, or if all values are Bad then the quality will be Bad, but if there is some Good and some Bad then the quality will be Uncertain.

A.1.2 Example Aggregate data – Historian 2

This example is included to illustrate non-periodic data. For the purposes of Historian 2 examples consider a source historian with the ~~following~~ data shown in Figure A.2:

Timestamp	Value	StatusCode	Notes
12:00:00	-	Bad_NoData	First archive entry, Point created
12:00:02	10	Raw, Good	
12:00:25	20	Raw, Good	
12:00:28	25	Raw, Good	
12:00:39	30	Raw, Good	
12:00:42	-	Raw, Bad	Bad quality data received, Bad data entered
12:00:48	40	Raw, Good	Received Good *StatusCode* value
12:00:52	50	Raw, Good	
12:01:12	60	Raw, Good	
12:01:17	70	Raw, Uncertain	Value is flagged as questionable
12:01:23	70	Raw, Good	
12:01:26	80	Raw, Good	
12:01:30	90	Raw, Good	
	-	No Data	No more entries, awaiting next Value

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

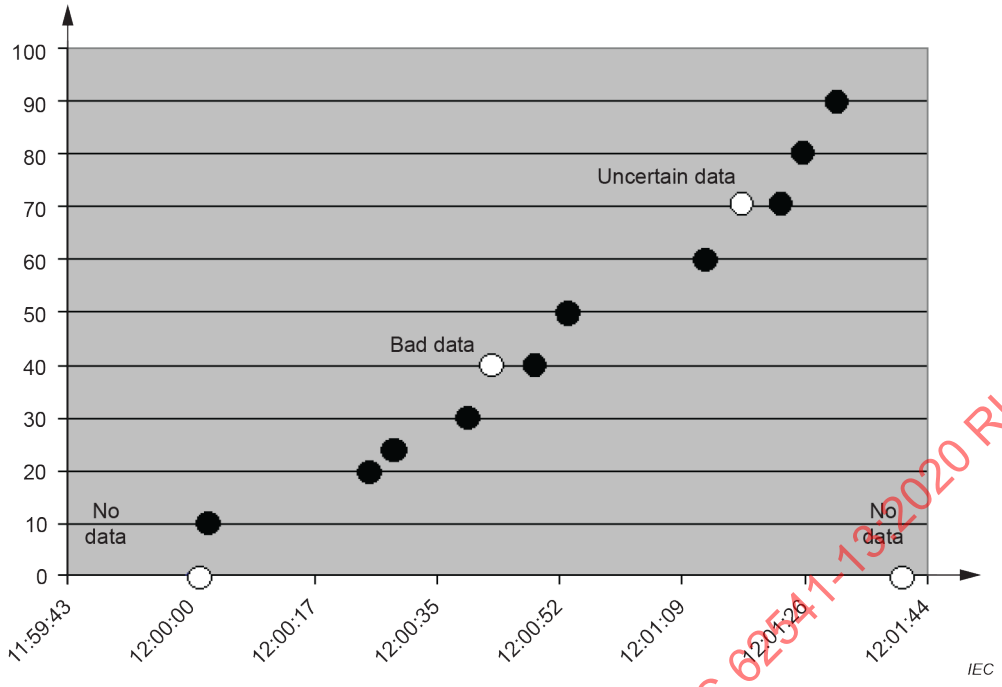


Figure A.2 – Historian 2

For the purposes of all Historian 2 examples:

1) *TreatUncertainAsBad* = True. Therefore Uncertain values are treated as Bad, and not included in the *Aggregate* call.
2) *Stepped Attribute* = False. Therefore *SlopedInterpolation* is used between data points.
3) *UseSlopedExtrapolation* = False. Therefore *SteppedExtrapolation* is used at end boundary conditions.
4) *PercentBad* = 100, *PercentGood* = 100. Therefore unless if all values are Good then the quality will be Good, or if all values are Bad then the quality will be Bad, but if there is some Good and some Bad then the quality will be Uncertain.

A.1.3 Example Aggregate data – Historian 3

This example is included to illustrate stepped data. For the purposes of Historian 3 examples consider a source historian with the ~~following~~ data given in Figure A.3:

Timestamp	Value	StatusCode	Notes
12:00:00	-	Bad_NoData	First archive entry, Point created
12:00:02	10	Raw, Good	
12:00:25	20	Raw, Good	
12:00:28	25	Raw, Good	
12:00:39	30	Raw, Good	
12:00:42	-	Raw, Bad	Bad quality data received, Bad data entered
12:00:48	40	Raw, Good	Received Good *StatusCode* value
12:00:52	50	Raw, Good	
12:01:12	60	Raw, Good	
12:01:17	70	Raw, Uncertain	Value is flagged as questionable
12:01:23	70	Raw, Good	
12:01:26	80	Raw, Good	
12:01:30	90	Raw, Good	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Timestamp	Value	StatusCode	Notes
	-	No Data	No more entries, awaiting next Value

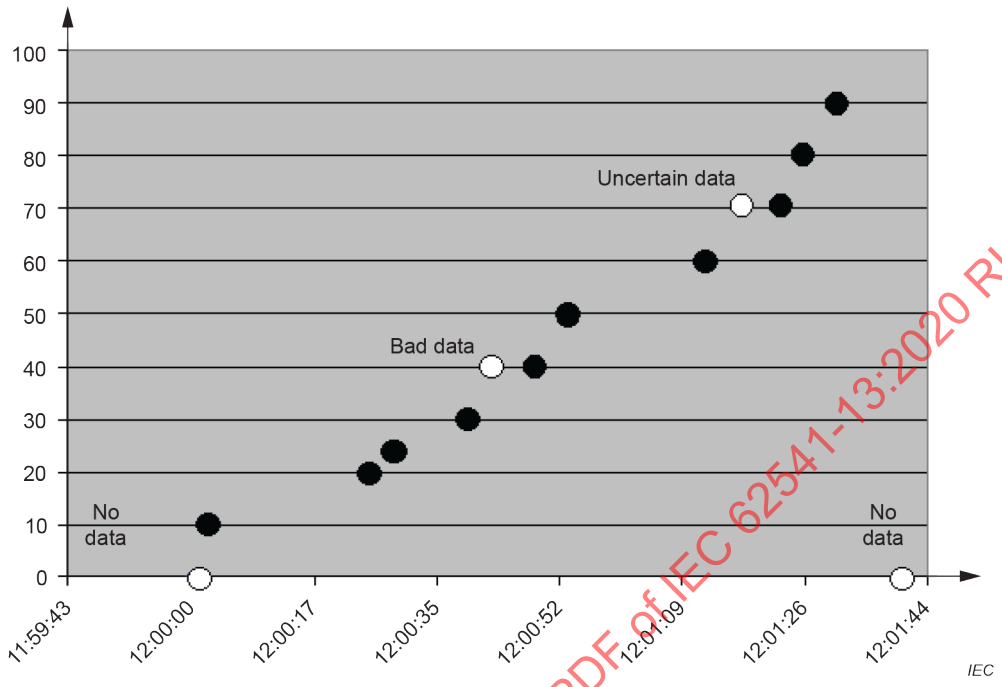


Figure A.3 – Historian 3

For the purposes of all Historian 3 examples:

1) *TreatUncertainAsBad* = True. Therefore Uncertain values are treated as Bad, and not included in the *Aggregate* call.
2) *Stepped Attribute* = True. Therefore *SteppedInterpolation* is used between data points.
3) *UseSlopedExtrapolation* = False. Therefore *SteppedExtrapolation* is used at end boundary conditions.
4) *PercentBad* = 50, *PercentGood* = 50. Therefore data will be either Good or Bad quality. Uncertain should not happen since a value is either Good or Bad.

A.1.4 Example Aggregate data – Historian 4

This example is included to illustrate Boolean data. For the purposes of Historian 4 examples consider a source historian with the following data:

Timestamp	Value	StatusCode	Notes
12:00:00	-	Bad_NoData	First archive entry, Point created
12:00:02	TRUE	Raw, Good	
12:00:25	FALSE	Raw, Good	
12:00:28	TRUE	Raw, Good	
12:00:39	TRUE	Raw, Good	
12:00:42	-	Raw, Bad	Bad quality data received, Bad data entered
12:00:48	TRUE	Raw, Good	Received Good *StatusCode* value
12:00:52	FALSE	Raw, Good	
12:01:12	FALSE	Raw, Good	
12:01:17	TRUE	Raw, Uncertain	Value is flagged as questionable

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Timestamp	Value	StatusCode	Notes
12:01:23	TRUE	Raw, Good	
12:01:26	FALSE	Raw, Good	
12:01:30	TRUE	Raw, Good	
	-	No Data	No more entries, awaiting next Value

For the purposes of all Historian 4 examples:

1) *TreatUncertainAsBad* = True. Therefore Uncertain values are treated as Bad, and not included in the *Aggregate* call.
2) *Stepped Attribute* = True. Therefore *SteppedInterpolation* is used between data points.
3) *UseSlopedExtrapolation* = False. Therefore *SteppedExtrapolation* is used at end boundary conditions.
4) *PercentGood = 100, PercentBad = 100.*

For Boolean data interpolation and extrapolation shall always be stepped.

A.2 Interpolative

A.2.1 Description

The following examples demonstrate Interpolative *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:05, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.2.2 Interpolative data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData	
12:00:05.000		BadNoData	
12:00:10.000	10	Good	
12:00:15.000	15	Good, Interpolated	
12:00:20.000	20	Good	
12:00:25.000	25	Good, Interpolated	
12:00:30.000	30	Good	
12:00:35.000	35	UncertainDataSubNormal, Interpolated	
12:00:40.000	40	UncertainDataSubNormal, Interpolated	
12:00:45.000	45	UncertainDataSubNormal, Interpolated	
12:00:50.000	50	Good	
12:00:55.000	55	Good, Interpolated	
12:01:00.000	60	Good	
12:01:05.000	65	UncertainDataSubNormal, Interpolated	
12:01:10.000	70	Uncertain	
12:01:15.000	75	UncertainDataSubNormal, Interpolated	
12:01:20.000	80	Good	
12:01:25.000	85	Good, Interpolated	
12:01:30.000	90	Good	
12:01:35.000	90	UncertainDataSubNormal, Interpolated	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData	
12:00:05.000	11.304	Good, Interpolated	
12:00:10.000	13.478	Good, Interpolated	
12:00:15.000	15.652	Good, Interpolated	
12:00:20.000	17.826	Good, Interpolated	
12:00:25.000	20	Good	
12:00:30.000	25.909	Good, Interpolated	
12:00:35.000	28.182	Good, Interpolated	
12:00:40.000	31.111	UncertainDataSubNormal, Interpolated	
12:00:45.000	36.667	UncertainDataSubNormal, Interpolated	
12:00:50.000	45	Good, Interpolated	
12:00:55.000	51.500	Good, Interpolated	
12:01:00.000	54	Good, Interpolated	
12:01:05.000	56.500	Good, Interpolated	
12:01:10.000	59	Good, Interpolated	
12:01:15.000	62.727	UncertainDataSubNormal, Interpolated	
12:01:20.000	67.273	UncertainDataSubNormal, Interpolated	
12:01:25.000	76.667	Good, Interpolated	
12:01:30.000	90	Good	
12:01:35.000	~~102.500~~ 90	UncertainDataSubNormal, Interpolated	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData	
12:00:05.000	10	Good, Interpolated	
12:00:10.000	10	Good, Interpolated	
12:00:15.000	10	Good, Interpolated	
12:00:20.000	10	Good, Interpolated	
12:00:25.000	20	Good	
12:00:30.000	25	Good, Interpolated	
12:00:35.000	25	Good, Interpolated	
12:00:40.000	30	Good, Interpolated	
12:00:45.000	30	UncertainDataSubNormal, Interpolated	
12:00:50.000	40	Good, Interpolated	
12:00:55.000	50	Good, Interpolated	
12:01:00.000	50	Good, Interpolated	
12:01:05.000	50	Good, Interpolated	
12:01:10.000	50	Good, Interpolated	
12:01:15.000	60	Good, Interpolated	
12:01:20.000	60	UncertainDataSubNormal, Interpolated	
12:01:25.000	70	Good, Interpolated	
12:01:30.000	90	Good	
12:01:35.000	90	UncertainDataSubNormal, Interpolated	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.3 Average

A.3.1 Description

The following examples demonstrate Average *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:05, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.3.2 Average data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData	
12:00:05.000		BadNoData	
12:00:10.000	10	Good, Calculated	
12:00:15.000		BadNoData	
12:00:20.000	20	Good, Calculated	
12:00:25.000		BadNoData	
12:00:30.000	30	Good, Calculated	
12:00:35.000		BadNoData	
12:00:40.000		BadNoData	
12:00:45.000		BadNoData	
12:00:50.000	50	Good, Calculated	
12:00:55.000		BadNoData	
12:01:00.000	60	Good, Calculated	
12:01:05.000		BadNoData	
12:01:10.000		BadNoData	
12:01:15.000		BadNoData	
12:01:20.000	80	Good, Calculated	
12:01:25.000		BadNoData	
12:01:30.000	90	Good, Calculated	
12:01:35.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated	
12:00:05.000		BadNoData	
12:00:10.000		BadNoData	
12:00:15.000		BadNoData	
12:00:20.000		BadNoData	
12:00:25.000	22.500	Good, Calculated	
12:00:30.000		BadNoData	
12:00:35.000	30	Good, Calculated	
12:00:40.000		BadNoData	
12:00:45.000	40	Good, Calculated	
12:00:50.000	50	Good, Calculated	
12:00:55.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:01:00.000		BadNoData	
12:01:05.000		BadNoData	
12:01:10.000	60	Good, Calculated	
12:01:15.000		BadNoData	
12:01:20.000	70	Good, Calculated	
12:01:25.000	80	Good, Calculated	
12:01:30.000	90	Good, Calculated	
12:01:35.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated	
12:00:05.000		BadNoData	
12:00:10.000		BadNoData	
12:00:15.000		BadNoData	
12:00:20.000		BadNoData	
12:00:25.000	22.500	Good, Calculated	
12:00:30.000		BadNoData	
12:00:35.000	30	Good, Calculated	
12:00:40.000		BadNoData	
12:00:45.000	40	Good, Calculated	
12:00:50.000	50	Good, Calculated	
12:00:55.000		BadNoData	
12:01:00.000		BadNoData	
12:01:05.000		BadNoData	
12:01:10.000	60	Good, Calculated	
12:01:15.000		BadNoData	
12:01:20.000	70	Good, Calculated	
12:01:25.000	80	Good, Calculated	
12:01:30.000	90	Good, Calculated	
12:01:35.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.4 TimeAverage

A.4.1 Description

The following examples demonstrate TimeAverage *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:05, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.4.2 TimeAverage data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData	
12:00:05.000		BadNoData	
12:00:10.000	12.500	Good, Calculated	
12:00:15.000	17.500	Good, Calculated	
12:00:20.000	22.500	Good, Calculated	
12:00:25.000	27.500	Good, Calculated	
12:00:30.000	32.500	UncertainDataSubNormal, Calculated	
12:00:35.000	37.500	UncertainDataSubNormal, Calculated	
12:00:40.000	42.500	UncertainDataSubNormal, Calculated	
12:00:45.000	47.500	UncertainDataSubNormal, Calculated	
12:00:50.000	52.500	Good, Calculated	
12:00:55.000	57.500	Good, Calculated	
12:01:00.000	62.500	UncertainDataSubNormal, Calculated	
12:01:05.000	67.500	UncertainDataSubNormal, Calculated	
12:01:10.000	72.500	UncertainDataSubNormal, Calculated	
12:01:15.000	77.500	UncertainDataSubNormal, Calculated	
12:01:20.000	82.500	Good, Calculated	
12:01:25.000	87.500	Good, Calculated	
12:01:30.000	90	UncertainDataSubNormal, Calculated	
12:01:35.000	90	UncertainDataSubNormal, Calculated	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10.652	UncertainDataSubNormal, Calculated, Partial	
12:00:05.000	12.391	Good, Calculated	
12:00:10.000	14.565	Good, Calculated	
12:00:15.000	16.739	Good, Calculated	
12:00:20.000	18.913	Good, Calculated	
12:00:25.000	23.682	Good, Calculated	
12:00:30.000	27.046	Good, Calculated	
12:00:35.000	29.384	UncertainDataSubNormal, Calculated	
12:00:40.000	33.889	UncertainDataSubNormal, Calculated	
12:00:45.000	40	UncertainDataSubNormal, Calculated	
12:00:50.000	49.450	Good, Calculated	
12:00:55.000	52.750	Good, Calculated	
12:01:00.000	55.250	Good, Calculated	
12:01:05.000	57.750	Good, Calculated	
12:01:10.000	60.618	UncertainDataSubNormal, Calculated	
12:01:15.000	65	UncertainDataSubNormal, Calculated	
12:01:20.000	70.515	UncertainDataSubNormal, Calculated	
12:01:25.000	83.667	Good, Calculated	
12:01:30.000	~~96.250~~ 90	UncertainDataSubNormal, Calculated	
12:01:35.000	~~108.750~~ 90	UncertainDataSubNormal, Calculated	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10.652	UncertainDataSubNormal, Calculated, Partial	
12:00:05.000	~~10~~ 12.391	Good, Calculated	
12:00:10.000	~~10~~ 14.565	Good, Calculated	
12:00:15.000	~~10~~ 16.739	Good, Calculated	
12:00:20.000	~~10~~ 18.913	Good, Calculated	
12:00:25.000	~~22~~ 23.682	Good, Calculated	
12:00:30.000	~~25~~ 27.046	Good, Calculated	
12:00:35.000	~~26~~ 29.384	~~Good~~ UncertainDataSubNormal, Calculated	
12:00:40.000	~~30~~ 33.889	UncertainDataSubNormal, Calculated	
12:00:45.000	~~34~~ 40	UncertainDataSubNormal, Calculated	
12:00:50.000	~~46~~ 49.450	Good, Calculated	
12:00:55.000	~~50~~ 52.750	Good, Calculated	
12:01:00.000	~~50~~ 55.250	Good, Calculated	
12:01:05.000	~~50~~ 57.750	Good, Calculated	
12:01:10.000	~~56~~ 60.618	~~Good~~ UncertainDataSubNormal, Calculated	
12:01:15.000	~~60~~ 65	UncertainDataSubNormal, Calculated	
12:01:20.000	~~64~~ 70.515	UncertainDataSubNormal, Calculated	
12:01:25.000	~~78~~ 83.667	Good, Calculated	
12:01:30.000	90	UncertainDataSubNormal, Calculated, Partial	
12:01:35.000	90	UncertainDataSubNormal, Calculated	

A.5 TimeAverage2

A.5.1 Description

The following examples demonstrate TimeAverage2 *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:05, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.5.2 TimeAverage2 data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData	
12:00:05.000		BadNoData	
12:00:10.000	12.500	Good, Calculated	
12:00:15.000	17.500	Good, Calculated	
12:00:20.000	22.500	Good, Calculated	
12:00:25.000	27.500	Good, Calculated	
12:00:30.000	30	UncertainDataSubNormal, Calculated	
12:00:35.000	30	UncertainDataSubNormal, Calculated	
12:00:40.000		BadNoData	
12:00:45.000		BadNoData	
12:00:50.000	52.500	Good, Calculated	
12:00:55.000	57.500	Good, Calculated	
12:01:00.000	62.500	UncertainDataSubNormal, Calculated	
12:01:05.000	67.500	UncertainDataSubNormal, Calculated	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:01:10.000	72.500	UncertainDataSubNormal, Calculated	
12:01:15.000	77.500	UncertainDataSubNormal, Calculated	
12:01:20.000	82.500	Good, Calculated	
12:01:25.000	87.500	Good, Calculated	
12:01:30.000	90	UncertainDataSubNormal, Calculated, Partial	
12:01:35.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10.652	UncertainDataSubNormal, Calculated, Partial	
12:00:05.000	12.391	Good, Calculated	
12:00:10.000	14.565	Good, Calculated	
12:00:15.000	16.739	Good, Calculated	
12:00:20.000	18.913	Good, Calculated	
12:00:25.000	23.682	Good, Calculated	
12:00:30.000	27.046	Good, Calculated	
12:00:35.000	29.273	UncertainDataSubNormal, Calculated	
12:00:40.000	30	UncertainDataSubNormal, Calculated	
12:00:45.000	42.500	UncertainDataSubNormal, Calculated	
12:00:50.000	49.450	Good, Calculated	
12:00:55.000	52.750	Good, Calculated	
12:01:00.000	55.250	Good, Calculated	
12:01:05.000	57.750	Good, Calculated	
12:01:10.000	59.800	UncertainDataSubNormal, Calculated	
12:01:15.000	60	UncertainDataSubNormal, Calculated	
12:01:20.000	73.333	UncertainDataSubNormal, Calculated	
12:01:25.000	83.667	Good, Calculated	
12:01:30.000	90	UncertainDataSubNormal, Calculated, Partial	
12:01:35.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated, Partial	
12:00:05.000	10	Good, Calculated	
12:00:10.000	10	Good, Calculated	
12:00:15.000	10	Good, Calculated	
12:00:20.000	10	Good, Calculated	
12:00:25.000	22	Good, Calculated	
12:00:30.000	25	Good, Calculated	
12:00:35.000	26	Good, Calculated	
12:00:40.000		Bad, Calculated	
12:00:45.000		Bad, Calculated	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:50.000	46	Good, Calculated	
12:00:55.000	50	Good, Calculated	
12:01:00.000	50	Good, Calculated	
12:01:05.000	50	Good, Calculated	
12:01:10.000	56	Good, Calculated	
12:01:15.000		Bad, Calculated	
12:01:20.000		Bad, Calculated	
12:01:25.000	78	Good, Calculated	
12:01:30.000	90	Good, Calculated, Partial	
12:01:35.000		BadNoData	

A.6 Total

A.6.1 Description

The following examples demonstrate Total *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:05, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.6.2 Total data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData	
12:00:05.000		BadNoData	
12:00:10.000	62.500	Good, Calculated	
12:00:15.000	87.500	Good, Calculated	
12:00:20.000	112.500	Good, Calculated	
12:00:25.000	137.500	Good, Calculated	
12:00:30.000	162.500	UncertainDataSubNormal, Calculated	
12:00:35.000	187.500	UncertainDataSubNormal, Calculated	
12:00:40.000	212.500	UncertainDataSubNormal, Calculated	
12:00:45.000	237.500	UncertainDataSubNormal, Calculated	
12:00:50.000	262.500	Good, Calculated	
12:00:55.000	287.500	Good, Calculated	
12:01:00.000	312.500	UncertainDataSubNormal, Calculated	
12:01:05.000	337.500	UncertainDataSubNormal, Calculated	
12:01:10.000	362.500	UncertainDataSubNormal, Calculated	
12:01:15.000	387.500	UncertainDataSubNormal, Calculated	
12:01:20.000	412.500	Good, Calculated	
12:01:25.000	437.500	Good, Calculated	
12:01:30.000	450	UncertainDataSubNormal, Calculated	
12:01:35.000	450	UncertainDataSubNormal, Calculated	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	31.957	UncertainDataSubNormal, Calculated, Partial	
12:00:05.000	61.957	Good, Calculated	
12:00:10.000	72.826	Good, Calculated	
12:00:15.000	83.696	Good, Calculated	
12:00:20.000	94.565	Good, Calculated	
12:00:25.000	118.409	Good, Calculated	
12:00:30.000	135.227	Good, Calculated	
12:00:35.000	146.919	UncertainDataSubNormal, Calculated	
12:00:40.000	169.444	UncertainDataSubNormal, Calculated	
12:00:45.000	200	UncertainDataSubNormal, Calculated	
12:00:50.000	247.250	Good, Calculated	
12:00:55.000	263.750	Good, Calculated	
12:01:00.000	276.250	Good, Calculated	
12:01:05.000	288.750	Good, Calculated	
12:01:10.000	303.091	UncertainDataSubNormal, Calculated	
12:01:15.000	325	UncertainDataSubNormal, Calculated	
12:01:20.000	352.576	UncertainDataSubNormal, Calculated	
12:01:25.000	418.333	Good, Calculated	
12:01:30.000	481.250	UncertainDataSubNormal, Calculated	
12:01:35.000	543.750	UncertainDataSubNormal, Calculated	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	30	UncertainDataSubNormal, Calculated, Partial	
12:00:05.000	50	Good, Calculated	
12:00:10.000	50	Good, Calculated	
12:00:15.000	50	Good, Calculated	
12:00:20.000	50	Good, Calculated	
12:00:25.000	110	Good, Calculated	
12:00:30.000	125	Good, Calculated	
12:00:35.000	130	Good, Calculated	
12:00:40.000	150	UncertainDataSubNormal, Calculated	
12:00:45.000	170	UncertainDataSubNormal, Calculated	
12:00:50.000	230	Good, Calculated	
12:00:55.000	250	Good, Calculated	
12:01:00.000	250	Good, Calculated	
12:01:05.000	250	Good, Calculated	
12:01:10.000	280	Good, Calculated	
12:01:15.000	300	UncertainDataSubNormal, Calculated	
12:01:20.000	320	UncertainDataSubNormal, Calculated	
12:01:25.000	390	Good, Calculated	
12:01:30.000	450	UncertainDataSubNormal, Calculated	
12:01:35.000	450	UncertainDataSubNormal, Calculated	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.7 Total2

A.7.1 Description

The following examples demonstrate Total2 *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:05, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.7.2 Total2 data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData	
12:00:05.000		BadNoData	
12:00:10.000	62.500	Good, Calculated	
12:00:15.000	87.500	Good, Calculated	
12:00:20.000	112.500	Good, Calculated	
12:00:25.000	137.500	Good, Calculated	
12:00:30.000	150	UncertainDataSubNormal, Calculated	
12:00:35.000	150	UncertainDataSubNormal, Calculated	
12:00:40.000		BadNoData	
12:00:45.000		BadNoData	
12:00:50.000	262.500	Good, Calculated	
12:00:55.000	287.500	Good, Calculated	
12:01:00.000	312.500	UncertainDataSubNormal, Calculated	
12:01:05.000	337.500	UncertainDataSubNormal, Calculated	
12:01:10.000	362.500	UncertainDataSubNormal, Calculated	
12:01:15.000	387.500	UncertainDataSubNormal, Calculated	
12:01:20.000	412.500	Good, Calculated	
12:01:25.000	437.500	Good, Calculated	
12:01:30.000	0.090	UncertainDataSubNormal, Calculated, Partial	
12:01:35.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	31.957	UncertainDataSubNormal, Calculated, Partial	
12:00:05.000	61.957	Good, Calculated	
12:00:10.000	72.826	Good, Calculated	
12:00:15.000	83.696	Good, Calculated	
12:00:20.000	94.565	Good, Calculated	
12:00:25.000	118.409	Good, Calculated	
12:00:30.000	135.227	Good, Calculated	
12:00:35.000	146.364	UncertainDataSubNormal, Calculated	
12:00:40.000	60	UncertainDataSubNormal, Calculated	
12:00:45.000	85	UncertainDataSubNormal, Calculated	
12:00:50.000	247.250	Good, Calculated	
12:00:55.000	263.750	Good, Calculated	
12:01:00.000	276.250	Good, Calculated	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:01:05.000	288.750	Good, Calculated	
12:01:10.000	299	UncertainDataSubNormal, Calculated	
12:01:15.000	120	UncertainDataSubNormal, Calculated	
12:01:20.000	146.667	UncertainDataSubNormal, Calculated	
12:01:25.000	418.333	Good, Calculated	
12:01:30.000	0.090	UncertainDataSubNormal, Calculated, Partial	
12:01:35.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	30	Good, Calculated, Partial	
12:00:05.000	50	Good, Calculated	
12:00:10.000	50	Good, Calculated	
12:00:15.000	50	Good, Calculated	
12:00:20.000	50	Good, Calculated	
12:00:25.000	110	Good, Calculated	
12:00:30.000	125	Good, Calculated	
12:00:35.000	130	Good, Calculated	
12:00:40.000		Bad, Calculated	
12:00:45.000		Bad, Calculated	
12:00:50.000	230	Good, Calculated	
12:00:55.000	250	Good, Calculated	
12:01:00.000	250	Good, Calculated	
12:01:05.000	250	Good, Calculated	
12:01:10.000	280	Good, Calculated	
12:01:15.000		Bad, Calculated	
12:01:20.000		Bad, Calculated	
12:01:25.000	390	Good, Calculated	
12:01:30.000	0.090	Good, Calculated, Partial	
12:01:35.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.8 Minimum

A.8.1 Description

The following examples demonstrate Minimum *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.8.2 Minimum data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated, Partial	
12:00:16.000	20	Good, Calculated	
12:00:32.000		BadNoData	
12:00:48.000	50	Good, Calculated	
12:01:04.000		BadNoData	
12:01:20.000	80	Good, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated, Partial	
12:00:16.000	20	Good, Calculated	
12:00:32.000	30	UncertainDataSubNormal, Calculated	
12:00:48.000	40	Good	
12:01:04.000	60	UncertainDataSubNormal, Calculated	
12:01:20.000	70	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated, Partial	
12:00:16.000	20	Good, Calculated	
12:00:32.000	30	UncertainDataSubNormal, Calculated	
12:00:48.000	40	Good	
12:01:04.000	60	UncertainDataSubNormal, Calculated	
12:01:20.000	70	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.9 Maximum

A.9.1 Description

The following examples demonstrate Maximum *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.9.2 Maximum data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated, Partial	
12:00:16.000	30	Good, Calculated	
12:00:32.000		BadNoData	
12:00:48.000	60	Good, Calculated	
12:01:04.000		BadNoData	
12:01:20.000	90	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated, Partial	
12:00:16.000	25	Good, Calculated	
12:00:32.000	30	UncertainDataSubNormal, Calculated	
12:00:48.000	50	Good, Calculated	
12:01:04.000	60	UncertainDataSubNormal, Calculated	
12:01:20.000	90	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated, Partial	
12:00:16.000	25	Good, Calculated	
12:00:32.000	30	UncertainDataSubNormal, Calculated	
12:00:48.000	50	Good, Calculated	
12:01:04.000	60	UncertainDataSubNormal, Calculated	
12:01:20.000	90	Good, Calculated, Partial	
12:01:36.000		BadNoData	

A.10 MininumActualTime

A.10.1 Description

The following examples demonstrate the MinimumActualTime *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.10.2 MinimumActualTime data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:10.000	10	Good, Partial	
12:00:20.000	20	Good	
12:00:32.000		BadNoData	
12:00:50.000	50	Good	
12:01:04.000		BadNoData	
12:01:20.000	80	Good, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	Good, Partial	
12:00:25.000	20	Good	
12:00:39.000	30	UncertainDataSubNormal	
12:00:48.000	40	Good	
12:01:12.000	60	UncertainDataSubNormal	
12:01:23.000	70	Good, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	Good, Partial	
12:00:25.000	20	Good	
12:00:39.000	30	UncertainDataSubNormal	
12:00:48.000	40	Good	
12:01:12.000	60	UncertainDataSubNormal	
12:01:23.000	70	Good, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.11 MaximumActualTime

A.11.1 Description

The following examples demonstrate MaximumActualTime *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.11.2 MaximumActualTime data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:10.000	10	Good, Partial	
12:00:30.000	30	Good	
12:00:32.000		BadNoData	
12:01:00.000	60	Good	
12:01:04.000		BadNoData	
12:01:30.000	90	Good, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	Good, Partial	
12:00:28.000	25	Good	
12:00:39.000	30	UncertainDataSubNormal	
12:00:52.000	50	Good	
12:01:12.000	60	UncertainDataSubNormal	
12:01:30.000	90	Good, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	Good, Partial	
12:00:28.000	25	Good	
12:00:39.000	30	UncertainDataSubNormal	
12:00:52.000	50	Good	
12:01:12.000	60	UncertainDataSubNormal	
12:01:30.000	90	Good, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.12 Range

A.12.1 Description

The following examples demonstrate Range *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.12.2 Range data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:16.000	10	Good, Calculated	
12:00:32.000		BadNoData	
12:00:48.000	10	Good, Calculated	
12:01:04.000		BadNoData	
12:01:20.000	10	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:16.000	5	Good, Calculated	
12:00:32.000	0	UncertainDataSubNormal, Calculated	
12:00:48.000	10	Good, Calculated	
12:01:04.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	20	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:16.000	5	Good, Calculated	
12:00:32.000	0	UncertainDataSubNormal, Calculated	
12:00:48.000	10	Good, Calculated	
12:01:04.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	20	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.13 Minimum2

A.13.1 Description

The following examples demonstrate Minimum2 *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.13.2 Minimum2 data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	UncertainDataSubNormal, Calculated, Partial	
12:00:16.000	16	UncertainDataSubNormal, Interpolated	
12:00:32.000	30	UncertainDataSubNormal, Interpolated	
12:00:48.000	50	UncertainDataSubNormal, Calculated	
12:01:04.000	64	UncertainDataSubNormal, Interpolated	
12:01:20.000	80	UncertainDataSubNormal, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	UncertainDataSubNormal, Calculated, Partial	
12:00:16.000	16.087	Good, Interpolated	
12:00:32.000	26.818	UncertainDataSubNormal, Interpolated	
12:00:48.000	40	Good	
12:01:04.000	56	UncertainDataSubNormal, Interpolated	
12:01:20.000	70	UncertainDataSubNormal, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated, Partial	
12:00:16.000	10	Good, Interpolated	
12:00:32.000	25	Good, Interpolated	
12:00:48.000	40	Good	
12:01:04.000	50	Good, Interpolated	
12:01:20.000	70	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.14 Maximum2

A.14.1 Description

The following examples demonstrate Maximum2 *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.14.2 Maximum2 data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	16	UncertainDataSubNormal, Interpolated, Partial	
12:00:16.000	30	UncertainDataSubNormal, Calculated, MultipleValues	
12:00:32.000	30	UncertainDataSubNormal, Interpolated	
12:00:48.000	64	UncertainDataSubNormal, Interpolated	
12:01:04.000	80	UncertainDataSubNormal, Calculated	
12:01:20.000	90	UncertainDataSubNormal, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	16.087	UncertainDataSubNormal, Interpolated, Partial	
12:00:16.000	26.818	Good, Interpolated	
12:00:32.000	40	UncertainDataSubNormal, Calculated	
12:00:48.000	56	Good, Interpolated	
12:01:04.000	60	UncertainDataSubNormal, Calculated	
12:01:20.000	90	UncertainDataSubNormal, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated, Partial	
12:00:16.000	25	Good, Calculated	
12:00:32.000	30	Good, Calculated	
12:00:48.000	50	Good, Calculated	
12:01:04.000	60	Good, Calculated	
12:01:20.000	90	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.15 MinimumActualTime2

A.15.1 Description

The following examples demonstrate MinimumActualTime2 *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.15.2 MinimumActualTime2 data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:10.000	10	UncertainDataSubNormal, Partial	
12:00:16.000	16	UncertainDataSubNormal, Interpolated	
12:00:32.000	30	UncertainDataSubNormal, Interpolated	
12:00:50.000	50	UncertainDataSubNormal	
12:01:04.000	64	UncertainDataSubNormal, Interpolated	
12:01:20.000	80	UncertainDataSubNormal, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	UncertainDataSubNormal, Partial	
12:00:16.000	16.087	Good, Interpolated	
12:00:32.000	26.818	UncertainDataSubNormal, Interpolated	
12:00:48.000	40	Good	
12:01:04.000	56	UncertainDataSubNormal, Interpolated	
12:01:23.000	70	UncertainDataSubNormal, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	Good, Partial	
12:00:16.000	10	Good, Interpolated	
12:00:32.000	25	Good, Interpolated	
12:00:48.000	40	Good	
12:01:04.000	50	Good, Interpolated	
12:01:23.000	70	Good, Partial	
12:01:36.000		BadNoData	

A.16 MaximumActualTime2

A.16.1 Description

The following examples demonstrate MaximumActualTime2 *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.16.2 MaximumActualTime2 data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:15.999	16	UncertainDataSubNormal, Interpolated, Partial	
12:00:30.000	30	UncertainDataSubNormal, MultipleValues	
12:00:32.000	30	UncertainDataSubNormal, Interpolated	
12:01:03.999	64	UncertainDataSubNormal, Interpolated	
12:01:19.999	80	UncertainDataSubNormal, Interpolated	
12:01:30.000	90	UncertainDataSubNormal, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:15.999	16.087	UncertainDataSubNormal, Interpolated, Partial	
12:00:31.999	26.818	Good, Interpolated	
12:00:47.999	40	UncertainDataSubNormal, Interpolated	
12:01:03.999	56	Good, Interpolated	
12:01:12.000	60	UncertainDataSubNormal	
12:01:30.000	90	UncertainDataSubNormal, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	Good, Partial	
12:00:28.000	25	Good	
12:00:39.000	30	Good	
12:00:52.000	50	Good	
12:01:12.000	60	Good	
12:01:30.000	90	Good, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.17 Range2

A.17.1 Description

The following examples demonstrate Range2 *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.17.2 Range2 data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	6	UncertainDataSubNormal, Calculated, Partial	
12:00:16.000	14	UncertainDataSubNormal, Calculated	
12:00:32.000	0	UncertainDataSubNormal, Calculated	
12:00:48.000	14	UncertainDataSubNormal, Calculated	
12:01:04.000	16	UncertainDataSubNormal, Calculated	
12:01:20.000	10	UncertainDataSubNormal, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	6.087	UncertainDataSubNormal, Calculated, Partial	
12:00:16.000	10.731	Good, Calculated	
12:00:32.000	13.182	UncertainDataSubNormal, Calculated	
12:00:48.000	16	Good, Calculated	
12:01:04.000	4	UncertainDataSubNormal, Calculated	
12:01:20.000	20	UncertainDataSubNormal, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:16.000	15	Good, Calculated	
12:00:32.000	5	Good, Calculated	
12:00:48.000	10	Good, Calculated	
12:01:04.000	10	Good, Calculated	
12:01:20.000	20	Good, Calculated, Partial	
12:01:36.000		BadNoData	

A.18 AnnotationCount

A.18.1 Description

The following examples demonstrate AnnotationCount *Aggregate* scenarios. This *Aggregate* does not apply to current values, since annotations are features of historical data. ***ProcessingInterval***: 00:01:00, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.18.2 AnnotationCount data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	3	Good, Calculated	
12:01:00.000	1	Good, Calculated	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated	
12:01:00.000	0	Good, Calculated	

A.19 Count

A.19.1 Description

The following examples demonstrate Count *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.19.2 Count data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	1	Good, Calculated, Partial	
12:00:16.000	2	Good, Calculated	
12:00:32.000		Bad	
12:00:48.000	2	Good, Calculated	
12:01:04.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	2	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	1	Good, Calculated, Partial	
12:00:16.000	2	Good, Calculated	
12:00:32.000	1	UncertainDataSubNormal, Calculated	
12:00:48.000	2	Good, Calculated	
12:01:04.000	1	UncertainDataSubNormal, Calculated	
12:01:20.000	3	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	1	Good, Calculated, Partial	
12:00:16.000	2	Good, Calculated	
12:00:32.000		Bad	
12:00:48.000	2	Good, Calculated	
12:01:04.000	1	Good, Calculated	
12:01:20.000	3	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Historian 4			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	1	Good, Calculated, Partial	
12:00:16.000	2	Good, Calculated	
12:00:32.000	1	UncertainDataSubNormal, Calculated	
12:00:48.000	2	Good, Calculated	
12:01:04.000	1	UncertainDataSubNormal, Calculated	
12:01:20.000	3	Good, Calculated, Partial	
12:01:36.000		BadNoData	

A.20 DurationInStateZero

A.20.1 Description

The following examples demonstrate DurationInStateZero *Aggregate* scenarios. The *Aggregate* only applies to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.20.2 DurationInStateZero data

Historian 4			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	UncertainDataSubNormal, Calculated, Partial	
12:00:16.000	3000	Good, Calculated	
12:00:32.000	0	UncertainDataSubNormal, Calculated	
12:00:48.000	12000	Good, Calculated	
12:01:04.000	13000	UncertainDataSubNormal, Calculated	
12:01:20.000	4000	UncertainDataSubNormal, Calculated, Partial	
12:01:36.000		BadNoData	

A.21 DurationInStateNonZero

A.21.1 Description

The following examples demonstrate DurationInStateNonZero *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.21.2 DurationInStateNonZero data

Historian 4			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	14000	UncertainDataSubNormal, Calculated, Partial	
12:00:16.000	13000	Good, Calculated	
12:00:32.000	10000	UncertainDataSubNormal, Calculated	
12:00:48.000	4000	Good, Calculated	
12:01:04.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	3001	UncertainDataSubNormal, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.22 NumberOfTransitions

A.22.1 Description

The following examples demonstrate NumberOfTransitions *Aggregate* scenarios. ***ProcessingInterval***: 00:00:~~05~~16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.22.2 NumberOfTransitions data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	~~0~~ 1	Good, Calculated, Partial	
12:00:16.000	2	Good, Calculated	
12:00:32.000		Bad	
12:00:48.000	2	Good, Calculated	
12:01:04.000	1	UncertainDataSubNormal, Calculated	
12:01:20.000	~~1~~ 2	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	~~0~~ 1	Good, Calculated, Partial	
12:00:16.000	2	Good, Calculated	
12:00:32.000	1	UncertainDataSubNormal, Calculated	
12:00:48.000	2	Good, Calculated	
12:01:04.000	1	UncertainDataSubNormal, Calculated	
12:01:20.000	3	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	~~0~~ 1	Good, Calculated, Partial	
12:00:16.000	2	Good, Calculated	
12:00:32.000		Bad	
12:00:48.000	2	Good, Calculated	
12:01:04.000	1	Good, Calculated	
12:01:20.000	3	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 4			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	~~0~~ 1	Good, Calculated, Partial	
12:00:16.000	2	Good, Calculated	
12:00:32.000	0	UncertainDataSubNormal, Calculated	
12:00:48.000	1	Good, Calculated	
12:01:04.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	3	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.23 Start

A.23.1 Description

The following examples demonstrate Start *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.23.2 Start data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:10.000	10	Good, Partial	
12:00:20.000	20	Good	
12:00:40.000		Bad	
12:00:50.000	50	Good	
12:01:10.000	70	Uncertain	
12:01:20.000	80	Good, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	Good, Partial	
12:00:25.000	20	Good	
12:00:39.000	30	Good	
12:00:48.000	40	Good	
12:01:12.000	60	Good	
12:01:23.000	70	Good, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	Good, Partial	
12:00:25.000	20	Good	
12:00:39.000	30	Good	
12:00:48.000	40	Good	
12:01:12.000	60	Good	
12:01:23.000	70	Good, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.24 End

A.24.1 Description

The following examples demonstrate End *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.24.2 End data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:10.000	10	Good, Partial	
12:00:30.000	30	Good	
12:00:40.000		Bad	
12:01:00.000	60	Good	
12:01:10.000	70	Uncertain	
12:01:30.000	90	Good, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	Good, Partial	
12:00:28.000	25	Good	
12:00:42.000		Bad	
12:00:52.000	50	Good	
12:01:17.000	70	Uncertain	
12:01:30.000	90	Good, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	Good, Partial	
12:00:28.000	25	Good	
12:00:42.000		Bad	
12:00:52.000	50	Good	
12:01:17.000	70	Uncertain	
12:01:30.000	90	Good, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.25 StartBound

A.25.1 Description

The following examples demonstrate StartBound *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.25.2 StartBound data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData, Partial	
12:00:16.000	16	Good, Interpolated	
12:00:32.000	30	UncertainDataSubNormal, Interpolated	
12:00:48.000		BadNoData	
12:01:04.000	64	UncertainDataSubNormal, Interpolated	
12:01:20.000	80	Good, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData, Partial	
12:00:16.000	16.087	Good, Interpolated	
12:00:32.000	26.818	Good, Interpolated	
12:00:48.000	40	Good	
12:01:04.000	56	Good, Interpolated	
12:01:20.000		BadNoData, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData, Partial	
12:00:16.000	10	Good, Interpolated	
12:00:32.000	25	Good, Interpolated	
12:00:48.000	40	Good	
12:01:04.000	50	Good, Interpolated	
12:01:20.000		BadNoData, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.26 EndBound

A.26.1 Description

The following examples demonstrate End *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.26.2 EndBound data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	16	Good, Calculated, Partial	
12:00:16.000	30	UncertainDataSubNormal, Calculated	
12:00:32.000		BadNoData	
12:00:48.000	64	UncertainDataSubNormal, Calculated	
12:01:04.000	80	Good, Calculated	
12:01:20.000		BadNoData, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	16.087	Good, Calculated, Partial	
12:00:16.000	26.818	Good, Calculated	
12:00:32.000	40	Good, Calculated	
12:00:48.000	56	Good, Calculated	
12:01:04.000		BadNoData	
12:01:20.000		BadNoData, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated, Partial	
12:00:16.000	25	Good, Calculated	
12:00:32.000	40	Good, Calculated	
12:00:48.000	50	Good, Calculated	
12:01:04.000		BadNoData	
12:01:20.000		BadNoData, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.27 Delta

A.27.1 Description

The following examples demonstrate Delta *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.27.2 Delta data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:16.000	10	Good, Calculated	
12:00:32.000	0	BadNoData	
12:00:48.000	10	Good, Calculated	
12:01:04.000	0	~~UncertainDataSubNormal, Calculated~~ BadNoData	
12:01:20.000	10	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:16.000	5	Good, Calculated	
12:00:32.000	0	UncertainDataSubNormal, Calculated	
12:00:48.000	10	Good, Calculated	
12:01:04.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	20	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:16.000	5	Good, Calculated	
12:00:32.000	0	UncertainDataSubNormal, Calculated	
12:00:48.000	10	Good, Calculated	
12:01:04.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	20	Good, Calculated, Partial	
12:01:36.000		BadNoData	

A.28 DeltaBounds

A.28.1 Description

The following examples demonstrate DeltaBounds *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.28.2 DeltaBounds data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData, Partial	
12:00:16.000	14	UncertainDataSubNormal, Calculated	
12:00:32.000		BadNoData	
12:00:48.000		BadNoData	
12:01:04.000	16	UncertainDataSubNormal, Calculated	
12:01:20.000		BadNoData, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData, Partial	
12:00:16.000	10.731	Good, Calculated	
12:00:32.000	13.182	Good, Calculated	
12:00:48.000	16	Good, Calculated	
12:01:04.000		BadNoData	
12:01:20.000		BadNoData, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData, Partial	
12:00:16.000	15	Good, Calculated	
12:00:32.000	15	Good, Calculated	
12:00:48.000	10	Good, Calculated	
12:01:04.000		BadNoData	
12:01:20.000		BadNoData, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.29 DurationGood

A.29.1 Description

The following examples demonstrate DurationGood *Aggregate* scenarios. Duration values are in milliseconds. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.29.2 DurationGood data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	6000	Good, Calculated, Partial	
12:00:16.000	16000	Good, Calculated	
12:00:32.000	0	Good, Calculated	
12:00:48.000	14000	Good, Calculated	
12:01:04.000	0	Good, Calculated	
12:01:20.000	10001	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	14000	Good, Calculated, Partial	
12:00:16.000	16000	Good, Calculated	
12:00:32.000	10000	Good, Calculated	
12:00:48.000	16000	Good, Calculated	
12:01:04.000	13000	Good, Calculated	
12:01:20.000	7001	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	14000	Good, Calculated, Partial	
12:00:16.000	16000	Good, Calculated	
12:00:32.000	10000	Good, Calculated	
12:00:48.000	16000	Good, Calculated	
12:01:04.000	13000	Good, Calculated	
12:01:20.000	7001	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 4			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	14000	Good, Calculated, Partial	
12:00:16.000	16000	Good, Calculated	
12:00:32.000	10000	Good, Calculated	
12:00:48.000	16000	Good, Calculated	
12:01:04.000	13000	Good, Calculated	
12:01:20.000	7001	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.30 DurationBad

A.30.1 Description

The following examples demonstrate DurationBad *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.30.2 DurationBad data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10000	Good, Calculated, Partial	
12:00:16.000	0	Good, Calculated	
12:00:32.000	8000	Good, Calculated	
12:00:48.000	2000	Good, Calculated	
12:01:04.000	0	Good, Calculated	
12:01:20.000	0	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	2000	Good, Calculated, Partial	
12:00:16.000	0	Good, Calculated	
12:00:32.000	6000	Good, Calculated	
12:00:48.000	0	Good, Calculated	
12:01:04.000	3000	Good, Calculated	
12:01:20.000	3000	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	2000	Good, Calculated, Partial	
12:00:16.000	0	Good, Calculated	
12:00:32.000	6000	Good, Calculated	
12:00:48.000	0	Good, Calculated	
12:01:04.000	3000	Good, Calculated	
12:01:20.000	3000	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 4			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	2000	Good, Calculated, Partial	
12:00:16.000	0	Good, Calculated	
12:00:32.000	6000	Good, Calculated	
12:00:48.000	0	Good, Calculated	
12:01:04.000	3000	Good, Calculated	
12:01:20.000	3000	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.31 PercentGood

A.31.1 Description

The following examples demonstrate PercentGood *Aggregate* scenarios. **ProcessingInterval**: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.31.2 PercentGood data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	37.500	Good, Calculated, Partial	
12:00:16.000	100	Good, Calculated	
12:00:32.000	0	Good, Calculated	
12:00:48.000	87.500	Good, Calculated	
12:01:04.000	0	Good, Calculated	
12:01:20.000	100	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	87.500	Good, Calculated, Partial	
12:00:16.000	100	Good, Calculated	
12:00:32.000	62.500	Good, Calculated	
12:00:48.000	100	Good, Calculated	
12:01:04.000	81.250	Good, Calculated	
12:01:20.000	70.003	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	87.500	Good, Calculated, Partial	
12:00:16.000	100	Good, Calculated	
12:00:32.000	62.500	Good, Calculated	
12:00:48.000	100	Good, Calculated	
12:01:04.000	81.250	Good, Calculated	
12:01:20.000	70.003	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 4			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	87.500	Good, Calculated, Partial	
12:00:16.000	100	Good, Calculated	
12:00:32.000	62.500	Good, Calculated	
12:00:48.000	100	Good, Calculated	
12:01:04.000	81.250	Good, Calculated	
12:01:20.000	70.003	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.32 PercentBad

A.32.1 Description

The following examples demonstrate PercentBad *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.32.2 PercentBad data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	62.500	Good, Calculated, Partial	
12:00:16.000	0	Good, Calculated	
12:00:32.000	50	Good, Calculated	
12:00:48.000	12.500	Good, Calculated	
12:01:04.000	0	Good, Calculated	
12:01:20.000	0	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	12.500	Good, Calculated, Partial	
12:00:16.000	0	Good, Calculated	
12:00:32.000	37.500	Good, Calculated	
12:00:48.000	0	Good, Calculated	
12:01:04.000	18.750	Good, Calculated	
12:01:20.000	29.997	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	12.500	Good, Calculated, Partial	
12:00:16.000	0	Good, Calculated	
12:00:32.000	37.500	Good, Calculated	
12:00:48.000	0	Good, Calculated	
12:01:04.000	18.750	Good, Calculated	
12:01:20.000	29.997	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 4			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	12.500	Good, Calculated, Partial	
12:00:16.000	0	Good, Calculated	
12:00:32.000	37.500	Good, Calculated	
12:00:48.000	0	Good, Calculated	
12:01:04.000	18.750	Good, Calculated	
12:01:20.000	29.997	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.33 WorstQuality

A.33.1 Description

The following examples demonstrate WorstQuality *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.33.2 WorstQuality data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	Good	Good, Calculated, Partial	
12:00:16.000	Good	Good, Calculated	
12:00:32.000	Bad	Good, Calculated	
12:00:48.000	Good	Good, Calculated	
12:01:04.000	Uncertain	Good, Calculated	
12:01:20.000	Good	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	Good	Good, Calculated, Partial	
12:00:16.000	Good	Good, Calculated	
12:00:32.000	Bad	Good, Calculated	
12:00:48.000	Good	Good, Calculated	
12:01:04.000	Uncertain	Good, Calculated	
12:01:20.000	Good	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	Good	Good, Calculated, Partial	
12:00:16.000	Good	Good, Calculated	
12:00:32.000	Bad	Good, Calculated	
12:00:48.000	Good	Good, Calculated	
12:01:04.000	Uncertain	Good, Calculated	
12:01:20.000	Good	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 4			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	Good	Good, Calculated, Partial	
12:00:16.000	Good	Good, Calculated	
12:00:32.000	Bad	Good, Calculated	
12:00:48.000	Good	Good, Calculated	
12:01:04.000	Uncertain	Good, Calculated	
12:01:20.000	Good	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.34 WorstQuality2

A.34.1 Description

The following examples demonstrate WorstQuality2 *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.34.2 WorstQuality2 data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	BadNoData	Good, Calculated, Partial	
12:00:16.000	UncertainDataSubNormal	Good, Calculated	
12:00:32.000	Bad	Good, Calculated, MultipleValues	
12:00:48.000	BadNoData	Good, Calculated	
12:01:04.000	UncertainDataSubNormal	Good, Calculated, MultipleValues	
12:01:20.000	BadNoData	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	BadNoData	Good, Calculated, Partial	
12:00:16.000	Good	Good, Calculated	
12:00:32.000	Bad	Good, Calculated	
12:00:48.000	Good	Good, Calculated	
12:01:04.000	BadNoData	Good, Calculated	
12:01:20.000	BadNoData	Good, Calculated, Partial, MultipleValues	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	BadNoData	Good, Calculated, Partial	
12:00:16.000	Good	Good, Calculated	
12:00:32.000	Bad	Good, Calculated	
12:00:48.000	Good	Good, Calculated	
12:01:04.000	BadNoData	Good, Calculated	
12:01:20.000	BadNoData	Good, Calculated, Partial, MultipleValues	
12:01:36.000		BadNoData	

Historian 4			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	BadNoData	Good, Calculated, Partial	
12:00:16.000	Good	Good, Calculated	
12:00:32.000	Bad	Good, Calculated	
12:00:48.000	Good	Good, Calculated	
12:01:04.000	BadNoData	Good, Calculated	
12:01:20.000	BadNoData	Good, Calculated, Partial, MultipleValues	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.35 StandardDeviationSample

A.35.1 Description

The following examples demonstrate StandardDeviationSample *Aggregate* scenarios. ***ProcessingInterval***: 00:00:20, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.35.2 StandardDeviationSample data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	~~BadNoData~~ Good, Calculated, Partial	
12:00:20.000	~~0~~ 7.071	Good, Calculated	
12:00:40.000	0	~~BadNoData~~ UncertainDataSubNormal, Calculated	
12:01:00.000	0	~~Good~~ UncertainDataSubNormal, Calculated	
12:01:20.000	~~0~~ 7.071	Good, Calculated, Partial	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	~~BadNoData~~ Good, Calculated, Partial	
12:00:20.000	~~6.250~~ 5	Good, Calculated	
12:00:40.000	~~0~~ 7.071	UncertainDataSubNormal, Calculated	
12:01:00.000	0	~~Good~~ UncertainDataSubNormal, Calculated	
12:01:20.000	~~25~~ 10	Good, Calculated, Partial	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	~~BadNoData~~ Good, Calculated, Partial	
12:00:20.000	~~6.250~~ 5	Good, Calculated	
12:00:40.000	~~0~~ 7.071	UncertainDataSubNormal, Calculated	
12:01:00.000	0	~~Good~~ UncertainDataSubNormal, Calculated	
12:01:20.000	~~25~~ 10	Good, Calculated, Partial	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.36 VarianceSample

A.36.1 Description

The following examples demonstrate VarianceSample *Aggregate* scenarios. ***ProcessingInterval***: 00:00:20, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.36.2 VarianceSample data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	~~UncertainDataSubNormal~~ Good, Calculated, Partial	
12:00:20.000	~~16.667~~ 50	Good, Calculated	
12:00:40.000	0	UncertainDataSubNormal, Calculated	
12:01:00.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	~~16.667~~ 50	Good, Calculated, Partial	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	~~UncertainDataSubNormal~~ Good, Calculated, Partial	
12:00:20.000	~~17.720~~ 25	Good, Calculated	
12:00:40.000	~~16.667~~ 50	UncertainDataSubNormal, Calculated	
12:01:00.000	~~6~~ 0	UncertainDataSubNormal, Calculated	
12:01:20.000	~~50~~ 100	~~UncertainDataSubNormal~~ Good, Calculated, Partial	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	~~UncertainDataSubNormal~~ Good, Calculated, Partial	
12:00:20.000	~~43.750~~ 25	Good, Calculated	
12:00:40.000	50	UncertainDataSubNormal, Calculated	
12:01:00.000	~~16.667~~ 0	UncertainDataSubNormal, Calculated	
12:01:20.000	~~50~~ 100	~~UncertainDataSubNormal~~ Good, Calculated, Partial	

A.37 StandardDeviationPopulation

A.37.1 Description

The following examples demonstrate StandardDeviationPopulation *Aggregate* scenarios. ***ProcessingInterval***: 00:00:20, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.37.2 StandardDeviationPopulation data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	~~BadNoData~~ Good, Calculated, Partial	
12:00:20.000	~~0~~ 5	Good, Calculated	
12:00:40.000	0	~~BadNoData~~ UncertainDataSubNormal, Calculated	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

12:01:00.000	0	~~Good~~ UncertainDataSubNormal, Calculated	
12:01:20.000	~~0~~ 5	Good, Calculated, Partial	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	~~BadNoData~~ Good, Calculated, Partial	
12:00:20.000	~~2.500~~ 4.082	Good, Calculated	
12:00:40.000	~~0~~ 5	UncertainDataSubNormal, Calculated	
12:01:00.000	0	~~Good~~ UncertainDataSubNormal, Calculated	
12:01:20.000	~~5~~ 8.165	Good, Calculated, Partial	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	~~BadNoData~~ Good, Calculated, Partial	
12:00:20.000	~~2.500~~ 4.082	Good, Calculated	
12:00:40.000	~~0~~ 5	UncertainDataSubNormal, Calculated	
12:01:00.000	0	~~Good~~ UncertainDataSubNormal, Calculated	
12:01:20.000	~~5~~ 8.165	Good, Calculated, Partial	

A.38 VariancePopulation

A.38.1 Description

The following examples demonstrate VariancePopulation *Aggregate* scenarios. ***ProcessingInterval***: 00:00:20, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.38.2 VariancePopulation data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	~~UncertainDataSubNormal~~ Good, Calculated, Partial	
12:00:20.000	~~4.083~~ 25	Good, Calculated	
12:00:40.000	0	UncertainDataSubNormal, Calculated	
12:01:00.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	~~4.083~~ 25	Good, Calculated, Partial	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	~~UncertainDataSubNormal~~ Good, Calculated, Partial	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

12:00:20.000	~~4.210~~ 16.667	Good, Calculated	
12:00:40.000	~~4.083~~ 25	UncertainDataSubNormal, Calculated	
12:01:00.000	~~2.450~~ 0	UncertainDataSubNormal, Calculated	
12:01:20.000	~~7.071~~ 66.667	~~UncertainDataSubNormal~~ Good, Calculated, Partial	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	~~UncertainDataSubNormal~~ Good, Calculated, Partial	
12:00:20.000	~~6.614~~ 16.667	Good, Calculated	
12:00:40.000	~~7.071~~ 25	UncertainDataSubNormal, Calculated	
12:01:00.000	~~4.083~~ 0	UncertainDataSubNormal, Calculated	
12:01:20.000	~~7.071~~ 66.667	~~UncertainDataSubNormal~~ Good, Calculated, Partial	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

~~Bibliography~~

~~IEC 62541-7, *OPC Unified Architecture – Part 7: Profiles*~~

~~IEC 62541-9, *OPC Unified Architecture – Part 9: Alarms and conditions*~~

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

IEC 62541-13

Edition 2.0 2020-06

INTERNATIONAL STANDARD

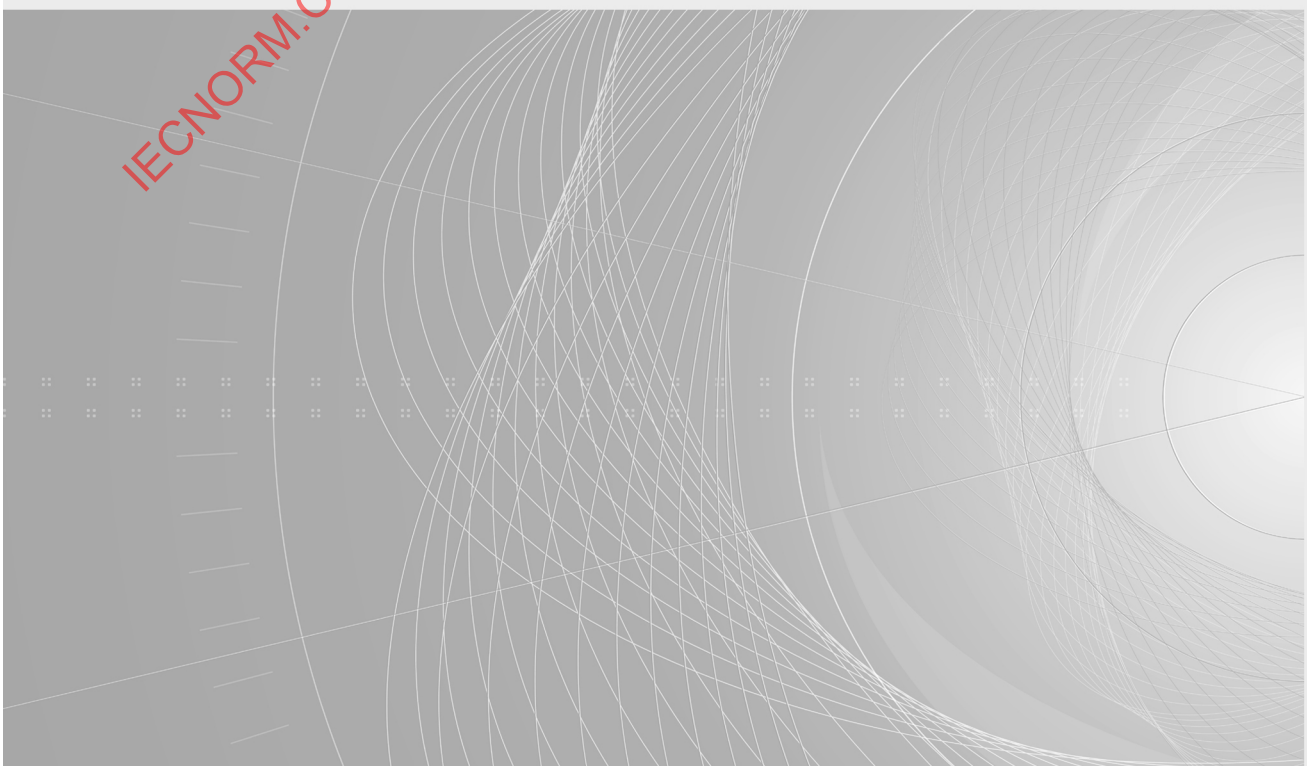
NORME INTERNATIONALE

colour inside

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

OPC Unified Architecture – Part 13: Aggregates

Architecture unifiée OPC – Partie 13: Agrégats



IEC 62541-13:2020-06(en-fr)

CONTENTS

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

INTERNATIONAL ELECTROTECHNICAL COMMISSION

_____

OPC UNIFIED ARCHITECTURE –

Part 13: Aggregates

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.

IEC 62541-13 has been prepared by subcommittee 65E: Devices and integration in enterprise systems, of IEC technical committee 65: Industrial-process measurement, control and automation.

This second edition cancels and replaces the first edition of IEC 62541-13, published in 2015. This edition constitutes a technical revision.

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

a) no technical changes but numerous clarifications. Also some corrections to the examples.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

The text of this standard is based on the following documents:

FDIS	Report on voting
65E/697/FDIS	65E/712/RVD

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

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

Throughout this document and the other Parts of the series, certain document conventions are used:

Italics are used to denote a defined term or definition that appears in the "Terms and definition" clause in one of the parts of the series.

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

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

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

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

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

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.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

OPC UNIFIED ARCHITECTURE –

Part 13: Aggregates

1 Scope

This part of IEC 62541 is part of the overall OPC Unified Architecture specification series and defines the information model associated with Aggregates.

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

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

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

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

IEC 62541-8, *OPC Unified Architecture – Part 8: Data Access*

IEC 62541-11, *OPC Unified Architecture – Part 11: Historical Access*

3 Terms, definitions, and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC TR 62541-1, IEC 62541-3, IEC 62541-4, and IEC 62541-11 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
ProcessingInterval
timespan for which derived values are produced based on a specified *Aggregate*

Note 1 to entry: The total time domain specified for ReadProcessed is divided by the *ProcessingInterval*. For example, performing a 10-minute Average over the time range 12:00 to 12:30 would result in a set of three intervals of *ProcessingInterval* length, with each interval having a start time of 12:00, 12:10 and 12:20 respectively. The rules used to determine the interval *Bounds* are discussed in 5.4.2.2.

3.1.2
interpolated data
data that is calculated from data samples

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Note 1 to entry: Data samples may be historical data or buffered real time data. An *interpolated* value is calculated from the data points on either side of the requested timestamp.

3.1.3
EffectiveEndTime
time immediately before *endTime*

Note 1 to entry: All *Aggregate* calculations include the *startTime* but exclude the *endTime*. However, it is sometimes necessary to return an *Interpolated* End Bound as the value for an *Interval* with a timestamp that is in the *interval*. *Servers* are expected to use the time immediately before *endTime* where the time resolution of the *Server* determines the exact value (do not confuse this with hardware or operating system time resolution). For example, if the *endTime* is 12:01:00, the time resolution is 1 second, then the *EffectiveEndTime* is 12:00:59. See 5.4.2.4.

If time is flowing backwards, *Servers* are expected to use the time immediately after *endTime* where the time resolution of the *Server* determines the exact value.

3.1.4
extrapolated data
data constructed from a discrete data set but is outside of the discrete data set

Note 1 to entry: It is similar to the process of interpolation, which constructs new points between known points, but its result is subject to greater uncertainty. *Extrapolated* data is used in cases where the requested time period falls farther into the future than the data available in the underlying system. See example in Table 1.

3.1.5
SlopedInterpolation
simple linear interpolation

Note 1 to entry: Compare to curve fitting using linear polynomials. See example in Table 1.

3.1.6
SteppedInterpolation
interpolation holding the last data point constant or interpolating the value based on a horizontal line fit

Note 1 to entry: Consider the following Table 1 of raw and *Interpolated*/*Extrapolated* values:

Table 1 – Interpolation examples

Timestamp	Raw Value	Sloped Interpolation	Stepped Interpolation
12:00:00	10		
12:00:05		15	10
12:00:08		18	10
12:00:10	20		
12:00:15		25	20
12:00:20	30		
		SlopedExtrapolation	SteppedExtrapolation
12:00:25		35	30
12:00:27		37	30

3.1.7
bounding values
values at the *startTime* and *endTime* needed for *Aggregates* to compute the result

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Note 1 to entry: If *Raw data* does not exist at the *startTime* and *endTime* a value shall be estimated. There are two ways to determine *Bounding Values* for an interval. One way (called *Interpolated Bounding Values*) uses the first non-Bad data points found before and after the timestamp to estimate the bound. The other (called *Simple Bounding Values*) uses the data points immediately before and after the boundary timestamps to estimate the bound even if these points are Bad. Subclauses 3.1.8 and 3.1.9 describe the two different approaches in more detail.

In all cases the *TreatUncertainAsBad* (see 4.2.1.2) flag is used to determine whether Uncertain values are Bad or non-Bad.

If a Raw value was not found and a non-Bad bounding value exists the *Aggregate* Bits (see 5.3.3) are set to 'Interpolated'.

When calculating *bounding values*, the value portion of *Raw data* that has Bad status is set to null. This means the value portion is not used in any calculation and a null is returned if the raw value is returned. The status portion is determined by the rules specified by the bound or *Aggregate*.

The *Interpolated Bounding Values* approach (see 3.1.8) is the same as what is used in Classic OPC Historical Data Access (HDA) and is important for applications such as advanced process control where having useful values at all times is important. The *Simple Bounding Values* approach (see 3.1.9) is new in this standard and is important for applications which shall produce regulatory reports and cannot use estimated values in place of Bad data.

3.1.8
interpolated bounding values
bounding values determined by a calculation using the nearest Good value

Note 1 to entry: *Interpolated Bounding Values* using *SlopedInterpolation* are calculated as follows:

- if a non-Bad Raw value exists at the timestamp then it is the bounding value;
- find the first non-Bad Raw value before the timestamp;
- find the first non-Bad Raw value after the timestamp;
- draw a line between before value and after value;
- use point where the line crosses the timestamp as an estimate of the bounding value.

The calculation can be expressed with the following formula:

$$V_{\text{bound}} = (T_{\text{bound}} - T_{\text{before}}) \times (V_{\text{after}} - V_{\text{before}})/(T_{\text{after}} - T_{\text{before}}) + V_{\text{before}}$$

where V_x is a value at 'x' and T_x is the timestamp associated with V_x.

If no non-Bad values exist before the timestamp the *StatusCode* is Bad_NoData. The *StatusCode* is *Uncertain_DataSubNormal* if any Bad values exist between the before value and after value. If either the before value or the after value are Uncertain the *StatusCode* is *Uncertain_DataSubNormal*. If the after value does not exist the before value shall be extrapolated using *SlopedExtrapolation* or *SteppedExtrapolation*.

The period of time that is searched to discover the Good values before and after the timestamp is *Server* dependent, but if a Good value is not found within some reasonable time range then the *Server* will assume it does not exist. The *Server* as a minimum should search a time range which is at least the size of the ProcessingInterval.

Interpolated Bounding Values using *SlopedExtrapolation* are calculated as follows:

- find the first non-Bad Raw value before timestamp;
- find the second non-Bad Raw value before timestamp;
- draw a line between these two values;
- extend the line to where it crosses the timestamp;
- use the point where the line crosses the timestamp as an estimate of the bounding value.

The formula is the same as the one used for *SlopedInterpolation*.

The *StatusCode* is always *Uncertain_DataSubNormal*. If only one non-Bad raw value can be found before the timestamp then *SteppedExtrapolation* is used to estimate the bounding value.

Interpolated Bounding Values using *SteppedInterpolation* are calculated as follows:

- if a non-Bad Raw value exists at the timestamp then it is the bounding value;
- find the first non-Bad Raw value before timestamp;
- use the value as an estimate of the bounding value.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

The *StatusCode* is *Uncertain_DataSubNormal* if any Bad values exist between the before value and the timestamp. If no non-Bad *Raw data* exists before the timestamp then the *StatusCode* is Bad_NoData. If the value before the timestamp is Uncertain the *StatusCode* is *Uncertain_DataSubNormal*. The value after the timestamp is not needed when using *SteppedInterpolation*; however, if the timestamp is after the end of the data then the bounding value is treated as extrapolated and the *StatusCode* is *Uncertain_DataSubNormal*.

SteppedExtrapolation is a term that describes *SteppedInterpolation* when a timestamp is after the last value in the history collection.

3.1.9
simple bounding values
bounding values determined by a calculation using the nearest value

Note 1 to entry: *Simple Bounding Values* using *SlopedInterpolation* are calculated as follows:

- if any Raw value exists at the timestamp then it is the bounding value;
- find the first Raw value before timestamp;
- find the first Raw value after timestamp;
- if the value after the timestamp is Bad then the before value is the bounding value;
- draw a line between before value and after value;
- use point where the line crosses the timestamp as an estimate of the bounding value.

The formula is the same as the one used for *SlopedInterpolation* in Clause 3.1.5.

If a Raw value at the timestamp is Bad the *StatusCode* is Bad_NoData. If the value before the timestamp is Bad the *StatusCode* is Bad_NoData. If the value before the timestamp is Uncertain the *StatusCode* is *Uncertain_DataSubNormal*. If the value after the timestamp is Bad or Uncertain the *StatusCode* is *Uncertain_DataSubNormal*.

Simple Bounding Values using *SteppedInterpolation* are calculated as follows:

- if any Raw value exists at the timestamp then it is the bounding value;
- find the first Raw value before timestamp;
- if the value before timestamp is non-Bad then it is the bounding value.

If a Raw value at the timestamp is Bad the *StatusCode* is Bad_NoData. If the value before the timestamp is Bad the *StatusCode* is Bad_NoData. If the value before the timestamp is Uncertain the *StatusCode* is *Uncertain_DataSubNormal*.

If either bounding time of an interval is beyond the last data point then the *Server* may use extrapolation or return an error. If extrapolation is used by the server the type [*SteppedExtrapolation* or *SloppedExtrapolation*] of extrapolation is server specific.

In some Historians, the last Raw value does not necessarily indicate the end of the data. Based on the Historian's knowledge of the data collection mechanism, i.e. frequency of data updates and latency, the Historian may extend the last value to a time known by the Historian to be covered. When calculating *Simple Bounding Values* the Historian will act as if there is another Raw value at this timestamp.

In the same way, if the earliest time of an interval starts before the first data point in history and the latest time is after the first data point in history, then the interval will be treated as if the interval extends from the first data point in history to the latest time of the interval and the *StatusCode* of the interval will have the Partial bit set (see 5.3.3.2).

The period of time that is searched to discover the values before and after the timestamp is *Server* dependent, but if a value is not found within some reasonable time range then the *Server* will assume it does not exist. The *Server* as a minimum should search a time range which is at least the size of the ProcessingInterval.

3.2 Abbreviated terms

DA Data Access

HA Historical Access (access to historical data or events)

HDA Historical Data Access

UA Unified Architecture

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

4 Aggregate information model

4.1 General

IEC 62541-3 and IEC 62541-5 standards define the representation of *Aggregate* historical or buffered real time data in the OPC Unified Architecture. This includes the definition of *Aggregates* used in processed data retrieval and in historical retrieval. This definition includes both standard *Reference* types and *Object* types.

4.2 Aggregate Objects

4.2.1 General

4.2.1.1 Overview

OPC UA *Servers* can support several different functionalities and capabilities. The following standard *Objects* are used to expose these capabilities in a common fashion, and there are several standard defined concepts that can be extended by vendors.

4.2.1.2 AggregateConfigurationType

The *AggregateConfigurationType* defines the general characteristics of a *Node* that defines the *Aggregate* configuration of any *Variable* or *Property*. *AggregateConfiguration Object* represents the browse entry point for information on how the *Server* treats *Aggregate* specific functionality such as handling Uncertain data. It is formally defined in Table 2.

Table 2 – AggregateConfigurationType Definition

Attribute	**Value**				
BrowseName	AggregateConfigurationType				
IsAbstract	False				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the *BaseObjectType* defined in IEC 62541-5					
HasProperty	Variable	TreatUncertainAsBad	Boolean	PropertyType	Mandatory
HasProperty	Variable	PercentDataBad	Byte	PropertyType	Mandatory
HasProperty	Variable	PercentDataGood	Byte	PropertyType	Mandatory
HasProperty	Variable	UseSlopedExtrapolation	Boolean	PropertyType	Mandatory

The *TreatUncertainAsBad Variable* indicates how the *Server* treats data returned with a *StatusCode* severity Uncertain with respect to *Aggregate* calculations. A value of True indicates the *Server* considers the severity equivalent to Bad, a value of False indicates the *Server* considers the severity equivalent to Good, unless the *Aggregate* definition says otherwise. The default value is True. Note that the value is still treated as Uncertain when the *StatusCode* for the result is calculated.

The *PercentDataBad Variable* indicates the minimum percentage of Bad data in a given interval required for the *StatusCode* for the given interval for processed data request to be set to Bad. (Uncertain is treated as defined above.) Refer to 5.4.3 for details on using this *Variable* when assigning *StatusCodes*. For details on which *Aggregates* use the *PercentDataBad Variable*, see the definition of each *Aggregate*. The default value is 100.

The *PercentDataGood Variable* indicates the minimum percentage of Good data in a given interval required for the *StatusCode* for the given interval for the processed data requests to be set to Good. Refer to 5.4.3 for details on using this *Variable* when assigning *StatusCodes*. For details on which *Aggregates* use the *PercentDataGood Variable*, see the definition of each *Aggregate*. The default value is 100.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

The *PercentDataGood* and *PercentDataBad* shall follow the following relationship *PercentDataGood* ≥ (100 – *PercentDataBad*). If they are equal the result of the *PercentDataGood* calculation is used. If the values entered for *PercentDataGood* and *PercentDataBad* do not result in a valid calculation (e.g. Bad = 80; Good = 0) the result will have a StatusCode of Bad_AggregateInvalidInputs The StatusCode Bad_AggregateInvalidInputs will be returned if the value of *PercentDataGood* or *PercentDataBad* exceed 100.

The *UseSlopedExtrapolation Variable* indicates how the *Server* interpolates data when no boundary value exists (i.e. extrapolating into the future from the last known value). A value of False indicates that the *Server* will use a *SteppedExtrapolation* format, and hold the last known value constant. A value of True indicates the *Server* will project the value using *UseSlopedExtrapolation* mode. The default value is False. For *SimpleBounds* this value is ignored.

4.2.2 AggregateFunction Object

4.2.2.1 General

This *Object* is used as the browse entry point for information about the *Aggregates* supported by a *Server*. The content of this *Object* is already defined by its type definition. All *Instances* of the *FolderType* use the standard *BrowseName* of 'AggregateFunctions'. The *HasComponent Reference* is used to relate a *ServerCapabilities Object* and/or any *HistoryServerCapabilitiesType Object* to an *AggregateFunction Object*. AggregateFunctions is formally defined in Table 3.

Table 3 – Aggregate Functions Definition

Attribute	**Value**				
BrowseName	AggregateFunctions				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Modelling Rule**
HasTypeDefinition	Object Type	*FolderType*	Defined in IEC 62541-5		

Each *ServerCapabilities* and *HistoryServerCapabilitiesType Object* shall reference an *AggregateFunction Object*. In addition, each *HistoricalConfiguration Object* belonging to a *HistoricalDataNode* may reference an *AggregateFunction Object* using the *HasComponent Reference*.

4.2.2.2 AggregateFunctionType

This *ObjectType* defines an *Aggregate* supported by a UA *Server*. This *Object* is formally defined in Table 4.

Table 4 – AggregateFunctionType Definition

Attribute	**Value**				
BrowseName	AggregateFunctionType				
IsAbstract	False				
References	**Node Class**	**BrowseName**	**DataType**	**Type Definition**	**Mod. Rule**
Subtype of the *BaseObjectType* defined in IEC 62541-5					

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

For the *AggregateFunctionType*, the *Description Attribute* (inherited from the *Base NodeClass*), is mandatory. The *Description Attribute* provides a localized description of the *Aggregate*.

Table 5 specifies the *BrowseName* and *Description Attributes* for the standard *Aggregate Objects*. The description is the localized "en" text. For other locales it shall be translated.

Table 5 – Standard AggregateType Nodes

BrowseName	Description
	Interpolation Aggregate
Interpolative	At the beginning of each interval, retrieve the calculated value from the data points on either side of the requested timestamp.
Average	Retrieve the average value of the data over the interval.
TimeAverage	Retrieve the time weighted average data over the interval using *Interpolated Bounding Values*.
TimeAverage2	Retrieve the time weighted average data over the interval using *Simple Bounding Values*.
Total	Retrieve the total (time integral) of the data over the interval using *Interpolated Bounding Values*.
Total2	Retrieve the total (time integral) of the data over the interval using *Simple Bounding Values*.
Minimum	Retrieve the minimum raw value in the interval with the timestamp of the start of the interval.
Maximum	Retrieve the maximum raw value in the interval with the timestamp of the start of the interval.
MinimumActualTime	Retrieve the minimum value in the interval and the timestamp of the minimum value.
MaximumActualTime	Retrieve the maximum value in the interval and the timestamp of the maximum value.
Range	Retrieve the difference between the minimum and maximum value over the interval.
Minimum2	Retrieve the minimum value in the interval including the *Simple Bounding Values*.
Maximum2	Retrieve the maximum value in the interval including the *Simple Bounding Values*.
MinimumActualTime2	Retrieve the minimum value with the actual timestamp including the *Simple Bounding Values*.
MaximumActualTime2	Retrieve the maximum value with the actual timestamp including the *Simple Bounding Values*.
Range2	Retrieve the difference between the Minimum2 and Maximum2 value over the interval.
Count	Retrieve the number of raw values over the interval.
DurationInStateZero	Retrieve the time a Boolean or numeric was in a zero state using *Simple Bounding Values*.
DurationInStateNonZero	Retrieve the time a Boolean or numeric was in a non-zero state using *Simple Bounding Values*.
NumberOfTransitions	Retrieve the number of changes between zero and non-zero that a Boolean or numeric value experienced in the interval.
Start	Retrieve the value at the beginning of the interval.
End	Retrieve the value at the end of the interval.
Delta	Retrieve the difference between the Start and End value in the interval.
StartBound	Retrieve the value at the beginning of the interval using *Simple Bounding Values*.
EndBound	Retrieve the value at the end of the interval using *Simple Bounding Values*.
DeltaBounds	Retrieve the difference between the StartBound and EndBound value in the interval using *Simple Bounding Values*.
DurationGood	Retrieve the total duration of time in the interval during which the data is Good.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

BrowseName	Description
	Interpolation Aggregate
DurationBad	Retrieve the total duration of time in the interval during which the data is Bad.
PercentGood	Retrieve the percentage of data (0 to 100) in the interval which has Good *StatusCode*.
PercentBad	Retrieve the percentage of data (0 to 100) in the interval which has Bad *StatusCode*.
WorstQuality	Retrieve the worst *StatusCode* of data in the interval.
WorstQuality2	Retrieve the worst *StatusCode* of data in the interval including the *Simple Bounding Values*.
AnnotationCount	Retrieve the number of *Annotation*s in the interval (applies to Historical *Aggregates* only).
StandardDeviationSample	Retrieve the standard deviation for the interval for a sample of the population (n-1).
VarianceSample	Retrieve the variance for the interval as calculated by the StandardDeviationSample.
StandardDeviation Population	Retrieve the standard deviation for the interval for a complete population (n) which includes *Simple Bounding Values*.
VariancePopulation	Retrieve the variance for the interval as calculated by the StandardDeviationPopulation which includes *Simple Bounding Values*.

4.3 MonitoredItem AggregateFilter

4.3.1 MonitoredItem AggregateFilter Defaults

The default values used for *MonitoredItem Aggregates* are the same as those used for historical *Aggregates*. They are defined in 4.2.1.2. For additional information on *MonitoredItem AggregateFilter* see IEC 62541-4.

4.3.2 MonitoredItem Aggregates and Bounding Values

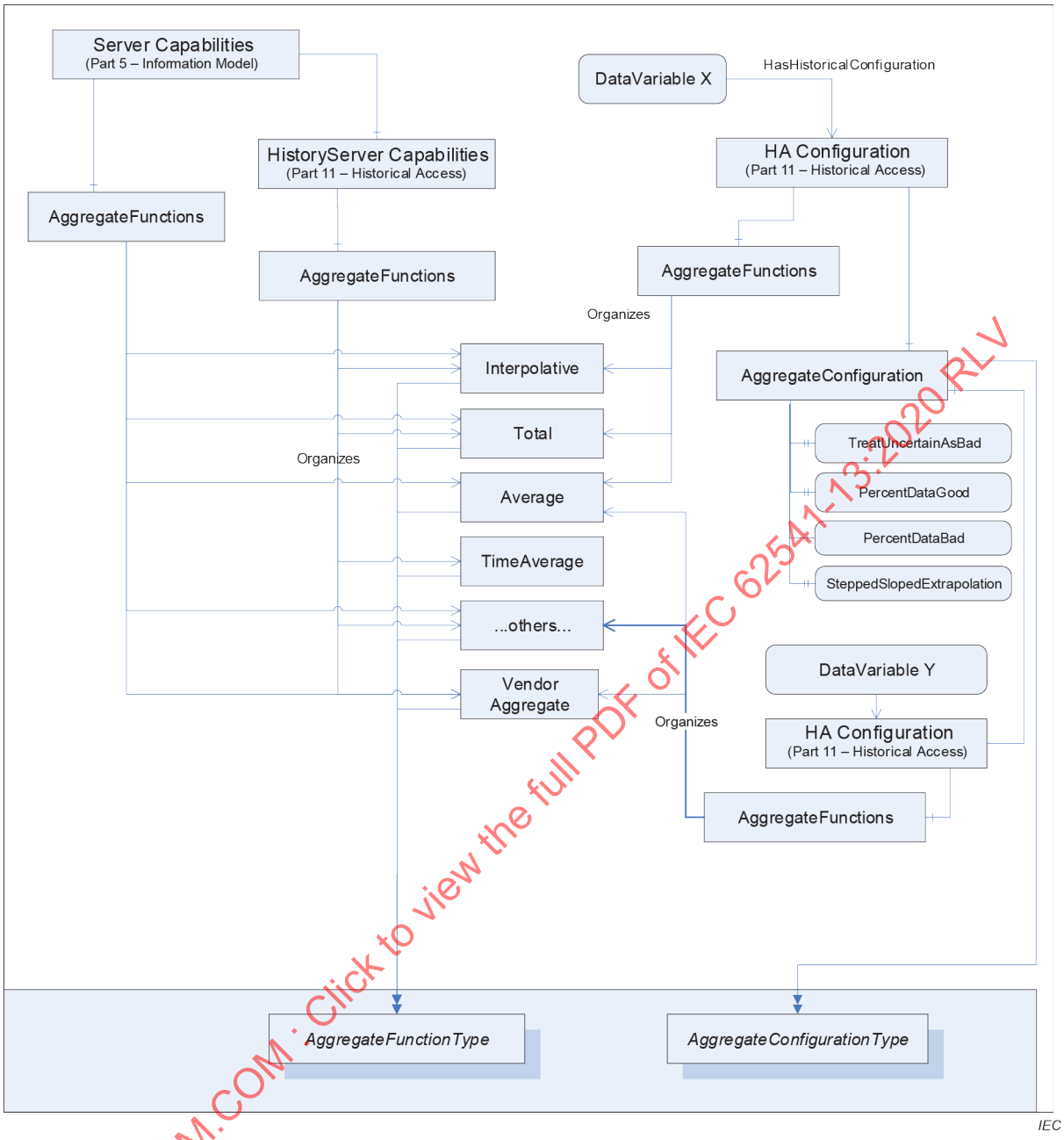
When calculating *MonitoredItem Aggregates* that require the use of *Bounding Values*, the bounds may not be known. The calculation is done in the same manner as a historical read with the Partial Bit set. The historian may wait some amount of time (normally no more than one processing interval) before calculating the interval to allow for any latency in data collection and reduce the use of the Partial Bit.

A historical read done after data collection and the data from the *MonitoredItem* over the same interval may not be the same.

4.4 Exposing Supported Functions and Capabilities

Figure 1 outlines a possible representation of *Aggregate* information in the *AddressSpace*. In this example, although the *Server* at the highest level may support *Aggregate* functionality for Interpolative, Total, Average, and others, *DataVariable* X only supports Interpolative, Total and Average, while *DataVariable* Y supports Average, a vendor defined *Aggregate* and other (unstated) *Aggregates.*

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV



IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Figure 1 – Representation of Aggregate Configuration information in the AddressSpace

5 Aggregate specific usage of Services

5.1 General

IEC 62541-4 specifies all Services needed for OPC UA *Aggregates*. In particular:

- The Browse Service Set or Query Service Set to detect *Aggregates* and their configuration.
- The HistoryRead Service of the Attribute Service Set to read the aggregated history of the HistoricalNodes.
- The CreateMonitoredItems Service allows specifying a filter for each MonitoredItem to read aggregated data.

5.2 Aggregate data handling

5.2.1 Overview

The *HistoryRead* service defined in IEC 62541-4 can perform several different functions. The *historyReadDetails* parameter is an *Extensible Parameter* that specifies which function to perform. The *ReadProcessedDetails* structure is used to read aggregated data for *HistoricalDataNodes*.

The *CreateMonitoredItems Service* allows specifying a filter for each *MonitoredItem*. The *MonitoringFilter* is an extensible parameter whose structure depends on the type of item being monitored. The *AggregateFilter* structure is used to obtain aggregated data for a subscription.

5.2.2 ReadProcessedDetails structure overview

ReadProcessedDetails structure is formally detailed in IEC 62541-11. Table 6 outlines the components of the ReadProcessedDetails structure for the purposes of discussion in this document.

Table 6 – ReadProcessedDetails

Name	Description
ReadProcessedDetails	Specifies the details used to perform a "processed" history read.
startTime	Beginning of period to read.
endTime	End of period to read.
processingInterval	Interval between returned *Aggregate* values.
aggregateType[]	The *NodeIds* of the *AggregateFunction Objects*. *AggregateFunction Objects* indicate the list of *Aggregates* to be used when retrieving processed history.
aggregateConfiguration	*Aggregate* configuration structure.
useServerDefaults	If True the *Server*'s default values are used and any values specified for the other parameters are ignored.
treatUncertainAsBad	See 4.2.1.2.
percentDataBad	See 4.2.1.2.
percentDataGood	See 4.2.1.2.
useSlopedExtrapolation	See 4.2.1.2.

5.2.3 AggregateFilter structure overview

The *AggregateFilter* defines the *Aggregate* function that should be used to calculate the values to be returned. The *AggregateFilter* is formally defined in IEC 62541-4. Table 7 outlines the components of the *AggregateFilter* structure for the purposes of discussion in this document.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 7 – AggregateFilter structure

Name	Description
AggregateFilter	
startTime	Beginning of period to calculate the *Aggregate* the first time.
aggregateType	The *NodeIds* of the *AggregateFunction Objects* that indicates the list of *Aggregates* to be used when retrieving processed data.
processingInterval	The period to be used to compute the *Aggregate*.
aggregateConfiguration	This parameter allows *Clients* to override the *Aggregate* configuration settings supplied by an *AggregateConfiguration Object* on a per monitored item basis.
useServerDefaults	If True the *Server*'s default values are used and any values specified for the other parameters are ignored.
treatUncertainAsBad	See 4.2.1.2.
percentDataBad	See 4.2.1.2.
percentDataGood	See 4.2.1.2.
useSlopedExtrapolation	See 4.2.1.2.

5.3 Aggregates StatusCodes

5.3.1 Overview

Subclause 5.3 defines additional codes and rules that apply to the *StatusCode* when used for *Aggregates*.

The general structure of the *StatusCode* is specified in IEC 62541-4. It includes a set of common operational result codes which also apply to *Aggregates.*

5.3.2 Operation level result codes

In OPC UA *Aggregates* the *StatusCode* is used to indicate the conditions under which a value or *Event* was stored, and thereby can be used as an indicator of its usability. Due to the nature of aggregated data, additional information beyond the basic quality and call result code needs to be conveyed to the client. For example, whether or not the result was *Interpolated*, were all data inputs to a calculation of Good quality, etc.

In the following, Table 8 contains codes with Bad severity indicating a failure; Table 9 contains codes with Uncertain severity indicating that the value has been retrieved under sub-normal conditions. It is important to note that these are the codes that are specific for OPC UA *Aggregates* and that they supplement the codes that apply to all types of data; they are therefore defined in IEC 62541-4, IEC 62541-8 and IEC 62541-11.

Table 8 – Bad operation level result codes

Symbolic Id	Description
Bad_AggregateListMismatch	The requested number of *Aggregates* does not match the requested number of *NodeIds*. When multiple *Aggregates* are requested, a corresponding *NodeId* is required for each *AggregateFunction*.
Bad_AggregateNotSupported	The requested *AggregateFunction* is not supported by the *Server* for the specified *Node*.
Bad_AggregateInvalidInputs	The *Aggregate* value could not be derived due to invalid data inputs, errors attempting to perform data conversions or similar situations.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 9 – Uncertain operation level result codes

Symbolic Id	Description
Uncertain_DataSubNormal	The value is derived from raw values and has less than the required number of Good values.

5.3.3 Aggregate Information Bits

5.3.3.1 General

These bits are set only when obtaining *Aggregate* data. They indicate where the data value came from and provide information that affects how the client uses the data value. Table 10 lists the bit settings which indicate the data location (i.e. is the value stored in the underlying data repository, or is the value the result of data aggregation). These bits are mutually exclusive.

Table 10 – Data location

StatusCode	Description
Raw	A *Raw data* value.
Calculated	A data value which was calculated.
Interpolated	A data value which was interpolated.

In the case where *Interpolated* data is requested, and there is an actual raw value for that timestamp, the *Server* should set the 'Raw' bit in the *StatusCode* of that value.

Table 11 lists the bit settings which indicate additional important information about the data values returned.

Table 11 – Additional information

StatusCode	Description
Partial	A calculated value that is not based on a complete interval. See 5.3.3.2.
Extra Data	If a *Server* chooses to set this bit, it indicates that a *Raw data* value supersedes other data at the same timestamp.
Multiple Values	Multiple values match the *Aggregate* criteria (i.e. multiple minimum values or multiple worst quality at different timestamps within the same *ProcessingInterval*).

The conditions under which these information bits are set depend on how the data has been requested and state of the underlying data repository.

5.3.3.2 Partial Information Bit

Partial bit is used to indicate that the interval is not a complete interval and that a client may receive a different value for the *Aggregate* if it re-fetches the interval with the same parameters.

The *Partial* bit will be set in the following examples:

Assume for these examples the first stored point in the collection is 1:01:10 and the last stored point in the collection is 1:31:20. Older data may exist but is unavailable or offline at the time of the query. Newer data may be available but has not yet been stored in the history collection.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

- The interval that overlaps the beginning of the history collection. If the start time is 1:00:00 and end time is 1:10:00 and the interval is 2 minutes then the first interval would have a partial bit set since it has no data for the first 70 seconds. The *Partial* bit will always be set for the first interval with data if the start time of the interval is before the first data value of the data collection. For intervals prior to the interval with a partial bit, these intervals will be flagged Bad_NoData.
- The interval that overlaps the latest point stored in the history collection. The last point in the collection is 1:31:20 and the historian was not shut down and is still running. A 6-minute interval that started at 1:30:00 would have the *Partial* bit set because the historian is expecting data, but just has not yet received anything. The *Partial* bit will always be set for the last interval with data if the end time of the interval is after the last data value stored in the data collection. Intervals entirely after the interval with a *Partial* bit will be flagged Bad_NoData. For those *Aggregates* with extrapolation, the *Partial* bit may be set. See the *Aggregate* specific characteristics for more details.
- If the start/end time does not result in an even interval and there is additional data beyond the end time then the last interval will have a *Partial* bit. If the start time is 1:00:00 and end time is 1:20:00 and the interval is 6 minutes then the last interval is just 2 minutes long and will have the partial bit set. Extrapolation does not apply in this case.

The *Partial* bit may be set with the Calculated bit when the Calculated bit is always set for the specific *Aggregate*.

5.4 Aggregate details

5.4.1 General

The purpose of Subclause 5.4 is to detail the requirements and behaviour for OPC UA *Servers* supporting *Aggregates*. The intent is to standardize the *Aggregates* so users can reliably predict the results of an *Aggregate* computation and understand its meaning. If users require custom functionality in the *Aggregates*, those *Aggregates* should be written as custom vendor defined *Aggregates*.

The standard *Aggregates* shall be as consistent as possible, meaning that each *Aggregate's* behaviour shall be similar to every other *Aggregate's* behaviour where input parameters, *Raw data*, and boundary conditions are similar. Where possible, the *Aggregates* should deal with input and preconditions in a similar manner.

Subclause 5.4 is divided up into two parts. Subclause 5.4.2 deals with *Aggregate* characteristics and behaviour that are common to all *Aggregates*. Subclause 5.4.3 deals with the characteristics and behaviour of *Aggregates* that are aggregate-specific.

5.4.2 Common characteristics

5.4.2.1 Description

Subclause 5.4.2 deals with *Aggregate* characteristics and behaviour that are common to all *Aggregates.*

5.4.2.2 Generating intervals

To read Historical *Aggregates*, OPC clients shall specify three time parameters:

– startTime (Start)
– endTime (End)
– *ProcessingInterval* (Int)

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

The OPC *Server* shall use these three parameters to generate a sequence of time intervals and then calculate an *Aggregate* for each interval. Subclause 5.4.2.2 specifies, given the three parameters, which time intervals are generated. Table 12 provides information on the intervals for each Start and End time combination. The range is defined to be |End – Start|.

All *Aggregates* return a timestamp of the start of the interval unless otherwise noted for the particular *Aggregate*.

Table 12 – History Aggregate interval information

Start/End Time	Interval	Resulting intervals
Start = *End*	*Int* = Anything	No intervals. Returns a Bad_InvalidArgument *StatusCode*, regardless of whether there is data at the specified time or not.
Start < *End*	*Int* = 0 or *Int* ≥ *Range*	One interval, starting at *Start* and ending at *End*. Includes *Start*, excludes *End*, i.e., *[Start, End)*.
Start < *End*	*Int* ≠ 0, *Int* < *Range*, *Int* divides Range evenly.	*Range/Int* intervals. Intervals are *[Start, Start + Int), [Start + Int, Start + 2 × Int),..., [End – Int, End)*.
Start < *End*	*Int* ≠ 0, *Int* < *Range*, *Int* does not divide *Range* evenly.	⌈*Range/Int*⌉ intervals. Intervals are *[Start, Start + Int), [Start + Int, Start + 2 × Int),..., [Start + (⌊Range/Int⌋– 1) × Int, Start + ⌊Range/Int⌋ × Int), [Start + ⌊Range/Int⌋ × Int, End)*. In other words, the last interval contains the "rest" that remains in the range after taking away ⌊*Range/Int*⌋ intervals of size *Int*.
Start > *End*	*Int* = 0 or *Int* ≥ *Range*	One interval, starting at *Start* and ending at *End*. Includes *Start*, excludes *End*, i.e.,[*Start*, *End*). [a]
Start > *End*	*Int* ≠ 0, *Int* < *Range*, *Int* divides *Range* evenly.	*Range/Int* intervals. Intervals are [*Start, Start- Int), [Start– Int, Start – 2 × Int),..., [End + Int, End)*. [a]
Start > *End*	*Int* ≠ 0, *Int* < *Range*, *Int* does not divide *Range* evenly.	⌈*Range/Int*⌉ intervals. Intervals are [*Start, Start – Int), [Start –Int,Start – 2 × Int),..., [Start – (⌊Range/Int⌋– 1) × Int , Start – ⌊Range/Int⌋ × Int), [Start – ⌊Range/Int⌋ × Int , End)*. In other words, the last interval contains the "rest" that remains in the range after taking away ⌊*Range/Int*⌋ intervals of size *Int* starting at Start. [a]
[a] In this case time is running backwards on the intervals.		

The calculation of all *Aggregates* when time flows backwards is the same as when time flows forwards with the exception that the 'early time' is excluded from the interval and the 'late time' is included. In most cases this means the value will be the same except the timestamps are shifted by one *ProcessingInterval*. E.g. when time flows forward the value at $T = n$ is the same as the value at $T = n + 1$ when time flows backward.

Note that when determining *Aggregates* with *MonitoredItem*, the interval is simply the *ProcessingInterval* parameter as defined in the *AggregateFilter* structure. See IEC 62541-4 for more details.

5.4.2.3 Data types

Table 13 outlines the valid *DataType* for each *Aggregate*. Some *Aggregates* are intended for numeric data types – i.e. integers or real/floating point numbers. Dates, strings, arrays, etc. are not supported. Other *Aggregates* are intended for digital data types – i.e. Boolean or enumerations. In addition some *Aggregates* may return results with a different DataType than those used to calculate the *Aggregate*. Table 13 also outlines the data type returned for each *Aggregate* .

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 13 – Standard History Aggregate Data Type information

BrowseName	Valid Data Type	Result Data Type
	Interpolation Aggregate	
Interpolative	Numeric	Raw Data Type
	Data Averaging Aggregates	
Average	Numeric	Double
TimeAverage	Numeric	Double
TimeAverage2	Numeric	Double
Total	Numeric	Double
Total2	Numeric	Double
	Data Variation Aggregates	
Minimum	Numeric	Raw data type
Maximum	Numeric	Raw data type
MinimumActualTime	Numeric	Raw data type
MaximumActualTime	Numeric	Raw data type
Range	Numeric	Raw data type
Minimum2	Numeric	Raw data type
Maximum2	Numeric	Raw data type
MinimumActualTime2	Numeric	Raw data type
MaximumActualTime2	Numeric	Raw data type
Range2	Numeric	Raw data type
	Counting Aggregates	
AnnotationCount	All	Integer
Count	All	Integer
DurationInStateZero	Numeric or Boolean	Duration
DurationInStateNonZero	Numeric or Boolean	Duration
NumberOfTransitions	Numeric or Boolean	Integer
	Time Aggregates	
Start	All	Raw data type
End	All	Raw data type
Delta	Numeric	Raw data type
StartBound	All	Raw data type
EndBound	All	Raw data type
DeltaBounds	Numeric	Raw data type
	Data Quality Aggregates	
DurationGood	All	Duration
DurationBad	All	Duration
PercentGood	All	Double
PercentBad	All	Double
WorstQuality	All	*StatusCode*
WorstQuality2	All	*StatusCode*
	Statistical Aggregates	
StandardDeviationSample	Numeric	Double
VarianceSample	Numeric	Double
StandardDeviationPopulation	Numeric	Double
VariancePopulation	Numeric	Double

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.2.4 Time calculation issues

The following issues may come up when calculating *Aggregates* that include time as part of the calculation.

- All *Aggregate* calculations include the *startTime* but exclude the *endTime*. However, it is sometimes necessary to return an *Interpolated* End Bound as the value for an *Interval* with a timestamp that is in the *Interval*. *Servers* are expected to use the time immediately before *endTime* where the time resolution of the *Server* determines the exact value (do not confuse this with hardware or operating system time resolution). For example, if the *endTime* is 12:01:00, the time resolution is 1 second, then the *EffectiveEndTime* is 12:00:59. If the *Server* time resolution is 1 millisecond the *EffectiveEndTime* is 12:00:59.999.

  If time is flowing backwards, *Servers* are expected to use the time immediately after *endTime* where the time resolution of the *Server* determines the exact value.
- If there is one data point in the *Interval* and it falls on the *StartTime* the time duration used in calculations is one unit of the time resolution of the *Server*.

5.4.3 Specific aggregated data handling

5.4.3.1 General

When accessing aggregated data using the *HistoryRead* or the *CreateMonitoredItems Service*, the following rules are used to handle specific *Aggregate* use cases.

If *ProcessingInterval* is 0, the *Server* shall create one *Aggregate* value for the entire time range. This allows *Aggregates* over large periods of time. A value with a timestamp equal to *endTime* will be excluded from that *Aggregate*, just as it would be excluded from an interval with that ending time. If the *ProcessingInterval* of 0 is passed in the *MonitoredItemFilter* it shall be revised to a suitable non-zero value.

The timestamp returned with the *Aggregate* shall be the time at the beginning of the interval, except where the *Aggregate* specifies a different timestamp.

If a requested timestamp is set to anything but the source timestamp the operation shall return the *Bad_TimestampToReturnInvalid StatusCode*. If a requested timestamp is not supported in any other way for a *HistoricalDataNode*, the operation shall return the *Bad_TimestampNotSupported StatusCode*. For *MonitoredItem* the *Server* shall not return past data if a requested timestamp is not supported by the history collection.

5.4.3.2 StatusCode calculation

5.4.3.2.1 General

StatusCodes for an *Aggregate* value shall take into account the values used to calculate them. In addition, the configuration parameters *PercentDataGood* and *PercentDataBad* allow the client to control how this calculation is done if supported by the *Server*.

If an *Aggregate* operates on raw values (e.g. Average) the calculation is done by counting values. If an *Aggregate* operates on raw values but can also return a *Bounding Value* then the *Bounding Values* are included in the count when computing the *StatusCode*. If an *Aggregate* does any sort of a time weighted calculation (e.g. TimeAverage or TimeAverage2) then the *StatusCode* calculation shall also be time weighted.

For purposes of calculating time weighted *StatusCodes* each interval shall be divided into regions of Good or Bad data. Creating these regions requires that the *bounding values* be calculated for each interval and the type of bounding value depends on the *Aggregate*.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

If *TreatUncertainAsBad* = False then Uncertain regions are included with the Good regions when calculating the above ratios, if the *TreatUncertainAsBad* = True then the Uncertain regions are included as Bad regions. The *StatusCode* of the value is still treated as Uncertain when the *StatusCode* for the result is calculated. If no Bad regions are in the interval then the *StatusCode* for the interval is Good. For any intervals containing regions where the *StatusCodes* are Bad, the total duration of all *Bad* regions is calculated and divided by the width of the interval. The resulting ratio is multiplied by 100 and compared to the *PercentDataBad* parameter. The *StatusCode* for the interval is Bad if the ratio is greater than or equal to the *PercentDataBad* parameter. For any interval which is not Bad, the total duration of all Good regions is then calculated and divided by the width of the interval. The resulting ratio is multiplied by 100 and compared to the *PercentDataGood* parameter. The *StatusCode* for the interval is Good if the ratio is greater than or equal to the *PercentDataGood* parameter. If for an interval neither ratio applies then that interval is *Uncertain_DataSubNormal*.

If there is no data in the interval and the interval is inside the range [start of data, end of data] and the *Aggregate* return data type is raw data type then the *StatusCodes* for the interval will be Bad_NoData unless an alternate status code is defined for a specific *Aggregate*.

The width of an interval is the *ProcessingInterval* unless it is a partial interval (i.e. has the Partial bit set). In these cases, the width is the time used when calculating the partial interval.

Subclauses 5.4.3.2.2 and 5.4.3.2.3 include diagrams that illustrate a request and data series. The colour of the time axis indicates the status for different regions. Red indicates Bad, green indicates Good and orange indicates Uncertain. These examples assume *TreatUncertainAsBad* = False.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.2.2 Sloped Interpolation and Simple Bounding Values

Figure 2 illustrates a data series for *Variable* with Stepped = False and an *Aggregate* that uses *Simple Bounding Values*. The request being processed has a Start Time that falls before the first point in the series and an End Time that does not fall on an integer multiple of the *ProcessingInterval*.

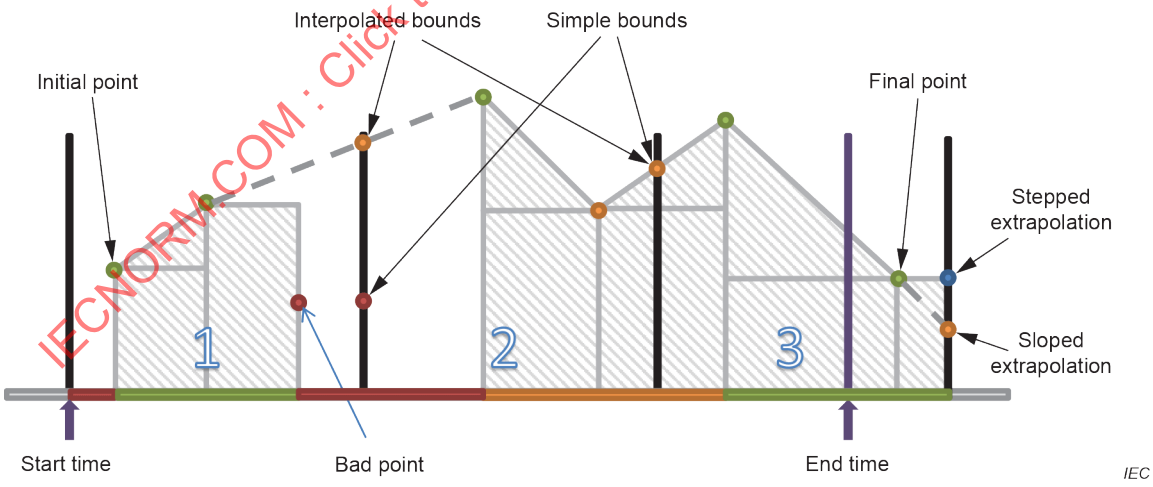


Figure 2 – Variable with Stepped = False and Simple Bounding Values

The first interval has four regions:

- the period before the first data point;
- the period between the first and second where *SlopedInterpolation* can be used;
- the period between the second and third point where *SteppedInterpolation* is used;
- the period after the Bad point where no data exists.

A region is Uncertain if a region ends in a Bad or Uncertain value and *SlopedInterpolation* is used. The end point has no effect on the region if *SteppedInterpolation* is used.

The second interval has three regions:

- the period before the first Good data point where no data exists;
- the period between the first and second where *SlopedInterpolation* can be used;
- the period between the second point and the bound calculated with *SlopedInterpolation*.

The third interval has three regions:

- the period between the simple bound and the first data point;
- the period between the first point and an interpolated point that falls on the end time;
- the period after the end time which is ignored.

This is a partial region and the data after the end time is not used, however, if sloped interpolation is used and the point after the endpoint is Uncertain then the region between the last point and the end time will be Uncertain.

5.4.3.2.3 Stepped Interpolation and Interpolated Bounding Values

Figure 3 illustrates a data series for *Variable* with Stepped = True and an *Aggregate* that uses *Interpolated Bounding Values*. The request being processed has a Start Time that falls before the first point in the series and an End Time that does not fall on an integer multiple of the *ProcessingInterval*.

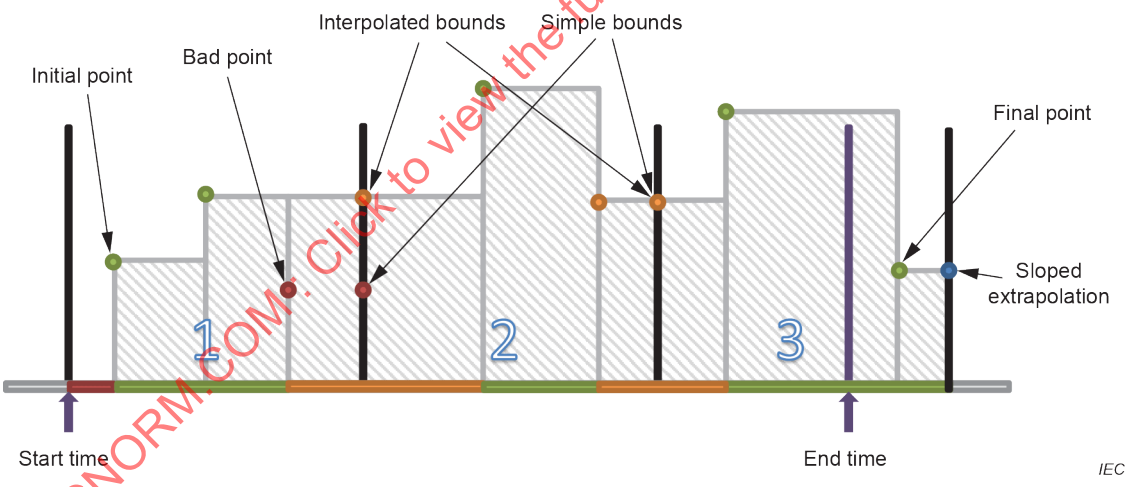


Figure 3 – Variable with Stepped = True and Interpolated Bounding Values

The first interval has three regions:

- the period before the first data point;
- the period between the first and second where *SteppedInterpolation* is used;
- the period between the second and the interpolated end bound.

The Bad point is ignored because of the interpolated end bound but this does create Uncertain regions. If *SlopedInterpolation* was used the Uncertain region would start at the second point. In this case, it only starts when the first Bad value is ignored.

The second interval has three regions:

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

- the period between the start bound and the first data point;
- the period between the first and second where *SteppedInterpolation* is used;
- the period between the second and the interpolated end bound.

The third interval has three regions:

- the period between the interpolated bound and the first data point;
- the period between the first point and an interpolated point that falls on the end time;
- the period after the end time which is ignored.

This is a partial region and the data after the end time is not used.

5.4.3.3 Description

Subclause 5.4.3.3 deals with *Aggregate* specific characteristics and behaviour that is specific to a particular *Aggregate*.

Each subclause has a table which formally expresses the *Aggregate* behaviour (including any exceptions). The meaning of each of the fields in the table is described in Table 14.

Description of Table 14:

- The first column is the common name for the item.
- The second column includes a description of the item and a list of the valid selections with for the item including a description of each selection.
- The second part of the table describes how the status associated with the *Aggregate* calculation is computed.
- The last part of the table lists what behaviour is expected from the *Aggregate* for some common special cases. These behaviours require text descriptions so there is no list of valid selections.

Table 14 – Aggregate table description

Aggregate Characteristics	
Type	The type of *Aggregate*. <Interpolated | Calculated | Raw> Interpolated: See definition for Interpolated. Calculated: Computed from defined calculation. Raw: Selects a raw value from within an interval.
Data Type	The data type of the result. <Double | Int32 | Same as Source>
Use Bounds	How the *Aggregate* deals with bounds. <None | Interpolated | Simple> None: Bounds do not apply to the *Aggregate*. Interpolated: Uses Interpolated Bounds. Simple: Uses Simple Bounds.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Aggregate Characteristics	
Timestamp	What is the time stamp of the resulting *Aggregate* value: <*startTime* | *endTime* | Raw> *startTime*: The time at the start of the interval. *endTime*: The time at the end of the interval. Raw: The time associated with a value in the interval.
StatusCode Calculations	
Calculation Method	How the status code is calculated: <PercentValues | PercentTime | Custom> PercentValues: Based on percentage of value counts. PercentTime: Based on percentage of time interval. Custom: Specific to the *Aggregate* (description included).
Partial	For partial intervals does the *Aggregate* set this bit <Set Sometimes | Not Set> It may also describe any special cases for setting this bit
Calculated	Describes the usage of the calculated bit. <Set Always | Set Sometimes | Not Set> Set Always: The bit is always set. Set Sometimes: The bit is sometimes set (describes when). Not Set: The bit is never set.
Interpolated	Describes the usage of the interpolated bit. <Set Always | Set Sometimes | Not Set> Set Always: The bit is always set. Set Sometimes: The bit is sometimes set (describes when). Not Set: The bit is never set.
Raw	Describes the usage of the Raw bit. <Set Always | Set Sometimes | Not Set> Set Always: The bit is always set. Set Sometimes: The bit is sometimes set (describes when). Not Set: The bit is never set.
Multi Value	Describes the usage of the multi value bit. <Set Sometimes | Not Set> Set Sometimes: The bit is used (see IEC 62541-11). Not Set: The bit is never set.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Aggregate Characteristics	
StatusCode Common Special Cases	
Before Start of Data	If the entire interval is before the start of data.
After End of Data	If the entire interval is after the end of data (as determined by the Historian).
Start Bound Not Found	If the starting bound is not found for the earliest interval and it is not partial, then what, if any, special processing should be done.
End Bound Not Found	If the ending bound is not found for the latest interval and it is not partial, then what, if any, special processing should be done.
Bound Bad	If the Bounding value is Bad, then what, if any, special processing should be done.
Bound Uncertain	If the Bounding value is uncertain, then what, if any, special processing should be done.

5.4.3.4 Interpolative

The Interpolative *Aggregate* defined in Table 15 returns the *Interpolated Bounding Value* (see 3.1.8) for the *startTime* of each interval.

When searching for Good values before or after the bounding value, the time period searched is *Server* specific, but the *Server* should search a time range which is at least the size of the *ProcessingInterval*.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 15 – Interpolative Aggregate summary

Interpolated Aggregate Characteristics	
Type	Interpolated
Data Type	Same as Source
Use Bounds	Interpolated
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Good if no Bad values skipped and Good values are used, Uncertain if Bad values skipped or if Uncertain values are used. If no starting value then *Bad_NoData*. See description of Interpolated Bounds (see 3.1.8) for more details
Partial bit	Not Set
Calculated bit	Not Set
Interpolated bit	Set Sometimes Always set except for when the Raw bit is set
Raw bit	Set Sometimes If a value exists with the exact time of interval Start
Multi Value bit	Not Set
StatusCode Common Special Cases	
Before Start of Data	Return *Bad_NoData*
After End of Data	Return extrapolated value (see 3.1.8) (sloped or stepped according to settings) Status code is *Uncertain_DataSubNormal*.
Start Bound Not Found	*Bad_NoData*.
End Bound Not Found	*See "After End of Data"*
Bound Bad	Does not return a Bad bound except as noted above
Bound Uncertain	Returned *Uncertain_DataSubNormal* if any Bad value(s) was/were skipped to calculate the bounding value.

5.4.3.5 Average

The Average *Aggregate* defined in Table 16 adds up the values of all Good *Raw data* for each interval, and divides the sum by the number of Good values. If any non-Good values are ignored in the computation, the *Aggregate StatusCode* will be determined using the *StatusCode* Calculation (see 5.3). This *Aggregate* is not time based so the PercentGood/PercentBad applies to the number of values in the interval.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 16 – Average Aggregate summary

Average Aggregate Characteristics	
Type	Calculated
Data Type	Double
Use Bounds	None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	PercentValues
Partial	Not Set
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Bounds not used
No End Bound	Bounds not used
Bound Bad	Bounds not used
Bound Uncertain	Bounds not used

5.4.3.6 TimeAverage

The TimeAverage *Aggregate* defined in Table 17 uses *Interpolated Bounding Values* (see 3.1.8) to find the value of a point at the beginning and end of an interval. Starting at the starting bounding value a straight line is drawn between each value in the *interval* ending at the ending bounding value (see examples for illustrations). The area under the lines is divided by the length of the *ProcessingInterval* to yield the average. Note that this calculation always uses a sloped line between points; TimeAverage2 uses a stepped or sloped line depending on the value of the Stepped *Property* for the *Variable*.

If one or more Bad Values exist in the interval then they are omitted from the calculation and the *StatusCode* is set to *Uncertain_DataSubNormal*. Sloped lines are drawn between the Good values when calculating the area.

The time resolution used in this calculation is *Server* specific.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 17 – TimeAverage Aggregate summary

TimeAverage Aggregate Characteristics	
Type	Calculated
Data Type	Double
Use Bounds	Interpolated
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Good if no Bad values skipped and Good values are used, Uncertain if Bad values are skipped or if Uncertain values are used
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Value extrapolated, Uncertain status
No Start Bound	Calculate Partial Interval
No End Bound	Extrapolate data, Uncertain status
Bound Bad	NA
Bound Uncertain	NA

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.7 TimeAverage2

The TimeAverage2 *Aggregate* defined in Table 18 uses *Simple Bounding Values* (see 3.1.9) to find the value of a point at the beginning and end of an interval. Starting at the starting bounding value a straight line is drawn between each value in the interval ending at the ending bounding value (see examples for illustrations). The area under the lines is divided by the length of the *ProcessingInterval* to yield the average. Note that this calculation uses a stepped or sloped line depending on what the value of the Stepped *Property* for the *Variable*; TimeAverage always uses a sloped line between points.

The time resolution used in this calculation is *Server* specific.

If any non-Good data exists in the interval, this data is omitted from the calculation and the time interval is reduced by the duration of the non-Good data; i.e. if a value was Bad for 1 minute in a 5-minute interval then the TimeAverage2 would be the area under the 4-minute period of Good values divided by 4 minutes. If a sub-interval ends at a Bad value then only the Good starting value is used to calculate the area of sub-interval preceding the Bad value.

The *Aggregate StatusCode* will be determined using the *StatusCode* Calculation (see 5.3).

Table 18 – TimeAverage2 Aggregate summary

TimeAverage2 Aggregate Characteristics	
Type	Calculated
Data Type	Double
Use Bounds	Simple
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	PercentTime
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Bound is Bad_NoData and treated as any other Bad value in the interval
No End Bound	Bound is Bad_NoData and treated as any other Bad value in the interval
Bound Bad	Treated as any other Bad value in the interval
Bound Uncertain	Treated as any other Uncertain value in the interval

5.4.3.8 Total

The Total *Aggregate* defined in Table 19 performs the following calculation for each interval:

Total = TimeAverage × *ProcessingInterval* (seconds)

where: TimeAverage is the result from the TimeAverage *Aggregate*, using the *ProcessingInterval* supplied to the Total call.

The resulting units would be normalized to seconds, i.e. [TimeAverage Units] × seconds.

The *Aggregate StatusCode* will be determined using the *StatusCode* Calculation (see 5.3).

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 19 – Total Aggregate summary

Total Aggregate Characteristics	
Type	Calculated
Data Type	Double
Use Bounds	Interpolated
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Good if no Bad values are skipped and Good values are used, Uncertain if Bad values are skipped or if Uncertain values are used
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Value extrapolated, Uncertain status
No Start Bound	Calculate Partial Interval
No End Bound	Extrapolate data, Uncertain status
Bound Bad	NA
Bound Uncertain	NA

5.4.3.9 Total2

The Total2 *Aggregate* defined in Table 20 performs the following calculation for each interval:

Total2 = TimeAverage2 × *ProcessingInterval* of Good data (seconds)

where TimeAverage2 is the result from the TimeAverage2 *Aggregate*, using the *ProcessingInterval* supplied to the Total2 call.

The interval of Good data is the sum of all sub-intervals where non-Bad data exists; i.e. if a value was Bad for 1 minute in a 5-minute interval then the interval of Good data would be the 4-minute period.

The resulting units would be normalized to seconds, i.e. [TimeAverage2 Units] × seconds.

The *Aggregate StatusCode* will be determined using the *StatusCode* Calculation (see 5.3).

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 20 – Total2 Aggregate summary

Total2 Aggregate Characteristics	
Type	Calculated
Data Type	Double
Use Bounds	Simple
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	PercentTime
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Value for Bound is Bad_NoData and is treated like any other Bad quality value in the calculation (ignored)
No End Bound	Value for Bound is Bad_NoData and is treated like any other Bad quality value in the calculation (ignored)
Bound Bad	Value is treated like any other Bad quality value in the calculation (ignored)
Bound Uncertain	Value is treated like any other non-Good quality value in the calculation (ignored)

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.10 Minimum

The Minimum *Aggregate* defined in Table 21 retrieves the minimum Good raw value within the interval, and returns that value with the timestamp at the start of the interval. Note that if the same minimum exists at more than one timestamp the MultipleValues bit is set.

Unless otherwise indicated, *StatusCodes* are Good, Calculated. If the minimum value is on the start time the status code will be Good, Raw. If only Bad quality values are available then the status is returned as *Bad_NoData*.

The timestamp of the *Aggregate* will always be the start of the interval for every *ProcessingInterval*.

Table 21 – Minimum Aggregate summary

Minimum Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom If no Bad values then the Status is Good. If Bad values exist then the Status is *Uncertain_SubNormal.* If an Uncertain value is less than the minimum Good value the Status is *Uncertain_SubNormal*.
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Sometimes If the Minimum value is not on the StartTime of the interval or if the Status was set to *Uncertain_SubNormal* because of non-Good values in the interval
Interpolated	Not Set
Raw	Set Sometimes If Minimum value is on the StartTime of the interval
Multi Value	Set Sometimes If multiple Good values exist with the Minimum value
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Not Applicable
No End Bound	Not Applicable
Bound Bad	Not Applicable
Bound Uncertain	Not Applicable

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.11 Maximum

The Maximum *Aggregate* defined in Table 22 retrieves the maximum Good raw value within the interval, and returns that value with the timestamp at the start of the interval. Note that if the same maximum exists at more than one timestamp the MultipleValues bit is set.

Unless otherwise indicated, *StatusCodes* are Good, Calculated. If the minimum value is on the interval start time the status code will be Good, Raw. If only Bad quality values are available then the status is returned as *Bad_NoData*.

The timestamp of the *Aggregate* will always be the start of the interval for every *ProcessingInterval*.

Table 22 – Maximum Aggregate summary

Maximum Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom If no Bad values then the Status is Good. If Bad values exist then the Status is *Uncertain_SubNormal.* If an Uncertain value is greater than the maximum Good value the Status is *Uncertain_SubNormal*
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Sometimes If the Maximum value is not on the *startTime* of the interval or if the Status was set to *Uncertain_SubNormal* because of non-Good values in the interval
Interpolated	Not Set
Raw	Set Sometimes If Maximum value is on the *startTime* of the interval
Multi Value	Set Sometimes If multiple Good values exist with the Maximum value
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Not Applicable
No End Bound	Not Applicable
Bound Bad	Not Applicable
Bound Uncertain	Not Applicable

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.12 MinimumActualTime

The MinimumActualTime *Aggregate* defined in Table 23 retrieves the minimum Good raw value within the interval, and returns that value with the timestamp at which that value occurs. Note that if the same minimum exists at more than one timestamp, the oldest one is retrieved and the *Aggregate Bits are* set to MultipleValues.

Table 23 – MinimumActualTime Aggregate summary

MinimumActualTime Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	None
Timestamp	Time of Minimum
StatusCode Calculations	
Calculation Method	Custom If no Bad values then the Status is Good. If Bad values exist then the Status is *Uncertain_SubNormal.* If an Uncertain value is less than the minimum Good value the Status is *Uncertain_SubNormal*
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Sometimes If the Status was set to *Uncertain_SubNormal* because of non-Good values in the interval
Interpolated	Not Set
Raw	Set Sometimes If a Good minimum value is returned
Multi Value	Set Sometimes If multiple Good values exist with the Minimum value
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Not Applicable
No End Bound	Not Applicable
Bound Bad	Not Applicable
Bound Uncertain	Not Applicable

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.13 MaximumActualTime

The MaximumActualTime *Aggregate* defined in Table 24 is the same as the MinimumActualTime *Aggregate*, except that the value is the maximum raw value within the interval. Note that if the same maximum exists at more than one timestamp, the oldest one is retrieved and the *Aggregate Bits are* set to MultipleValues.

Table 24 – MaximumActualTime Aggregate summary

MaximumActualTime Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	None
Timestamp	Time of Maximum
StatusCode Calculations	
Calculation Method	Custom If no Bad values then the Status is Good. If Bad values exist then the Status is *Uncertain_SubNormal.* If an Uncertain value is greater than the maximum Good value the Status is *Uncertain_SubNormal*
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Sometimes If the Status was set to *Uncertain_SubNormal* because of non-Good values in the interval
Interpolated	Not Set
Raw	Set Sometimes If a Good maximum value is returned
Multi Value	Set Sometimes If multiple Good values exist with the maximum value
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Not Applicable
No End Bound	Not Applicable
Bound Bad	Not Applicable
Bound Uncertain	Not Applicable

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.14 Range

The Range *Aggregate* defined in Table 25 finds the difference between the maximum and minimum Good raw values in the interval. If only one *Good* value exists in the interval, the range is zero. Note that the range is always zero or positive. If non-Good values are ignored when finding the minimum or maximum values or if Bad values exist then the status is *Uncertain_DataSubNormal.*

Table 25 – Range Aggregate summary

Range Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom If no Bad values then the Status is Good. If Bad values exist then the Status is *Uncertain_SubNormal.* If an Uncertain value is greater than the maximum or less than the minimum Good value the Status is *Uncertain_SubNormal*
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Not Applicable
No End Bound	Not Applicable
Bound Bad	Not Applicable
Bound Uncertain	Not Applicable

5.4.3.15 Minimum2

The Minimum2 *Aggregate* defined in Table 26 retrieves the minimum Good value for each interval as defined for Minimum except that *Simple Bounding Values* are included. The *Simple Bounding Values* for the interval are found according to the definition of *Simple Bounding Values* (see 3.1.9). Any Bad values are ignored in the computation. The *Aggregate StatusCode* will be determined using the *StatusCode* Calculation (see 5.3) for time based *Aggregates*. If a bounding value is returned then the status will indicate, Raw, Calculated or Interpolated.

If *TreatUncertainAsBad* is false and an Uncertain raw value is the minimum then that Uncertain value is used. Uncertain values are ignored otherwise.

If sloped interpolation is used and the End bound is the minimum value then End bound is used as the Minimum with the timestamp set to the *startTime* of the interval. The End bound is ignored in all other cases.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 26 – Minimum2 Aggregate summary

Minimum2 Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	Simple
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	PercentTime
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Sometimes Set unless the StartBound is the Minimum
Interpolated	Set Sometimes If an Interpolated bound is the Minimum
Raw	Set Sometimes If a raw value is the Minimum.
Multi Value	Set Sometimes If more than one Good values exist with the same
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Treat the beginning value as Bad_NoData and compute the *Aggregate*
No End Bound	Treat the ending value as Bad_NoData and compute the *Aggregate*
Bound Bad	Use as value and compute the *Aggregate* as defined
Bound Uncertain	Use as value and compute the *Aggregate* as defined

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.16 Maximum2

The Maximum2 *Aggregate* defined in Table 27 retrieves the maximum Good value for each interval as defined for Maximum except that *Simple Bounding Values* are included. The *Simple Bounding Values* for the interval are found according to the definition of *Simple Bounding Values* (see 3.1.9). Any Bad values are ignored in the computation. The *Aggregate StatusCode* will be determined using the *StatusCode* Calculation (see 5.3) for time based *Aggregates*. If a bounding value is returned then the status will indicate, Raw, Calculated or Interpolated.

If *TreatUncertainAsBad* is false and an Uncertain raw value is the maximum then that Uncertain value is used. Uncertain values are ignored otherwise.

If sloped interpolation is used and the End bound is the maximum value then End bound is used as the maximum with the timestamp set to the *startTime* of the interval. The End bound is ignored in all other cases.

Table 27 – Maximum2 Aggregate summary

Maximum2 Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	Simple
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	PercentTime
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Sometimes Set unless the StartBound is the Maximum
Interpolated	Set Sometimes If an Interpolated bound is the Maximum
Raw	Set Sometimes If a raw value is the Maximum.
Multi Value	Set Sometimes If more than one Good values exist with the same
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Treat the beginning value as Bad_NoData and compute the *Aggregate*
No End Bound	Treat the ending value as Bad_NoData and compute the *Aggregate*
Bound Bad	Use as value and compute the *Aggregate* as defined
Bound Uncertain	Use as value and compute the *Aggregate* as defined

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.17 MinimumActualTime2

The MinimumActualTime2 *Aggregate* defined in Table 28 retrieves the minimum Good value for each interval as defined for MinimumActualTime except that *Simple Bounding Values* are included. The *Simple Bounding Values* for the interval are found according to the definition of *Simple Bounding Values* (see 3.1.9). Any Bad values are ignored in the computation. The *Aggregate StatusCode* will be determined using the *StatusCode* Calculation (see 5.3) for time based *Aggregates*. If a bounding value is returned then the status will indicate, Raw, Calculated or Interpolated.

If *TreatUncertainAsBad* is false and an Uncertain raw value is the minimum then that Uncertain value is used. Uncertain values are ignored otherwise.

If sloped interpolation is used and the End bound is the minimum value then End bound is used as the minimum with the timestamp set to the *EffectiveEndTime* of the interval. The End bound is ignored in all other cases.

Table 28 – MinimumActualTime2 Aggregate summary

MinumumActualTime2 Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	Simple
Timestamp	Time of minimum
StatusCode Calculations	
Calculation Method	PercentTime
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Not Set
Interpolated	Set Sometimes If an Interpolated bound is the Minimum
Raw	Set Sometimes If a raw value is the Minimum
Multi Value	Set Sometimes If more than one Good values exist with the same value
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Treat the beginning value as Bad_NoData and compute the *Aggregate*
No End Bound	Treat the ending value as Bad_NoData and compute the *Aggregate*
Bound Bad	Use as value and compute the *Aggregate* as defined
Bound Uncertain	Use as value and compute the *Aggregate* as defined

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.18 MaximumActualTime2

The MaximumActualTime2 *Aggregate* defined in Table 29 retrieves the maximum Good value for each interval as defined for MaximumActualTime except that *Simple Bounding Values* are included. The *Simple Bounding Values* for the interval are found according to the definition of *Simple Bounding Values* (see 3.1.9). Any Bad values are ignored in the computation. The *Aggregate StatusCode* will be determined using the *StatusCode* Calculation (see 5.3) for time based *Aggregates*. If a bounding value is returned then the status will indicate, Raw, Calculated or Interpolated.

If *TreatUncertainAsBad* is false and an Uncertain raw value is the maximum then that Uncertain value is used. Uncertain values are ignored otherwise.

If sloped interpolation is used and the End bound is the maximum value then End bound is used as the maximum with the timestamp set to the EffectiveEndTime of the interval. The End bound is ignored in all other cases.

Table 29 – MaximumActualTime2 Aggregate summary

MaximumActualTime2 Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	Simple
Timestamp	Time of maximum
StatusCode Calculations	
Calculation Method	PercentTime
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Not Set
Interpolated	Set Sometimes If an Interpolated bound is the Maximum
Raw	Set Sometimes If a raw value is the Maximum
Multi Value	Set Sometimes If more than one value is equal to the Maximum
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Treat the beginning value as Bad_NoData and compute the *Aggregate*
No End Bound	Treat the ending value as Bad_NoData and compute the *Aggregate*
Bound Bad	Use as value and compute the *Aggregate* as defined
Bound Uncertain	Use as value and compute the *Aggregate* as defined

5.4.3.19 Range2

The Range2 *Aggregate* defined in Table 30 finds the difference between the maximum and minimum values in the interval as returned by the Minimum2 and Maximum2 *Aggregates*. Note that the range is always zero or positive.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 30 – Range2 Aggregate summary

Range2 Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	Simple (used in Minimum2 and Maximum2 calculations)
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom If Minimum2 or Maximum2 are Bad then the status is Bad_NoData. If Minimum2 or Maximum2 are Uncertain then the status is *Uncertain_DataSubNormal*. Good otherwise
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Handled by Minimum2 and Maximum2
No End Bound	Handled by Minimum2 and Maximum2
Bound Bad	Handled by Minimum2 and Maximum2
Bound Uncertain	Handled by Minimum2 and Maximum2

5.4.3.20 AnnotationCount

The AnnotationCount *Aggregate* defined in Table 31 returns a count of all *Annotations* in the interval.

The *StatusCodes* are Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 31 – AnnotationCount Aggregate summary

AnnotationCount Aggregate Characteristics	
Type	Calculated
Data Type	Int32 (negative values are not allowed)
Use Bounds	None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Good unless the interval is before the start of data or after the end of data
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Does not apply
No End Bound	Does not apply
Bound Bad	Does not apply
Bound Uncertain	Does not apply

5.4.3.21 Count

The Count *Aggregate* defined in Table 32 retrieves a count of all the raw values within an interval. If one or more raw values are non-Good, they are not included in the count, and the *Aggregate StatusCode* is determined using the *StatusCode* Calculation (see 5.4.3) for non-time based *Aggregates*. If no Good data exists for an interval, the count is zero.

Unless otherwise indicated, *StatusCodes* are Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 32 – Count Aggregate summary

Count Aggregate Characteristics	
Type	Calculated
Data Type	Int32 (negative values are not allowed)
Use Bounds	None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	PercentValues
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Does not apply
No End Bound	Does not apply
Bound Bad	Does not apply
Bound Uncertain	Does not apply

5.4.3.22 DurationInStateZero

The DurationInStateZero *Aggregate* defined in Table 33 returns the time *Duration* during the interval that the *Variable* was in the zero state. The *Simple Bounding Values* for the interval are used to determine initial value (start time < end time) or ending value (if start time > end time). If one or more raw values are non-Good, they are not included in the *Duration*, and the *Aggregate StatusCode* is determined using the *StatusCode* Calculation (see 5.3) for time based *Aggregates*. *Duration* is in milliseconds. Unless otherwise indicated, *StatusCodes* are Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 33 – DurationInStateZero Aggregate summary

DurationInStateZero Aggregate Characteristics	
Type	Calculated
Data Type	Duration
Use Bounds	Simple
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	PercentTime
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Treat the beginning value as Bad_NoData and compute the *Aggregate*
No End Bound	Treat the ending value as Bad_NoData and compute the *Aggregate*
Bound Bad	Use as value and compute the *Aggregate* as defined
Bound Uncertain	Use as value and compute the *Aggregate* as defined

5.4.3.23 DurationInStateNonZero

The DurationInStateNonZero *Aggregate* defined in Table 34 returns the time *Duration* during the interval that the *Variable* was in the one state. The *Simple Bounding Values* for the interval are used to determine initial value (start time < end time) or ending value (if start time > end time). If one or more raw values are non-Good, they are not included in the *Duration*, and the *Aggregate StatusCode* is determined using the *StatusCode* Calculation (see 5.3) for time based *Aggregates*.

Duration is in milliseconds. Unless otherwise indicated, *StatusCodes* are Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 34 – DurationInStateNonZero Aggregate Summary

DurationInStateNonZero Aggregate Characteristics	
Type	Calculated
Data Type	Duration
Use Bounds	Simple
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	PercentTime
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Treat the beginning value as Bad_NoData and compute the *Aggregate*
No End Bound	Treat the ending value as Bad_NoData and compute the *Aggregate*
Bound Bad	Use as value and compute the *Aggregate* as defined
Bound Uncertain	Use as value and compute the *Aggregate* as defined

5.4.3.24 NumberOfTransitions

The NumberOfTransitions *Aggregate* defined in Table 35 returns a count of the number of transition the *Variable* had during the interval. If one or more raw values are Bad, they are not included in the count, and the *Aggregate StatusCode* is determined using the *StatusCode* Calculation (see 5.3) for non-time based *Aggregates*.

The earliest transition shall be calculated by comparing the earliest non-Bad value in the interval to the previous non-Bad value. A transition occurred if no previous non-Bad value exists or if the earliest non-Bad value is different. The *endTime* is not considered part of the interval, so a transition occurring at the *endTime* is not included.

Unless otherwise indicated, *StatusCodes* are Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 35 – NumberOfTransitions Aggregate summary

NumberOfTransitions Aggregate Characteristics	
Type	Calculated
Data Type	Int32 (negative values are not allowed)
Use Bounds	Custom, a non-Bad value prior to the interval is used
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	PercentValues
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Treat the beginning value as Bad_NoData and compute the *Aggregate*
No End Bound	Treat the ending value as Bad_NoData and compute the *Aggregate*
Bound Bad	Use as value and compute the *Aggregate* as defined
Bound Uncertain	Use as value and compute the *Aggregate* as defined

5.4.3.25 Start

The Start *Aggregate* defined in Table 36 retrieves the earliest raw value within the interval, and returns that value and status with the timestamp at which that value occurs. If no values are in the interval then the *StatusCode* is *Bad_NoData*.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 36 – Start Aggregate summary

Start Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	None
Timestamp	Time of Raw Value
StatusCode Calculations	
Calculation Method	Custom The raw value status is returned
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Not Set
Interpolated	Not Set
Raw	Always
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Does not apply
No End Bound	Does not apply
Bound Bad	Does not apply
Bound Uncertain	Does not apply

5.4.3.26 End

The End *Aggregate* defined in Table 37 retrieves the latest raw value within the interval, and returns that value and status with the timestamp at which that value occurs. If no values are in the interval then the *StatusCode* is *Bad_NoData*.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 37 – End Aggregate summary

End Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	None
Timestamp	Time of Raw Value
StatusCode Calculations	
Calculation Method	Custom The raw value status is returned
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Not Set
Interpolated	Not Set
Raw	Always
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Does not apply
No End Bound	Does not apply
Bound Bad	Does not apply
Bound Uncertain	Does not apply

5.4.3.27 Delta

The Delta *Aggregate* defined in Table 38 retrieves the difference between the earliest and latest Good raw values in the interval. The *Aggregate* is negative if the latest value is less than the earliest value. The status is *Uncertain_DataSubNormal* if non-Good values are skipped while looking for the first or last values. The status is *Good* otherwise. The status is *Bad_NoData* if no Good raw values exist.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 38 – Delta Aggregate summary

Delta Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom *Uncertain_DataSubNormal* if non-Good values are skipped while looking for the first or last values
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Always
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Does not apply
No End Bound	Does not apply
Bound Bad	Does not apply
Bound Uncertain	Does not apply

5.4.3.28 StartBound

The StartBound *Aggregate* defined in Table 39 returns the value and status at the *StartTime* for the interval by calculating the *Simple Bounding Values* for the interval (see 3.1.9).

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 39 – StartBound Aggregate summary

StartBound Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	Simple
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom The status of the start bound.
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Not Set
Interpolated	Set Sometimes If the bound is interpolated
Raw	Set Sometimes If a value exists at the start time
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Bad_NoData
No End Bound	Does not apply
Bound Bad	Same as bound
Bound Uncertain	Same as bound

5.4.3.29 EndBound

The EndBound *Aggregate* defined in Table 40 returns the value and status at the *EndTime* for the interval by calculating the *Simple Bounding Values* for the interval (see 3.1.9).

The timestamp returned is always the start of the interval and Calculated bit is set.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 40 – EndBound Aggregate summary

EndBound Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	Simple
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom The status of the end bound.
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Does not apply
No End Bound	Bad_NoData
Bound Bad	Same as bound
Bound Uncertain	Same as bound

5.4.3.30 DeltaBounds

The DeltaBounds *Aggregate* defined in Table 41 returns the difference between the *StartBound* and the *EndBound Aggregates* with the exception that both the start and end shall be Good. If the end value is less than the start value, the result will be negative. If the end value is the same as the start value the result will be zero. If the end value is greater than the start value, the result will be positive. If one or both values are Bad the return status will be *Bad_NoData*. If one or both values are Uncertain the status will be *Uncertain_DataSubNormal*.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 41 – DeltaBounds Aggregate summary

DeltaBounds Aggregate Characteristics	
Type	Calculated
Data Type	Same as Source
Use Bounds	Simple
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Good if both bounds are Good *Uncertain_DataSubNormal* if either bound is uncertain Bad_NoData if either bound is Bad
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	Bad_NoData
No End Bound	Bad_NoData
Bound Bad	Bad_NoData
Bound Uncertain	Uncertain_DataSubNormal

5.4.3.31 DurationGood

The DurationGood *Aggregate* defined in Table 42 divides the interval into regions of Good and non-Good data. Each region starts with a data point in the interval. If that data point is Good the region is Good. The *Aggregate* is the sum of the duration of all Good regions expressed in milliseconds.

The status of the first region is determined by finding the first data point at or before the start of the interval. If no value exists, the first region is Bad.

Each *Aggregate* is returned with timestamp of the start of the interval. *StatusCodes* are Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 42 – DurationGood Aggregate summary

DurationGood Aggregate Characteristics	
Type	Calculated
Data Type	Duration
Use Bounds	Uses status of bounding value
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom *StatusCode* is always *Good*, *Calculated*
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	No special handing required
No End Bound	No special handing required
Bound Bad	No special handing required
Bound Uncertain	No special handing required

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.32 DurationBad

The DurationBad *Aggregate* defined in Table 43 divides the interval into regions of Bad and non-Bad data. Each region starts with a data point in the interval. If that data point is Bad the region is Bad. The *Aggregate* is the sum of the duration of all Bad regions expressed in milliseconds.

The status of the first region is determined by finding the first data point at or before the start of the interval. If no value exists, the first region is Bad.

Each *Aggregate* is returned with timestamp of the start of the interval. *StatusCodes* are Good, Calculated.

Table 43 – DurationBad Aggregate summary

DurationBad Aggregate Characteristics	
Type	Calculated
Data Type	Duration
Use Bounds	Uses status of bounding value
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom *StatusCode* is always *Good, Calculated*
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	No special handing required
No End Bound	No special handing required
Bound Bad	No special handing required
Bound Uncertain	No special handing required

5.4.3.33 PercentGood

The PercentGood *Aggregate* defined in Table 44 performs the following calculation:

PercentGood = DurationGood / *ProcessingInterval* × 100

where:

DurationGood is the result from the DurationGood *Aggregate*, calculated using the *ProcessingInterval* supplied to *PercentGood* call.

ProcessingInterval is the duration of interval.

If the last interval is a partial interval then the duration of the partial interval is used in the calculation. Each *Aggregate* is returned with timestamp of the start of the interval. *StatusCodes* are Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 44 – PercentGood Aggregate summary

PercentGood Aggregate Characteristics	
Type	Calculated
Data Type	Double (percent)
Use Bounds	Simple (used in DurationGood calculation)
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Always Good
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	No special handing required
No End Bound	No special handing required
Bound Bad	No special handing required
Bound Uncertain	No special handing required

5.4.3.34 PercentBad

The PercentBad *Aggregate* defined in Table 45 performs the following calculation:

PercentBad = DurationBad / *ProcessingInterval* × 100

where:

DurationBad is the result from the DurationBad *Aggregate*, calculated using the *ProcessingInterval* supplied to *PercentBad* call.

ProcessingInterval is the duration of interval.

If the last interval is a partial interval then the duration of the partial interval is used in the calculation. Each *Aggregate* is returned with timestamp of the start of the interval. *StatusCodes* are Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 45 – PercentBad Aggregate summary

PercentBad Aggregate Characteristics	
Type	Calculated
Data Type	Double (percent)
Use Bounds	Simple (used in DurationBad calculation)
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Always Good.
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	No special handing required
No End Bound	No special handing required
Bound Bad	No special handing required
Bound Uncertain	No special handing required

5.4.3.35 WorstQuality

The WorstQuality *Aggregate* defined in Table 46 returns the worst status of the raw values in the interval where a Bad status is worse than Uncertain, which is worse than Good. No distinction is made between the specific reasons for the status.

If multiple values exist with the worst quality but different *StatusCodes* then the *StatusCode* of the first value is returned and the *MultipleValues* bit is set.

This *Aggregate* returns the worst *StatusCode* as the value of the *Aggregate*.

The timestamp is always the start of the interval. The *StatusCodes* are Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 46 – WorstQuality Aggregate summary

WorstQuality Aggregate Characteristics	
Type	Calculated
Data Type	StatusCode
Use Bounds	None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Always Good
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Used
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	No special handing required
No End Bound	No special handing required
Bound Bad	No special handing required
Bound Uncertain	No special handing required

5.4.3.36 WorstQuality2

The *WorstQuality2 Aggregate* defined in Table 47 returns the worst status of the raw values in the interval where a Bad status is worse than Uncertain, which is worse than Good. No distinction is made between the specific reasons for the status.

The start bound calculated using *Simple Bounding Values* (see 3.1.9) is always included when determining the worst quality.

If multiple values exist with the worst quality but different *StatusCodes* then the *StatusCode* of the first value is returned and the *MultipleValues* bit is set.

This *Aggregate* returns the worst *StatusCode* as the value of the *Aggregate*.

The timestamp is always the start of the interval. The *StatusCodes* are Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 47 – WorstQuality2 Aggregate summary

WorstQuality2 Aggregate Characteristics	
Type	Calculated
Data Type	StatusCode
Use Bounds	Simple
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Always Good
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Used
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	No special handing required
No End Bound	No special handing required
Bound Bad	No special handing required
Bound Uncertain	No special handing required

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.37 StandardDeviationSample

The *StandardDeviationSample Aggregate* defined in Table 48 uses the formula:

$$\sqrt{\frac{\sum_{1}^{n}(X - \text{Avg}(X))^2}{n-1}}$$

where X is each Good raw value in the interval, Avg(X) is the average of the Good raw values, and n is the number of Good raw values in the interval.

For every interval where n = 1, a value of 0 is returned.

If any non-Good values were ignored, the *Aggregate* quality is uncertain/subnormal.

All interval *Aggregates* return timestamp of the start of the interval. Unless otherwise indicated, qualities are *Good, Calculated*.

This calculation is for a sample population where the calculation is done on a subset of the full set of data. Use *StandardDeviationPopulation* to calculate the standard deviation of a full set of data (see 5.4.3.39). An example would be when the underlying data is sampled from the data source versus stored on an exception basis.

Table 48 – StandardDeviationSample Aggregate summary

StandardDeviationSample Aggregate Characteristics	
Type	Calculated
Data Type	StatusCode
Use Bounds	None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Always Good
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	No special handing required
No End Bound	No special handing required
Bound Bad	No special handing required
Bound Uncertain	No special handing required

5.4.3.38 VarianceSample

The *VarianceSample Aggregate* defined in Table 49 retrieves the square of the standard deviation. Its behaviour is the same as the *StandardDeviationSample Aggregate*. Unless otherwise indicated, qualities are *Good, Calculated*.

This calculation is for a sample population where the calculation is done on a subset of the full population. Use *VariancePopulation* to calculate the variance of a full set of data (5.4.3.40).

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 49 – VarianceSample Aggregate summary

VarianceSample Aggregate Characteristics	
Type	Calculated
Data Type	StatusCode
Use Bounds	None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Always Good
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	No special handing required
No End Bound	No special handing required
Bound Bad	No special handing required
Bound Uncertain	No special handing required

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.39 StandardDeviationPopulation

The *StandardDeviation Population Aggregate* defined in Table 50 uses the formula:

$$\sqrt{\frac{\sum_{1}^{n}(X - \text{Avg}(X))^2}{n}}$$

where X is each Good raw value in the interval, Avg(X) is the average of the Good raw values, and n is the number of Good raw values in the interval.

For every interval where n = 1, a value of 0 is returned.

If any non-Good values were ignored, the *Aggregate* quality is uncertain/subnormal.

All interval *Aggregates* return timestamp of the start of the interval. Unless otherwise indicated, qualities are *Good, Calculated*.

This calculation is for a full population where the calculation is done on the full set of data. Use *StandardDeviationSample* to calculate the standard deviation of a subset of the full population (5.4.3.37). An example would be when the underlying data is collected on an exception basis versus sampled from the data source.

Table 50 – StandardDeviationPopulation Aggregate summary

StandardDeviationPopulation Aggregate Characteristics	
Type	Calculated
Data Type	StatusCode
Use Bounds	None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Always Good
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	No special handing required
No End Bound	No special handing required
Bound Bad	No special handing required
Bound Uncertain	No special handing required

5.4.3.40 VariancePopulation

The *VariancePopulation Aggregate* defined in Table 51 retrieves the square of the standard deviation. Its behaviour is the same as the *StandardDeviationPopulation Aggregate*. Unless otherwise indicated, qualities are *Good, Calculated*.

This calculation is for a full population where the calculation is done on the full set of data. Use *VarianceSample* to calculate the variance of a subset of the full population (5.4.3.38).

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Table 51 – VariancePopulation Aggregate summary

VariancePopulation Aggregate Characteristics	
Type	Calculated
Data Type	StatusCode
Use Bounds	None
Timestamp	StartTime
StatusCode Calculations	
Calculation Method	Custom Always Good
Partial	Set Sometimes If an interval is not a complete interval
Calculated	Set Always
Interpolated	Not Set
Raw	Not Set
Multi Value	Not Set
StatusCode Common Special Cases	
Before Start of Data	Bad_NoData
After End of Data	Bad_NoData
No Start Bound	No special handing required
No End Bound	No special handing required
Bound Bad	No special handing required
Bound Uncertain	No special handing required

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Annex A
(informative)

Aggregate Specific examples – Historical Access

A.1 Historical Aggregate specific characteristics

A.1.1 Example Aggregate data – Historian 1

For the purposes of Historian 1 examples consider a source historian with the data given in Figure A.1:

Timestamp	Value	StatusCode	Notes
12:00:00	-	Bad_NoData	First archive entry, Point created
12:00:10	10	Raw, Good	
12:00:20	20	Raw, Good	
12:00:30	30	Raw, Good	
12:00:40	40	Raw, Bad	**ANNOTATION**: Operator 1 Jan-02-2012 8:00:00 Scan failed, Bad data entered **ANNOTATION:** Jan-04-2012 7:10:00 Value cannot be verified
12:00:50	50	Raw, Good	**ANNOTATION**: Engineer1 Jan-04-2012 7:00:00 Scanner fixed
12:01:00	60	Raw, Good	
12:01:10	70	Raw, Uncertain	**ANNOTATION: Technician_1** **Jan-02-2012 8:00:00 Value flagged as questionable**
12:01:20	80	Raw, Good	
12:01:30	90	Raw, Good	
	null	No Data	No more entries, awaiting next scan

The example historian also has *Annotations* associated with three data points.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

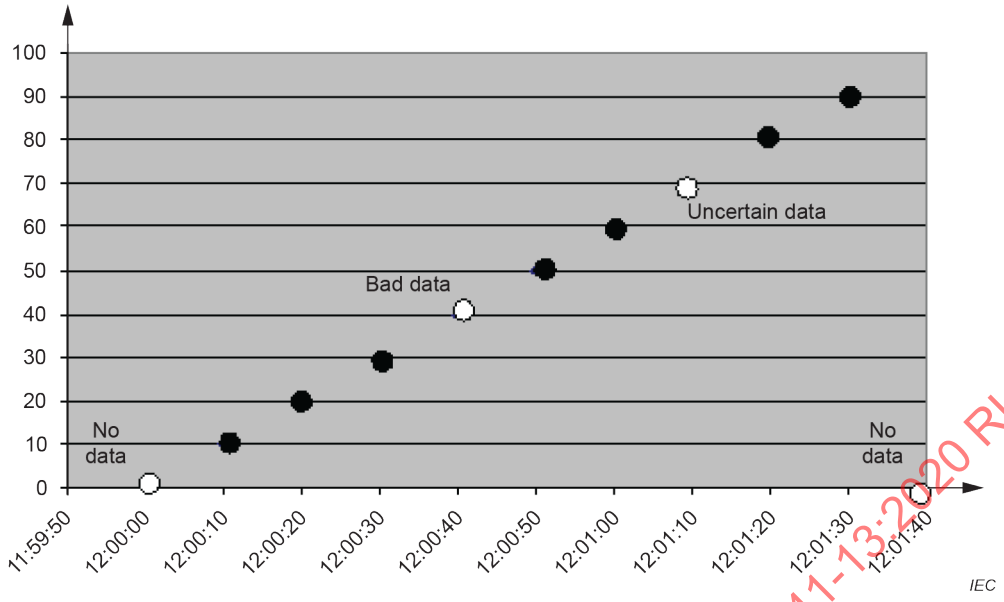


Figure A.1 – Historian 1

For the purposes of all Historian 1 examples:

1) *TreatUncertainAsBad* = False. Therefore Uncertain values are included in *Aggregate* calls.
2) *Stepped Attribute* = False. Therefore *SlopedInterpolation* is used between data points.
3) *UseSlopedExtrapolation* = False. Therefore *SteppedExtrapolation* is used at end boundary conditions.
4) *PercentBad* = 100, *PercentGood* = 100. Therefore if all values are Good then the quality will be Good, or if all values are Bad then the quality will be Bad, but if there is some Good and some Bad then the quality will be Uncertain.

A.1.2 Example Aggregate data – Historian 2

This example is included to illustrate non-periodic data. For the purposes of Historian 2 examples consider a source historian with the data shown in Figure A.2:

Timestamp	Value	StatusCode	Notes
12:00:00	-	Bad_NoData	First archive entry, Point created
12:00:02	10	Raw, Good	
12:00:25	20	Raw, Good	
12:00:28	25	Raw, Good	
12:00:39	30	Raw, Good	
12:00:42	-	Raw, Bad	Bad quality data received, Bad data entered
12:00:48	40	Raw, Good	Received Good *StatusCode* value
12:00:52	50	Raw, Good	
12:01:12	60	Raw, Good	
12:01:17	70	Raw, Uncertain	Value is flagged as questionable
12:01:23	70	Raw, Good	
12:01:26	80	Raw, Good	
12:01:30	90	Raw, Good	
	-	No Data	No more entries, awaiting next Value

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

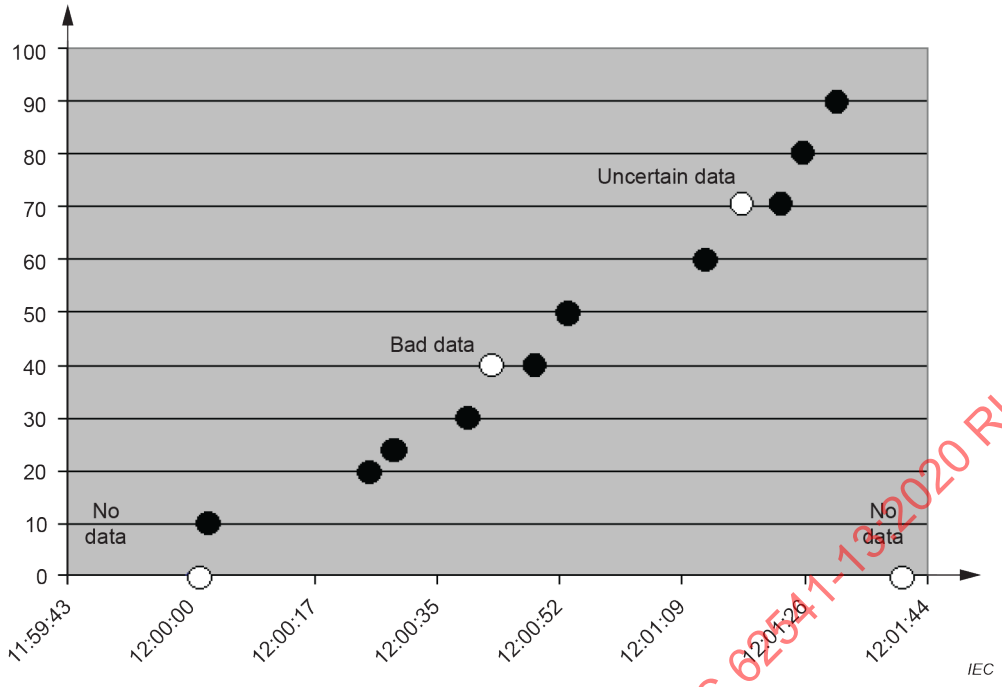


Figure A.2 – Historian 2

For the purposes of all Historian 2 examples:

1) *TreatUncertainAsBad* = True. Therefore Uncertain values are treated as Bad, and not included in the *Aggregate* call.
2) *Stepped Attribute* = False. Therefore *SlopedInterpolation* is used between data points.
3) *UseSlopedExtrapolation* = False. Therefore *SteppedExtrapolation* is used at end boundary conditions.
4) *PercentBad* = 100, *PercentGood* = 100. Therefore unless if all values are Good then the quality will be Good, or if all values are Bad then the quality will be Bad, but if there is some Good and some Bad then the quality will be Uncertain.

A.1.3 Example Aggregate data – Historian 3

This example is included to illustrate stepped data. For the purposes of Historian 3 examples consider a source historian with the data given in Figure A.3:

Timestamp	Value	StatusCode	Notes
12:00:00	-	Bad_NoData	First archive entry, Point created
12:00:02	10	Raw, Good	
12:00:25	20	Raw, Good	
12:00:28	25	Raw, Good	
12:00:39	30	Raw, Good	
12:00:42	-	Raw, Bad	Bad quality data received, Bad data entered
12:00:48	40	Raw, Good	Received Good *StatusCode* value
12:00:52	50	Raw, Good	
12:01:12	60	Raw, Good	
12:01:17	70	Raw, Uncertain	Value is flagged as questionable
12:01:23	70	Raw, Good	
12:01:26	80	Raw, Good	
12:01:30	90	Raw, Good	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Timestamp	Value	StatusCode	Notes
	-	No Data	No more entries, awaiting next Value

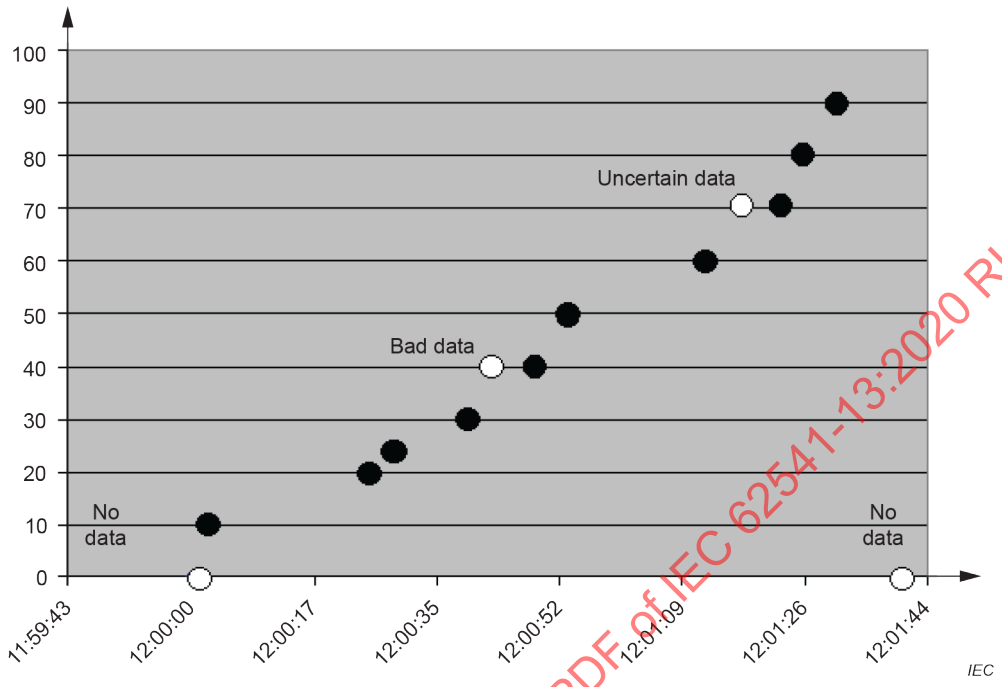


Figure A.3 – Historian 3

For the purposes of all Historian 3 examples:

1) *TreatUncertainAsBad* = True. Therefore Uncertain values are treated as Bad, and not included in the *Aggregate* call.
2) *Stepped Attribute* = True. Therefore *SteppedInterpolation* is used between data points.
3) *UseSlopedExtrapolation* = False. Therefore *SteppedExtrapolation* is used at end boundary conditions.
4) *PercentBad* = 50, *PercentGood* = 50. Therefore data will be either Good or Bad quality. Uncertain should not happen since a value is either Good or Bad.

A.1.4 Example Aggregate data – Historian 4

This example is included to illustrate Boolean data. For the purposes of Historian 4 examples consider a source historian with the following data:

Timestamp	Value	StatusCode	Notes
12:00:00	-	Bad_NoData	First archive entry, Point created
12:00:02	TRUE	Raw, Good	
12:00:25	FALSE	Raw, Good	
12:00:28	TRUE	Raw, Good	
12:00:39	TRUE	Raw, Good	
12:00:42	-	Raw, Bad	Bad quality data received, Bad data entered
12:00:48	TRUE	Raw, Good	Received Good *StatusCode* value
12:00:52	FALSE	Raw, Good	
12:01:12	FALSE	Raw, Good	
12:01:17	TRUE	Raw, Uncertain	Value is flagged as questionable

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Timestamp	Value	StatusCode	Notes
12:01:23	TRUE	Raw, Good	
12:01:26	FALSE	Raw, Good	
12:01:30	TRUE	Raw, Good	
	-	No Data	No more entries, awaiting next Value

For the purposes of all Historian 4 examples:

1) *TreatUncertainAsBad* = True. Therefore Uncertain values are treated as Bad, and not included in the *Aggregate* call.
2) *Stepped Attribute* = True. Therefore *SteppedInterpolation* is used between data points.
3) *UseSlopedExtrapolation* = False. Therefore *SteppedExtrapolation* is used at end boundary conditions.
4) *PercentGood = 100, PercentBad = 100.*

For Boolean data interpolation and extrapolation shall always be stepped.

A.2 Interpolative

A.2.1 Description

The following examples demonstrate Interpolative *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:05, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.2.2 Interpolative data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData	
12:00:05.000		BadNoData	
12:00:10.000	10	Good	
12:00:15.000	15	Good, Interpolated	
12:00:20.000	20	Good	
12:00:25.000	25	Good, Interpolated	
12:00:30.000	30	Good	
12:00:35.000	35	UncertainDataSubNormal, Interpolated	
12:00:40.000	40	UncertainDataSubNormal, Interpolated	
12:00:45.000	45	UncertainDataSubNormal, Interpolated	
12:00:50.000	50	Good	
12:00:55.000	55	Good, Interpolated	
12:01:00.000	60	Good	
12:01:05.000	65	UncertainDataSubNormal, Interpolated	
12:01:10.000	70	Uncertain	
12:01:15.000	75	UncertainDataSubNormal, Interpolated	
12:01:20.000	80	Good	
12:01:25.000	85	Good, Interpolated	
12:01:30.000	90	Good	
12:01:35.000	90	UncertainDataSubNormal, Interpolated	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData	
12:00:05.000	11.304	Good, Interpolated	
12:00:10.000	13.478	Good, Interpolated	
12:00:15.000	15.652	Good, Interpolated	
12:00:20.000	17.826	Good, Interpolated	
12:00:25.000	20	Good	
12:00:30.000	25.909	Good, Interpolated	
12:00:35.000	28.182	Good, Interpolated	
12:00:40.000	31.111	UncertainDataSubNormal, Interpolated	
12:00:45.000	36.667	UncertainDataSubNormal, Interpolated	
12:00:50.000	45	Good, Interpolated	
12:00:55.000	51.500	Good, Interpolated	
12:01:00.000	54	Good, Interpolated	
12:01:05.000	56.500	Good, Interpolated	
12:01:10.000	59	Good, Interpolated	
12:01:15.000	62.727	UncertainDataSubNormal, Interpolated	
12:01:20.000	67.273	UncertainDataSubNormal, Interpolated	
12:01:25.000	76.667	Good, Interpolated	
12:01:30.000	90	Good	
12:01:35.000	90	UncertainDataSubNormal, Interpolated	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData	
12:00:05.000	10	Good, Interpolated	
12:00:10.000	10	Good, Interpolated	
12:00:15.000	10	Good, Interpolated	
12:00:20.000	10	Good, Interpolated	
12:00:25.000	20	Good	
12:00:30.000	25	Good, Interpolated	
12:00:35.000	25	Good, Interpolated	
12:00:40.000	30	Good, Interpolated	
12:00:45.000	30	UncertainDataSubNormal, Interpolated	
12:00:50.000	40	Good, Interpolated	
12:00:55.000	50	Good, Interpolated	
12:01:00.000	50	Good, Interpolated	
12:01:05.000	50	Good, Interpolated	
12:01:10.000	50	Good, Interpolated	
12:01:15.000	60	Good, Interpolated	
12:01:20.000	60	UncertainDataSubNormal, Interpolated	
12:01:25.000	70	Good, Interpolated	
12:01:30.000	90	Good	
12:01:35.000	90	UncertainDataSubNormal, Interpolated	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.3 Average

A.3.1 Description

The following examples demonstrate Average *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:05, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.3.2 Average data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData	
12:00:05.000		BadNoData	
12:00:10.000	10	Good, Calculated	
12:00:15.000		BadNoData	
12:00:20.000	20	Good, Calculated	
12:00:25.000		BadNoData	
12:00:30.000	30	Good, Calculated	
12:00:35.000		BadNoData	
12:00:40.000		BadNoData	
12:00:45.000		BadNoData	
12:00:50.000	50	Good, Calculated	
12:00:55.000		BadNoData	
12:01:00.000	60	Good, Calculated	
12:01:05.000		BadNoData	
12:01:10.000		BadNoData	
12:01:15.000		BadNoData	
12:01:20.000	80	Good, Calculated	
12:01:25.000		BadNoData	
12:01:30.000	90	Good, Calculated	
12:01:35.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated	
12:00:05.000		BadNoData	
12:00:10.000		BadNoData	
12:00:15.000		BadNoData	
12:00:20.000		BadNoData	
12:00:25.000	22.500	Good, Calculated	
12:00:30.000		BadNoData	
12:00:35.000	30	Good, Calculated	
12:00:40.000		BadNoData	
12:00:45.000	40	Good, Calculated	
12:00:50.000	50	Good, Calculated	
12:00:55.000		BadNoData	
12:01:00.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:01:05.000		BadNoData	
12:01:10.000	60	Good, Calculated	
12:01:15.000		BadNoData	
12:01:20.000	70	Good, Calculated	
12:01:25.000	80	Good, Calculated	
12:01:30.000	90	Good, Calculated	
12:01:35.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated	
12:00:05.000		BadNoData	
12:00:10.000		BadNoData	
12:00:15.000		BadNoData	
12:00:20.000		BadNoData	
12:00:25.000	22.500	Good, Calculated	
12:00:30.000		BadNoData	
12:00:35.000	30	Good, Calculated	
12:00:40.000		BadNoData	
12:00:45.000	40	Good, Calculated	
12:00:50.000	50	Good, Calculated	
12:00:55.000		BadNoData	
12:01:00.000		BadNoData	
12:01:05.000		BadNoData	
12:01:10.000	60	Good, Calculated	
12:01:15.000		BadNoData	
12:01:20.000	70	Good, Calculated	
12:01:25.000	80	Good, Calculated	
12:01:30.000	90	Good, Calculated	
12:01:35.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.4 TimeAverage

A.4.1 Description

The following examples demonstrate TimeAverage *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:05, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.4.2 TimeAverage data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData	
12:00:05.000		BadNoData	
12:00:10.000	12.500	Good, Calculated	
12:00:15.000	17.500	Good, Calculated	
12:00:20.000	22.500	Good, Calculated	
12:00:25.000	27.500	Good, Calculated	
12:00:30.000	32.500	UncertainDataSubNormal, Calculated	
12:00:35.000	37.500	UncertainDataSubNormal, Calculated	
12:00:40.000	42.500	UncertainDataSubNormal, Calculated	
12:00:45.000	47.500	UncertainDataSubNormal, Calculated	
12:00:50.000	52.500	Good, Calculated	
12:00:55.000	57.500	Good, Calculated	
12:01:00.000	62.500	UncertainDataSubNormal, Calculated	
12:01:05.000	67.500	UncertainDataSubNormal, Calculated	
12:01:10.000	72.500	UncertainDataSubNormal, Calculated	
12:01:15.000	77.500	UncertainDataSubNormal, Calculated	
12:01:20.000	82.500	Good, Calculated	
12:01:25.000	87.500	Good, Calculated	
12:01:30.000	90	UncertainDataSubNormal, Calculated	
12:01:35.000	90	UncertainDataSubNormal, Calculated	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10.652	UncertainDataSubNormal, Calculated, Partial	
12:00:05.000	12.391	Good, Calculated	
12:00:10.000	14.565	Good, Calculated	
12:00:15.000	16.739	Good, Calculated	
12:00:20.000	18.913	Good, Calculated	
12:00:25.000	23.682	Good, Calculated	
12:00:30.000	27.046	Good, Calculated	
12:00:35.000	29.384	UncertainDataSubNormal, Calculated	
12:00:40.000	33.889	UncertainDataSubNormal, Calculated	
12:00:45.000	40	UncertainDataSubNormal, Calculated	
12:00:50.000	49.450	Good, Calculated	
12:00:55.000	52.750	Good, Calculated	
12:01:00.000	55.250	Good, Calculated	
12:01:05.000	57.750	Good, Calculated	
12:01:10.000	60.618	UncertainDataSubNormal, Calculated	
12:01:15.000	65	UncertainDataSubNormal, Calculated	
12:01:20.000	70.515	UncertainDataSubNormal, Calculated	
12:01:25.000	83.667	Good, Calculated	
12:01:30.000	90	UncertainDataSubNormal, Calculated	
12:01:35.000	90	UncertainDataSubNormal, Calculated	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10.652	UncertainDataSubNormal, Calculated, Partial	
12:00:05.000	12.391	Good, Calculated	
12:00:10.000	14.565	Good, Calculated	
12:00:15.000	16.739	Good, Calculated	
12:00:20.000	18.913	Good, Calculated	
12:00:25.000	23.682	Good, Calculated	
12:00:30.000	27.046	Good, Calculated	
12:00:35.000	29.384	UncertainDataSubNormal, Calculated	
12:00:40.000	33.889	UncertainDataSubNormal, Calculated	
12:00:45.000	40	UncertainDataSubNormal, Calculated	
12:00:50.000	49.450	Good, Calculated	
12:00:55.000	52.750	Good, Calculated	
12:01:00.000	55.250	Good, Calculated	
12:01:05.000	57.750	Good, Calculated	
12:01:10.000	60.618	UncertainDataSubNormal, Calculated	
12:01:15.000	65	UncertainDataSubNormal, Calculated	
12:01:20.000	70.515	UncertainDataSubNormal, Calculated	
12:01:25.000	83.667	Good, Calculated	
12:01:30.000	90	UncertainDataSubNormal, Calculated, Partial	
12:01:35.000	90	UncertainDataSubNormal, Calculated	

A.5 TimeAverage2

A.5.1 Description

The following examples demonstrate TimeAverage2 *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:05, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.5.2 TimeAverage2 data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData	
12:00:05.000		BadNoData	
12:00:10.000	12.500	Good, Calculated	
12:00:15.000	17.500	Good, Calculated	
12:00:20.000	22.500	Good, Calculated	
12:00:25.000	27.500	Good, Calculated	
12:00:30.000	30	UncertainDataSubNormal, Calculated	
12:00:35.000	30	UncertainDataSubNormal, Calculated	
12:00:40.000		BadNoData	
12:00:45.000		BadNoData	
12:00:50.000	52.500	Good, Calculated	
12:00:55.000	57.500	Good, Calculated	
12:01:00.000	62.500	UncertainDataSubNormal, Calculated	
12:01:05.000	67.500	UncertainDataSubNormal, Calculated	
12:01:10.000	72.500	UncertainDataSubNormal, Calculated	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:01:15.000	77.500	UncertainDataSubNormal, Calculated	
12:01:20.000	82.500	Good, Calculated	
12:01:25.000	87.500	Good, Calculated	
12:01:30.000	90	UncertainDataSubNormal, Calculated, Partial	
12:01:35.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10.652	UncertainDataSubNormal, Calculated, Partial	
12:00:05.000	12.391	Good, Calculated	
12:00:10.000	14.565	Good, Calculated	
12:00:15.000	16.739	Good, Calculated	
12:00:20.000	18.913	Good, Calculated	
12:00:25.000	23.682	Good, Calculated	
12:00:30.000	27.046	Good, Calculated	
12:00:35.000	29.273	UncertainDataSubNormal, Calculated	
12:00:40.000	30	UncertainDataSubNormal, Calculated	
12:00:45.000	42.500	UncertainDataSubNormal, Calculated	
12:00:50.000	49.450	Good, Calculated	
12:00:55.000	52.750	Good, Calculated	
12:01:00.000	55.250	Good, Calculated	
12:01:05.000	57.750	Good, Calculated	
12:01:10.000	59.800	UncertainDataSubNormal, Calculated	
12:01:15.000	60	UncertainDataSubNormal, Calculated	
12:01:20.000	73.333	UncertainDataSubNormal, Calculated	
12:01:25.000	83.667	Good, Calculated	
12:01:30.000	90	UncertainDataSubNormal, Calculated, Partial	
12:01:35.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated, Partial	
12:00:05.000	10	Good, Calculated	
12:00:10.000	10	Good, Calculated	
12:00:15.000	10	Good, Calculated	
12:00:20.000	10	Good, Calculated	
12:00:25.000	22	Good, Calculated	
12:00:30.000	25	Good, Calculated	
12:00:35.000	26	Good, Calculated	
12:00:40.000		Bad, Calculated	
12:00:45.000		Bad, Calculated	
12:00:50.000	46	Good, Calculated	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:55.000	50	Good, Calculated	
12:01:00.000	50	Good, Calculated	
12:01:05.000	50	Good, Calculated	
12:01:10.000	56	Good, Calculated	
12:01:15.000		Bad, Calculated	
12:01:20.000		Bad, Calculated	
12:01:25.000	78	Good, Calculated	
12:01:30.000	90	Good, Calculated, Partial	
12:01:35.000		BadNoData	

A.6 Total

A.6.1 Description

The following examples demonstrate Total *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:05, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.6.2 Total data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData	
12:00:05.000		BadNoData	
12:00:10.000	62.500	Good, Calculated	
12:00:15.000	87.500	Good, Calculated	
12:00:20.000	112.500	Good, Calculated	
12:00:25.000	137.500	Good, Calculated	
12:00:30.000	162.500	UncertainDataSubNormal, Calculated	
12:00:35.000	187.500	UncertainDataSubNormal, Calculated	
12:00:40.000	212.500	UncertainDataSubNormal, Calculated	
12:00:45.000	237.500	UncertainDataSubNormal, Calculated	
12:00:50.000	262.500	Good, Calculated	
12:00:55.000	287.500	Good, Calculated	
12:01:00.000	312.500	UncertainDataSubNormal, Calculated	
12:01:05.000	337.500	UncertainDataSubNormal, Calculated	
12:01:10.000	362.500	UncertainDataSubNormal, Calculated	
12:01:15.000	387.500	UncertainDataSubNormal, Calculated	
12:01:20.000	412.500	Good, Calculated	
12:01:25.000	437.500	Good, Calculated	
12:01:30.000	450	UncertainDataSubNormal, Calculated	
12:01:35.000	450	UncertainDataSubNormal, Calculated	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	31.957	UncertainDataSubNormal, Calculated, Partial	
12:00:05.000	61.957	Good, Calculated	
12:00:10.000	72.826	Good, Calculated	
12:00:15.000	83.696	Good, Calculated	
12:00:20.000	94.565	Good, Calculated	
12:00:25.000	118.409	Good, Calculated	
12:00:30.000	135.227	Good, Calculated	
12:00:35.000	146.919	UncertainDataSubNormal, Calculated	
12:00:40.000	169.444	UncertainDataSubNormal, Calculated	
12:00:45.000	200	UncertainDataSubNormal, Calculated	
12:00:50.000	247.250	Good, Calculated	
12:00:55.000	263.750	Good, Calculated	
12:01:00.000	276.250	Good, Calculated	
12:01:05.000	288.750	Good, Calculated	
12:01:10.000	303.091	UncertainDataSubNormal, Calculated	
12:01:15.000	325	UncertainDataSubNormal, Calculated	
12:01:20.000	352.576	UncertainDataSubNormal, Calculated	
12:01:25.000	418.333	Good, Calculated	
12:01:30.000	481.250	UncertainDataSubNormal, Calculated	
12:01:35.000	543.750	UncertainDataSubNormal, Calculated	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	30	UncertainDataSubNormal, Calculated, Partial	
12:00:05.000	50	Good, Calculated	
12:00:10.000	50	Good, Calculated	
12:00:15.000	50	Good, Calculated	
12:00:20.000	50	Good, Calculated	
12:00:25.000	110	Good, Calculated	
12:00:30.000	125	Good, Calculated	
12:00:35.000	130	Good, Calculated	
12:00:40.000	150	UncertainDataSubNormal, Calculated	
12:00:45.000	170	UncertainDataSubNormal, Calculated	
12:00:50.000	230	Good, Calculated	
12:00:55.000	250	Good, Calculated	
12:01:00.000	250	Good, Calculated	
12:01:05.000	250	Good, Calculated	
12:01:10.000	280	Good, Calculated	
12:01:15.000	300	UncertainDataSubNormal, Calculated	
12:01:20.000	320	UncertainDataSubNormal, Calculated	
12:01:25.000	390	Good, Calculated	
12:01:30.000	450	UncertainDataSubNormal, Calculated	
12:01:35.000	450	UncertainDataSubNormal, Calculated	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.7 Total2

A.7.1 Description

The following examples demonstrate Total2 *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:05, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.7.2 Total2 data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData	
12:00:05.000		BadNoData	
12:00:10.000	62.500	Good, Calculated	
12:00:15.000	87.500	Good, Calculated	
12:00:20.000	112.500	Good, Calculated	
12:00:25.000	137.500	Good, Calculated	
12:00:30.000	150	UncertainDataSubNormal, Calculated	
12:00:35.000	150	UncertainDataSubNormal, Calculated	
12:00:40.000		BadNoData	
12:00:45.000		BadNoData	
12:00:50.000	262.500	Good, Calculated	
12:00:55.000	287.500	Good, Calculated	
12:01:00.000	312.500	UncertainDataSubNormal, Calculated	
12:01:05.000	337.500	UncertainDataSubNormal, Calculated	
12:01:10.000	362.500	UncertainDataSubNormal, Calculated	
12:01:15.000	387.500	UncertainDataSubNormal, Calculated	
12:01:20.000	412.500	Good, Calculated	
12:01:25.000	437.500	Good, Calculated	
12:01:30.000	0.090	UncertainDataSubNormal, Calculated, Partial	
12:01:35.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	31.957	UncertainDataSubNormal, Calculated, Partial	
12:00:05.000	61.957	Good, Calculated	
12:00:10.000	72.826	Good, Calculated	
12:00:15.000	83.696	Good, Calculated	
12:00:20.000	94.565	Good, Calculated	
12:00:25.000	118.409	Good, Calculated	
12:00:30.000	135.227	Good, Calculated	
12:00:35.000	146.364	UncertainDataSubNormal, Calculated	
12:00:40.000	60	UncertainDataSubNormal, Calculated	
12:00:45.000	85	UncertainDataSubNormal, Calculated	
12:00:50.000	247.250	Good, Calculated	
12:00:55.000	263.750	Good, Calculated	
12:01:00.000	276.250	Good, Calculated	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:01:05.000	288.750	Good, Calculated	
12:01:10.000	299	UncertainDataSubNormal, Calculated	
12:01:15.000	120	UncertainDataSubNormal, Calculated	
12:01:20.000	146.667	UncertainDataSubNormal, Calculated	
12:01:25.000	418.333	Good, Calculated	
12:01:30.000	0.090	UncertainDataSubNormal, Calculated, Partial	
12:01:35.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	30	Good, Calculated, Partial	
12:00:05.000	50	Good, Calculated	
12:00:10.000	50	Good, Calculated	
12:00:15.000	50	Good, Calculated	
12:00:20.000	50	Good, Calculated	
12:00:25.000	110	Good, Calculated	
12:00:30.000	125	Good, Calculated	
12:00:35.000	130	Good, Calculated	
12:00:40.000		Bad, Calculated	
12:00:45.000		Bad, Calculated	
12:00:50.000	230	Good, Calculated	
12:00:55.000	250	Good, Calculated	
12:01:00.000	250	Good, Calculated	
12:01:05.000	250	Good, Calculated	
12:01:10.000	280	Good, Calculated	
12:01:15.000		Bad, Calculated	
12:01:20.000		Bad, Calculated	
12:01:25.000	390	Good, Calculated	
12:01:30.000	0.090	Good, Calculated, Partial	
12:01:35.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.8 Minimum

A.8.1 Description

The following examples demonstrate Minimum *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.8.2 Minimum data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated, Partial	
12:00:16.000	20	Good, Calculated	
12:00:32.000		BadNoData	
12:00:48.000	50	Good, Calculated	
12:01:04.000		BadNoData	
12:01:20.000	80	Good, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated, Partial	
12:00:16.000	20	Good, Calculated	
12:00:32.000	30	UncertainDataSubNormal, Calculated	
12:00:48.000	40	Good	
12:01:04.000	60	UncertainDataSubNormal, Calculated	
12:01:20.000	70	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated, Partial	
12:00:16.000	20	Good, Calculated	
12:00:32.000	30	UncertainDataSubNormal, Calculated	
12:00:48.000	40	Good	
12:01:04.000	60	UncertainDataSubNormal, Calculated	
12:01:20.000	70	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.9 Maximum

A.9.1 Description

The following examples demonstrate Maximum *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.9.2 Maximum data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated, Partial	
12:00:16.000	30	Good, Calculated	
12:00:32.000		BadNoData	
12:00:48.000	60	Good, Calculated	
12:01:04.000		BadNoData	
12:01:20.000	90	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated, Partial	
12:00:16.000	25	Good, Calculated	
12:00:32.000	30	UncertainDataSubNormal, Calculated	
12:00:48.000	50	Good, Calculated	
12:01:04.000	60	UncertainDataSubNormal, Calculated	
12:01:20.000	90	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated, Partial	
12:00:16.000	25	Good, Calculated	
12:00:32.000	30	UncertainDataSubNormal, Calculated	
12:00:48.000	50	Good, Calculated	
12:01:04.000	60	UncertainDataSubNormal, Calculated	
12:01:20.000	90	Good, Calculated, Partial	
12:01:36.000		BadNoData	

A.10 MininumActualTime

A.10.1 Description

The following examples demonstrate the MinimumActualTime *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.10.2 MinimumActualTime data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:10.000	10	Good, Partial	
12:00:20.000	20	Good	
12:00:32.000		BadNoData	
12:00:50.000	50	Good	
12:01:04.000		BadNoData	
12:01:20.000	80	Good, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	Good, Partial	
12:00:25.000	20	Good	
12:00:39.000	30	UncertainDataSubNormal	
12:00:48.000	40	Good	
12:01:12.000	60	UncertainDataSubNormal	
12:01:23.000	70	Good, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	Good, Partial	
12:00:25.000	20	Good	
12:00:39.000	30	UncertainDataSubNormal	
12:00:48.000	40	Good	
12:01:12.000	60	UncertainDataSubNormal	
12:01:23.000	70	Good, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.11 MaximumActualTime

A.11.1 Description

The following examples demonstrate MaximumActualTime *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.11.2 MaximumActualTime data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:10.000	10	Good, Partial	
12:00:30.000	30	Good	
12:00:32.000		BadNoData	
12:01:00.000	60	Good	
12:01:04.000		BadNoData	
12:01:30.000	90	Good, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	Good, Partial	
12:00:28.000	25	Good	
12:00:39.000	30	UncertainDataSubNormal	
12:00:52.000	50	Good	
12:01:12.000	60	UncertainDataSubNormal	
12:01:30.000	90	Good, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	Good, Partial	
12:00:28.000	25	Good	
12:00:39.000	30	UncertainDataSubNormal	
12:00:52.000	50	Good	
12:01:12.000	60	UncertainDataSubNormal	
12:01:30.000	90	Good, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.12 Range

A.12.1 Description

The following examples demonstrate Range *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.12.2 Range data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:16.000	10	Good, Calculated	
12:00:32.000		BadNoData	
12:00:48.000	10	Good, Calculated	
12:01:04.000		BadNoData	
12:01:20.000	10	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:16.000	5	Good, Calculated	
12:00:32.000	0	UncertainDataSubNormal, Calculated	
12:00:48.000	10	Good, Calculated	
12:01:04.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	20	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:16.000	5	Good, Calculated	
12:00:32.000	0	UncertainDataSubNormal, Calculated	
12:00:48.000	10	Good, Calculated	
12:01:04.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	20	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.13 Minimum2

A.13.1 Description

The following examples demonstrate Minimum2 *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.13.2 Minimum2 data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	UncertainDataSubNormal, Calculated, Partial	
12:00:16.000	16	UncertainDataSubNormal, Interpolated	
12:00:32.000	30	UncertainDataSubNormal, Interpolated	
12:00:48.000	50	UncertainDataSubNormal, Calculated	
12:01:04.000	64	UncertainDataSubNormal, Interpolated	
12:01:20.000	80	UncertainDataSubNormal, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	UncertainDataSubNormal, Calculated, Partial	
12:00:16.000	16.087	Good, Interpolated	
12:00:32.000	26.818	UncertainDataSubNormal, Interpolated	
12:00:48.000	40	Good	
12:01:04.000	56	UncertainDataSubNormal, Interpolated	
12:01:20.000	70	UncertainDataSubNormal, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated, Partial	
12:00:16.000	10	Good, Interpolated	
12:00:32.000	25	Good, Interpolated	
12:00:48.000	40	Good	
12:01:04.000	50	Good, Interpolated	
12:01:20.000	70	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.14 Maximum2

A.14.1 Description

The following examples demonstrate Maximum2 *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.14.2 Maximum2 data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	16	UncertainDataSubNormal, Interpolated, Partial	
12:00:16.000	30	UncertainDataSubNormal, Calculated, MultipleValues	
12:00:32.000	30	UncertainDataSubNormal, Interpolated	
12:00:48.000	64	UncertainDataSubNormal, Interpolated	
12:01:04.000	80	UncertainDataSubNormal, Calculated	
12:01:20.000	90	UncertainDataSubNormal, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	16.087	UncertainDataSubNormal, Interpolated, Partial	
12:00:16.000	26.818	Good, Interpolated	
12:00:32.000	40	UncertainDataSubNormal, Calculated	
12:00:48.000	56	Good, Interpolated	
12:01:04.000	60	UncertainDataSubNormal, Calculated	
12:01:20.000	90	UncertainDataSubNormal, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated, Partial	
12:00:16.000	25	Good, Calculated	
12:00:32.000	30	Good, Calculated	
12:00:48.000	50	Good, Calculated	
12:01:04.000	60	Good, Calculated	
12:01:20.000	90	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.15 MinimumActualTime2

A.15.1 Description

The following examples demonstrate MinimumActualTime2 *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.15.2 MinimumActualTime2 data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:10.000	10	UncertainDataSubNormal, Partial	
12:00:16.000	16	UncertainDataSubNormal, Interpolated	
12:00:32.000	30	UncertainDataSubNormal, Interpolated	
12:00:50.000	50	UncertainDataSubNormal	
12:01:04.000	64	UncertainDataSubNormal, Interpolated	
12:01:20.000	80	UncertainDataSubNormal, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	UncertainDataSubNormal, Partial	
12:00:16.000	16.087	Good, Interpolated	
12:00:32.000	26.818	UncertainDataSubNormal, Interpolated	
12:00:48.000	40	Good	
12:01:04.000	56	UncertainDataSubNormal, Interpolated	
12:01:23.000	70	UncertainDataSubNormal, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	Good, Partial	
12:00:16.000	10	Good, Interpolated	
12:00:32.000	25	Good, Interpolated	
12:00:48.000	40	Good	
12:01:04.000	50	Good, Interpolated	
12:01:23.000	70	Good, Partial	
12:01:36.000		BadNoData	

A.16 MaximumActualTime2

A.16.1 Description

The following examples demonstrate MaximumActualTime2 *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.16.2 MaximumActualTime2 data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:15.999	16	UncertainDataSubNormal, Interpolated, Partial	
12:00:30.000	30	UncertainDataSubNormal, MultipleValues	
12:00:32.000	30	UncertainDataSubNormal, Interpolated	
12:01:03.999	64	UncertainDataSubNormal, Interpolated	
12:01:19.999	80	UncertainDataSubNormal, Interpolated	
12:01:30.000	90	UncertainDataSubNormal, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:15.999	16.087	UncertainDataSubNormal, Interpolated, Partial	
12:00:31.999	26.818	Good, Interpolated	
12:00:47.999	40	UncertainDataSubNormal, Interpolated	
12:01:03.999	56	Good, Interpolated	
12:01:12.000	60	UncertainDataSubNormal	
12:01:30.000	90	UncertainDataSubNormal, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	Good, Partial	
12:00:28.000	25	Good	
12:00:39.000	30	Good	
12:00:52.000	50	Good	
12:01:12.000	60	Good	
12:01:30.000	90	Good, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.17 Range2

A.17.1 Description

The following examples demonstrate Range2 *Aggregate* scenarios. This *Aggregate* does not apply to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.17.2 Range2 data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	6	UncertainDataSubNormal, Calculated, Partial	
12:00:16.000	14	UncertainDataSubNormal, Calculated	
12:00:32.000	0	UncertainDataSubNormal, Calculated	
12:00:48.000	14	UncertainDataSubNormal, Calculated	
12:01:04.000	16	UncertainDataSubNormal, Calculated	
12:01:20.000	10	UncertainDataSubNormal, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	6.087	UncertainDataSubNormal, Calculated, Partial	
12:00:16.000	10.731	Good, Calculated	
12:00:32.000	13.182	UncertainDataSubNormal, Calculated	
12:00:48.000	16	Good, Calculated	
12:01:04.000	4	UncertainDataSubNormal, Calculated	
12:01:20.000	20	UncertainDataSubNormal, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:16.000	15	Good, Calculated	
12:00:32.000	5	Good, Calculated	
12:00:48.000	10	Good, Calculated	
12:01:04.000	10	Good, Calculated	
12:01:20.000	20	Good, Calculated, Partial	
12:01:36.000		BadNoData	

A.18 AnnotationCount

A.18.1 Description

The following examples demonstrate AnnotationCount *Aggregate* scenarios. This *Aggregate* does not apply to current values, since annotations are features of historical data. ***ProcessingInterval***: 00:01:00, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.18.2 AnnotationCount data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	3	Good, Calculated	
12:01:00.000	1	Good, Calculated	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated	
12:01:00.000	0	Good, Calculated	

A.19 Count

A.19.1 Description

The following examples demonstrate Count *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.19.2 Count data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	1	Good, Calculated, Partial	
12:00:16.000	2	Good, Calculated	
12:00:32.000		Bad	
12:00:48.000	2	Good, Calculated	
12:01:04.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	2	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	1	Good, Calculated, Partial	
12:00:16.000	2	Good, Calculated	
12:00:32.000	1	UncertainDataSubNormal, Calculated	
12:00:48.000	2	Good, Calculated	
12:01:04.000	1	UncertainDataSubNormal, Calculated	
12:01:20.000	3	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	1	Good, Calculated, Partial	
12:00:16.000	2	Good, Calculated	
12:00:32.000		Bad	
12:00:48.000	2	Good, Calculated	
12:01:04.000	1	Good, Calculated	
12:01:20.000	3	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Historian 4			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	1	Good, Calculated, Partial	
12:00:16.000	2	Good, Calculated	
12:00:32.000	1	UncertainDataSubNormal, Calculated	
12:00:48.000	2	Good, Calculated	
12:01:04.000	1	UncertainDataSubNormal, Calculated	
12:01:20.000	3	Good, Calculated, Partial	
12:01:36.000		BadNoData	

A.20 DurationInStateZero

A.20.1 Description

The following examples demonstrate DurationInStateZero *Aggregate* scenarios. The *Aggregate* only applies to Historian 4. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.20.2 DurationInStateZero data

Historian 4			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	UncertainDataSubNormal, Calculated, Partial	
12:00:16.000	3000	Good, Calculated	
12:00:32.000	0	UncertainDataSubNormal, Calculated	
12:00:48.000	12000	Good, Calculated	
12:01:04.000	13000	UncertainDataSubNormal, Calculated	
12:01:20.000	4000	UncertainDataSubNormal, Calculated, Partial	
12:01:36.000		BadNoData	

A.21 DurationInStateNonZero

A.21.1 Description

The following examples demonstrate DurationInStateNonZero *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.21.2 DurationInStateNonZero data

Historian 4			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	14000	UncertainDataSubNormal, Calculated, Partial	
12:00:16.000	13000	Good, Calculated	
12:00:32.000	10000	UncertainDataSubNormal, Calculated	
12:00:48.000	4000	Good, Calculated	
12:01:04.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	3001	UncertainDataSubNormal, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.22 NumberOfTransitions

A.22.1 Description

The following examples demonstrate NumberOfTransitions *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.22.2 NumberOfTransitions data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	1	Good, Calculated, Partial	
12:00:16.000	2	Good, Calculated	
12:00:32.000		Bad	
12:00:48.000	2	Good, Calculated	
12:01:04.000	1	UncertainDataSubNormal, Calculated	
12:01:20.000	2	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	1	Good, Calculated, Partial	
12:00:16.000	2	Good, Calculated	
12:00:32.000	1	UncertainDataSubNormal, Calculated	
12:00:48.000	2	Good, Calculated	
12:01:04.000	1	UncertainDataSubNormal, Calculated	
12:01:20.000	3	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	1	Good, Calculated, Partial	
12:00:16.000	2	Good, Calculated	
12:00:32.000		Bad	
12:00:48.000	2	Good, Calculated	
12:01:04.000	1	Good, Calculated	
12:01:20.000	3	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 4			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	1	Good, Calculated, Partial	
12:00:16.000	2	Good, Calculated	
12:00:32.000	0	UncertainDataSubNormal, Calculated	
12:00:48.000	1	Good, Calculated	
12:01:04.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	3	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.23 Start

A.23.1 Description

The following examples demonstrate Start *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.23.2 Start data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:10.000	10	Good, Partial	
12:00:20.000	20	Good	
12:00:40.000		Bad	
12:00:50.000	50	Good	
12:01:10.000	70	Uncertain	
12:01:20.000	80	Good, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	Good, Partial	
12:00:25.000	20	Good	
12:00:39.000	30	Good	
12:00:48.000	40	Good	
12:01:12.000	60	Good	
12:01:23.000	70	Good, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	Good, Partial	
12:00:25.000	20	Good	
12:00:39.000	30	Good	
12:00:48.000	40	Good	
12:01:12.000	60	Good	
12:01:23.000	70	Good, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.24 End

A.24.1 Description

The following examples demonstrate End *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.24.2 End data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:10.000	10	Good, Partial	
12:00:30.000	30	Good	
12:00:40.000		Bad	
12:01:00.000	60	Good	
12:01:10.000	70	Uncertain	
12:01:30.000	90	Good, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	Good, Partial	
12:00:28.000	25	Good	
12:00:42.000		Bad	
12:00:52.000	50	Good	
12:01:17.000	70	Uncertain	
12:01:30.000	90	Good, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:02.000	10	Good, Partial	
12:00:28.000	25	Good	
12:00:42.000		Bad	
12:00:52.000	50	Good	
12:01:17.000	70	Uncertain	
12:01:30.000	90	Good, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.25 StartBound

A.25.1 Description

The following examples demonstrate StartBound *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.25.2 StartBound data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData, Partial	
12:00:16.000	16	Good, Interpolated	
12:00:32.000	30	UncertainDataSubNormal, Interpolated	
12:00:48.000		BadNoData	
12:01:04.000	64	UncertainDataSubNormal, Interpolated	
12:01:20.000	80	Good, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData, Partial	
12:00:16.000	16.087	Good, Interpolated	
12:00:32.000	26.818	Good, Interpolated	
12:00:48.000	40	Good	
12:01:04.000	56	Good, Interpolated	
12:01:20.000		BadNoData, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData, Partial	
12:00:16.000	10	Good, Interpolated	
12:00:32.000	25	Good, Interpolated	
12:00:48.000	40	Good	
12:01:04.000	50	Good, Interpolated	
12:01:20.000		BadNoData, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.26 EndBound

A.26.1 Description

The following examples demonstrate End *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.26.2 EndBound data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	16	Good, Calculated, Partial	
12:00:16.000	30	UncertainDataSubNormal, Calculated	
12:00:32.000		BadNoData	
12:00:48.000	64	UncertainDataSubNormal, Calculated	
12:01:04.000	80	Good, Calculated	
12:01:20.000		BadNoData, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	16.087	Good, Calculated, Partial	
12:00:16.000	26.818	Good, Calculated	
12:00:32.000	40	Good, Calculated	
12:00:48.000	56	Good, Calculated	
12:01:04.000		BadNoData	
12:01:20.000		BadNoData, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10	Good, Calculated, Partial	
12:00:16.000	25	Good, Calculated	
12:00:32.000	40	Good, Calculated	
12:00:48.000	50	Good, Calculated	
12:01:04.000		BadNoData	
12:01:20.000		BadNoData, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.27 Delta

A.27.1 Description

The following examples demonstrate Delta *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.27.2 Delta data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:16.000	10	Good, Calculated	
12:00:32.000	0	BadNoData	
12:00:48.000	10	Good, Calculated	
12:01:04.000	0	BadNoData	
12:01:20.000	10	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:16.000	5	Good, Calculated	
12:00:32.000	0	UncertainDataSubNormal, Calculated	
12:00:48.000	10	Good, Calculated	
12:01:04.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	20	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:16.000	5	Good, Calculated	
12:00:32.000	0	UncertainDataSubNormal, Calculated	
12:00:48.000	10	Good, Calculated	
12:01:04.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	20	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.28 DeltaBounds

A.28.1 Description

The following examples demonstrate DeltaBounds *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.28.2 DeltaBounds data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData, Partial	
12:00:16.000	14	UncertainDataSubNormal, Calculated	
12:00:32.000		BadNoData	
12:00:48.000		BadNoData	
12:01:04.000	16	UncertainDataSubNormal, Calculated	
12:01:20.000		BadNoData, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData, Partial	
12:00:16.000	10.731	Good, Calculated	
12:00:32.000	13.182	Good, Calculated	
12:00:48.000	16	Good, Calculated	
12:01:04.000		BadNoData	
12:01:20.000		BadNoData, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000		BadNoData, Partial	
12:00:16.000	15	Good, Calculated	
12:00:32.000	15	Good, Calculated	
12:00:48.000	10	Good, Calculated	
12:01:04.000		BadNoData	
12:01:20.000		BadNoData, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.29 DurationGood

A.29.1 Description

The following examples demonstrate DurationGood *Aggregate* scenarios. Duration values are in milliseconds. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.29.2 DurationGood data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	6000	Good, Calculated, Partial	
12:00:16.000	16000	Good, Calculated	
12:00:32.000	0	Good, Calculated	
12:00:48.000	14000	Good, Calculated	
12:01:04.000	0	Good, Calculated	
12:01:20.000	10001	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	14000	Good, Calculated, Partial	
12:00:16.000	16000	Good, Calculated	
12:00:32.000	10000	Good, Calculated	
12:00:48.000	16000	Good, Calculated	
12:01:04.000	13000	Good, Calculated	
12:01:20.000	7001	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	14000	Good, Calculated, Partial	
12:00:16.000	16000	Good, Calculated	
12:00:32.000	10000	Good, Calculated	
12:00:48.000	16000	Good, Calculated	
12:01:04.000	13000	Good, Calculated	
12:01:20.000	7001	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 4			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	14000	Good, Calculated, Partial	
12:00:16.000	16000	Good, Calculated	
12:00:32.000	10000	Good, Calculated	
12:00:48.000	16000	Good, Calculated	
12:01:04.000	13000	Good, Calculated	
12:01:20.000	7001	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.30 DurationBad

A.30.1 Description

The following examples demonstrate DurationBad *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.30.2 DurationBad data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	10000	Good, Calculated, Partial	
12:00:16.000	0	Good, Calculated	
12:00:32.000	8000	Good, Calculated	
12:00:48.000	2000	Good, Calculated	
12:01:04.000	0	Good, Calculated	
12:01:20.000	0	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	2000	Good, Calculated, Partial	
12:00:16.000	0	Good, Calculated	
12:00:32.000	6000	Good, Calculated	
12:00:48.000	0	Good, Calculated	
12:01:04.000	3000	Good, Calculated	
12:01:20.000	3000	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	2000	Good, Calculated, Partial	
12:00:16.000	0	Good, Calculated	
12:00:32.000	6000	Good, Calculated	
12:00:48.000	0	Good, Calculated	
12:01:04.000	3000	Good, Calculated	
12:01:20.000	3000	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 4			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	2000	Good, Calculated, Partial	
12:00:16.000	0	Good, Calculated	
12:00:32.000	6000	Good, Calculated	
12:00:48.000	0	Good, Calculated	
12:01:04.000	3000	Good, Calculated	
12:01:20.000	3000	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.31 PercentGood

A.31.1 Description

The following examples demonstrate PercentGood *Aggregate* scenarios. **ProcessingInterval**: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.31.2 PercentGood data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	37.500	Good, Calculated, Partial	
12:00:16.000	100	Good, Calculated	
12:00:32.000	0	Good, Calculated	
12:00:48.000	87.500	Good, Calculated	
12:01:04.000	0	Good, Calculated	
12:01:20.000	100	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	87.500	Good, Calculated, Partial	
12:00:16.000	100	Good, Calculated	
12:00:32.000	62.500	Good, Calculated	
12:00:48.000	100	Good, Calculated	
12:01:04.000	81.250	Good, Calculated	
12:01:20.000	70.003	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	87.500	Good, Calculated, Partial	
12:00:16.000	100	Good, Calculated	
12:00:32.000	62.500	Good, Calculated	
12:00:48.000	100	Good, Calculated	
12:01:04.000	81.250	Good, Calculated	
12:01:20.000	70.003	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 4			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	87.500	Good, Calculated, Partial	
12:00:16.000	100	Good, Calculated	
12:00:32.000	62.500	Good, Calculated	
12:00:48.000	100	Good, Calculated	
12:01:04.000	81.250	Good, Calculated	
12:01:20.000	70.003	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.32 PercentBad

A.32.1 Description

The following examples demonstrate PercentBad *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.32.2 PercentBad data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	62.500	Good, Calculated, Partial	
12:00:16.000	0	Good, Calculated	
12:00:32.000	50	Good, Calculated	
12:00:48.000	12.500	Good, Calculated	
12:01:04.000	0	Good, Calculated	
12:01:20.000	0	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	12.500	Good, Calculated, Partial	
12:00:16.000	0	Good, Calculated	
12:00:32.000	37.500	Good, Calculated	
12:00:48.000	0	Good, Calculated	
12:01:04.000	18.750	Good, Calculated	
12:01:20.000	29.997	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	12.500	Good, Calculated, Partial	
12:00:16.000	0	Good, Calculated	
12:00:32.000	37.500	Good, Calculated	
12:00:48.000	0	Good, Calculated	
12:01:04.000	18.750	Good, Calculated	
12:01:20.000	29.997	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 4			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	12.500	Good, Calculated, Partial	
12:00:16.000	0	Good, Calculated	
12:00:32.000	37.500	Good, Calculated	
12:00:48.000	0	Good, Calculated	
12:01:04.000	18.750	Good, Calculated	
12:01:20.000	29.997	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.33 WorstQuality

A.33.1 Description

The following examples demonstrate WorstQuality *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.33.2 WorstQuality data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	Good	Good, Calculated, Partial	
12:00:16.000	Good	Good, Calculated	
12:00:32.000	Bad	Good, Calculated	
12:00:48.000	Good	Good, Calculated	
12:01:04.000	Uncertain	Good, Calculated	
12:01:20.000	Good	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	Good	Good, Calculated, Partial	
12:00:16.000	Good	Good, Calculated	
12:00:32.000	Bad	Good, Calculated	
12:00:48.000	Good	Good, Calculated	
12:01:04.000	Uncertain	Good, Calculated	
12:01:20.000	Good	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	Good	Good, Calculated, Partial	
12:00:16.000	Good	Good, Calculated	
12:00:32.000	Bad	Good, Calculated	
12:00:48.000	Good	Good, Calculated	
12:01:04.000	Uncertain	Good, Calculated	
12:01:20.000	Good	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 4			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	Good	Good, Calculated, Partial	
12:00:16.000	Good	Good, Calculated	
12:00:32.000	Bad	Good, Calculated	
12:00:48.000	Good	Good, Calculated	
12:01:04.000	Uncertain	Good, Calculated	
12:01:20.000	Good	Good, Calculated, Partial	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.34 WorstQuality2

A.34.1 Description

The following examples demonstrate WorstQuality2 *Aggregate* scenarios. ***ProcessingInterval***: 00:00:16, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.34.2 WorstQuality2 data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	BadNoData	Good, Calculated, Partial	
12:00:16.000	UncertainDataSubNormal	Good, Calculated	
12:00:32.000	Bad	Good, Calculated, MultipleValues	
12:00:48.000	BadNoData	Good, Calculated	
12:01:04.000	UncertainDataSubNormal	Good, Calculated, MultipleValues	
12:01:20.000	BadNoData	Good, Calculated, Partial	
12:01:36.000		BadNoData	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	BadNoData	Good, Calculated, Partial	
12:00:16.000	Good	Good, Calculated	
12:00:32.000	Bad	Good, Calculated	
12:00:48.000	Good	Good, Calculated	
12:01:04.000	BadNoData	Good, Calculated	
12:01:20.000	BadNoData	Good, Calculated, Partial, MultipleValues	
12:01:36.000		BadNoData	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	BadNoData	Good, Calculated, Partial	
12:00:16.000	Good	Good, Calculated	
12:00:32.000	Bad	Good, Calculated	
12:00:48.000	Good	Good, Calculated	
12:01:04.000	BadNoData	Good, Calculated	
12:01:20.000	BadNoData	Good, Calculated, Partial, MultipleValues	
12:01:36.000		BadNoData	

Historian 4			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	BadNoData	Good, Calculated, Partial	
12:00:16.000	Good	Good, Calculated	
12:00:32.000	Bad	Good, Calculated	
12:00:48.000	Good	Good, Calculated	
12:01:04.000	BadNoData	Good, Calculated	
12:01:20.000	BadNoData	Good, Calculated, Partial, MultipleValues	
12:01:36.000		BadNoData	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.35 StandardDeviationSample

A.35.1 Description

The following examples demonstrate StandardDeviationSample *Aggregate* scenarios. ***ProcessingInterval***: 00:00:20, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.35.2 StandardDeviationSample data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:20.000	7.071	Good, Calculated	
12:00:40.000	0	UncertainDataSubNormal, Calculated	
12:01:00.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	7.071	Good, Calculated, Partial	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:20.000	5	Good, Calculated	
12:00:40.000	7.071	UncertainDataSubNormal, Calculated	
12:01:00.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	10	Good, Calculated, Partial	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:20.000	5	Good, Calculated	
12:00:40.000	7.071	UncertainDataSubNormal, Calculated	
12:01:00.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	10	Good, Calculated, Partial	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

A.36 VarianceSample

A.36.1 Description

The following examples demonstrate VarianceSample *Aggregate* scenarios. ***ProcessingInterval***: 00:00:20, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.36.2 VarianceSample data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:20.000	50	Good, Calculated	
12:00:40.000	0	UncertainDataSubNormal, Calculated	
12:01:00.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	50	Good, Calculated, Partial	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:20.000	25	Good, Calculated	
12:00:40.000	50	UncertainDataSubNormal, Calculated	
12:01:00.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	100	Good, Calculated, Partial	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:20.000	25	Good, Calculated	
12:00:40.000	50	UncertainDataSubNormal, Calculated	
12:01:00.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	100	Good, Calculated, Partial	

A.37 StandardDeviationPopulation

A.37.1 Description

The following examples demonstrate StandardDeviationPopulation *Aggregate* scenarios. ***ProcessingInterval***: 00:00:20, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.37.2 StandardDeviationPopulation data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:20.000	5	Good, Calculated	
12:00:40.000	0	UncertainDataSubNormal, Calculated	
12:01:00.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	5	Good, Calculated, Partial	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:20.000	4.082	Good, Calculated	
12:00:40.000	5	UncertainDataSubNormal, Calculated	
12:01:00.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	8.165	Good, Calculated, Partial	

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:20.000	4.082	Good, Calculated	
12:00:40.000	5	UncertainDataSubNormal, Calculated	
12:01:00.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	8.165	Good, Calculated, Partial	

A.38 VariancePopulation

A.38.1 Description

The following examples demonstrate VariancePopulation *Aggregate* scenarios. ***ProcessingInterval***: 00:00:20, ***StartTime***: 12:00:00, ***EndTime***: 12:01:40.

A.38.2 VariancePopulation data

Historian 1			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:20.000	25	Good, Calculated	
12:00:40.000	0	UncertainDataSubNormal, Calculated	
12:01:00.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	25	Good, Calculated, Partial	

Historian 2			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:20.000	16.667	Good, Calculated	
12:00:40.000	25	UncertainDataSubNormal, Calculated	
12:01:00.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	66.667	Good, Calculated, Partial	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Historian 3			
Timestamp	**Value**	**StatusCode**	**Notes**
12:00:00.000	0	Good, Calculated, Partial	
12:00:20.000	16.667	Good, Calculated	
12:00:40.000	25	UncertainDataSubNormal, Calculated	
12:01:00.000	0	UncertainDataSubNormal, Calculated	
12:01:20.000	66.667	Good, Calculated, Partial	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

SOMMAIRE

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

_____

ARCHITECTURE UNIFIÉE OPC –

Partie 13: Agrégats

AVANT-PROPOS

1) La Commission Electrotechnique Internationale (IEC) est une organisation mondiale de normalisation composée de l'ensemble des comités électrotechniques nationaux (Comités nationaux de l'IEC). L'IEC a pour objet de favoriser la coopération internationale pour toutes les questions de normalisation dans les domaines de l'électricité et de l'électronique. A cet effet, l'IEC – entre autres activités – publie des Normes internationales, des Spécifications techniques, des Rapports techniques, des Spécifications accessibles au public (PAS) et des Guides (ci-après dénommés "Publication(s) de l'IEC"). Leur élaboration est confiée à des comités d'études, aux travaux desquels tout Comité national intéressé par le sujet traité peut participer. Les organisations internationales, gouvernementales et non gouvernementales, en liaison avec l'IEC, participent également aux travaux. L'IEC collabore étroitement avec l'Organisation Internationale de Normalisation (ISO), selon des conditions fixées par accord entre les deux organisations.
2) Les décisions ou accords officiels de l'IEC concernant les questions techniques représentent, dans la mesure du possible, un accord international sur les sujets étudiés, étant donné que les Comités nationaux de l'IEC intéressés sont représentés dans chaque comité d'études.
3) Les Publications de l'IEC se présentent sous la forme de recommandations internationales et sont agréées comme telles par les Comités nationaux de l'IEC. Tous les efforts raisonnables sont entrepris afin que l'IEC s'assure de l'exactitude du contenu technique de ses publications; l'IEC ne peut pas être tenue responsable de l'éventuelle mauvaise utilisation ou interprétation qui en est faite par un quelconque utilisateur final.
4) Dans le but d'encourager l'uniformité internationale, les Comités nationaux de l'IEC s'engagent, dans toute la mesure possible, à appliquer de façon transparente les Publications de l'IEC dans leurs publications nationales et régionales. Toutes divergences entre toutes Publications de l'IEC et toutes publications nationales ou régionales correspondantes doivent être indiquées en termes clairs dans ces dernières.
5) L'IEC elle-même ne fournit aucune attestation de conformité. Des organismes de certification indépendants fournissent des services d'évaluation de conformité et, dans certains secteurs, accèdent aux marques de conformité de l'IEC. L'IEC n'est responsable d'aucun des services effectués par les organismes de certification indépendants.
6) Tous les utilisateurs doivent s'assurer qu'ils sont en possession de la dernière édition de cette publication.
7) Aucune responsabilité ne doit être imputée à l'IEC, à ses administrateurs, employés, auxiliaires ou mandataires, y compris ses experts particuliers et les membres de ses comités d'études et des Comités nationaux de l'IEC, pour tout préjudice causé en cas de dommages corporels et matériels, ou de tout autre dommage de quelque nature que ce soit, directe ou indirecte, ou pour supporter les coûts (y compris les frais de justice) et les dépenses découlant de la publication ou de l'utilisation de cette Publication de l'IEC ou de toute autre Publication de l'IEC, ou au crédit qui lui est accordé.
8) L'attention est attirée sur les références normatives citées dans cette publication. L'utilisation de publications référencées est obligatoire pour une application correcte de la présente publication.
9) L'attention est attirée sur le fait que certains des éléments de la présente Publication de l'IEC peuvent faire l'objet de droits de brevet. L'IEC ne saurait être tenue pour responsable de ne pas avoir identifié de tels droits de brevets et de ne pas avoir signalé leur existence.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

L'IEC 62541-13 a été établie par le sous-comité 65E: Les dispositifs et leur intégration dans les systèmes de l'entreprise, du comité d'études 65 de l'IEC: Mesure, commande et automation dans les processus industriels.

Cette seconde édition annule et remplace la première édition de l'IEC 62541-13 parue en 2015. Cette édition constitue une révision technique.

Cette édition inclut les modifications techniques majeures suivantes par rapport à l'édition précédente:

a) aucune modification technique n'a été apportée, mais de nombreuses explications ont été introduites. Des corrections ont été apportées aux exemples.

Le texte de cette norme est issu des documents suivants:

FDIS	Rapport de vote
65E/697/FDIS	65E/712/RVD

Le rapport de vote indiqué dans le tableau ci-dessus donne toute information sur le vote ayant abouti à l'approbation de cette norme.

Cette publication a été rédigée selon les Directives ISO/IEC, Partie 2.

Dans l'ensemble du présent document et dans les autres parties de la série, certaines conventions de document sont utilisées:

le format *italique* est utilisé pour mettre en évidence un terme défini ou une définition qui apparaît à l'article "Termes et définitions" dans l'une des parties de la série;

le format *italique* est également utilisé pour mettre en évidence le nom d'un paramètre d'entrée ou de sortie de service, ou le nom d'une structure ou d'un élément de structure habituellement défini dans les tableaux;

par ailleurs, les *termes et les noms en italique* sont souvent écrits en camel-case (pratique qui consiste à joindre, sans espace, les éléments des mots ou expressions composés, la première lettre de chaque élément étant en majuscule). Par exemple, le terme défini est *AddressSpace* et non Espace d'Adressage. Cela permet de mieux comprendre qu'il existe une définition unique pour *AddressSpace*, et non deux définitions distinctes pour Espace et pour Adressage.

Une liste de toutes les parties de la série IEC 62541, publiées sous le titre général *Architecture unifiée OPC*, peut être consultée sur le site web de l'IEC.

Le comité a décidé que le contenu de cette publication ne sera pas modifié avant la date de stabilité indiquée sur le site web de l'IEC sous "http://webstore.iec.ch" dans les données relatives à la publication recherchée. A cette date, la publication sera

- reconduite,
- supprimée,
- remplacée par une édition révisée, ou
- amendée.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

IMPORTANT – Le logo "colour inside" qui se trouve sur la page de couverture de cette publication indique qu'elle contient des couleurs qui sont considérées comme utiles à une bonne compréhension de son contenu. Les utilisateurs devraient, par conséquent, imprimer cette publication en utilisant une imprimante couleur.

ARCHITECTURE UNIFIÉE OPC –

Partie 13: Agrégats

1 Domaine d'application

La présente partie de l'IEC 62541 fait partie de la série de spécifications générales sur l'Architecture unifiée OPC (OPC UA) globale. Elle définit le modèle d'information associé aux Agrégats.

2 Références normatives

Les documents suivants cités dans le texte constituent, pour tout ou partie de leur contenu, des exigences du présent document. Pour les références datées, seule l'édition citée s'applique. Pour les références non datées, la dernière édition du document de référence s'applique (y compris les éventuels amendements).

IEC TR 62541-1, *OPC Unified Architecture – Part 1: Overview and Concepts* (disponible en anglais seulement)

IEC 62541-3, *Architecture unifiée OPC – Partie 3: Modèle de l'Espace d'Adressage*

IEC 62541-4, *Architecture unifiée OPC – Partie 4: Services*

IEC 62541-5, *Architecture unifiée OPC – Partie 5: Modèle d'informations*

IEC 62541-8, *Architecture unifiée OPC – Partie 8: Accès aux données*

IEC 62541-11, *Architecture unifiée OPC – Partie 11: Accès à l'Historique*

3 Termes, définitions et termes abrégés

3.1 Termes et définitions

Pour les besoins du présent document, les termes et définitions donnés dans l'IEC TR 62541-1, l'IEC 62541-3, l'IEC 62541-4 et l'IEC 62541-11, ainsi que les suivants s'appliquent.

L'ISO et l'IEC tiennent à jour des bases de données terminologiques destinées à être utilisées en normalisation, consultables aux adresses suivantes:

- IEC Electropedia: disponible à l'adresse http://www.electropedia.org/
- ISO Online browsing platform: disponible à l'adresse http://www.iso.org/obp

3.1.1
ProcessingInterval
durée pendant laquelle les valeurs obtenues sont générées en fonction d'un *Agrégat* spécifié

Note 1 à l'article: Le domaine temporel total spécifié pour ReadProcessed est divisé par le *ProcessingInterval*. Par exemple, le calcul d'une moyenne de 10 min sur la plage de temps 12:00-12:30 donne lieu à un ensemble de trois intervalles de longueur *ProcessingInterval*, chaque intervalle commençant respectivement à 12:00, 12:10 et 12:20. Les règles permettant de déterminer les *Limites* de l'intervalle sont présentées en 5.4.2.2.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

3.1.2
données interpolées
données calculées à partir d'échantillons de données

Note 1 à l'article: Les échantillons de données peuvent être des données historiques ou des données en temps réel mises en mémoire tampon. Une valeur *interpolée* est calculée à partir des points de données d'un côté ou de l'autre de l'horodatage demandé.

3.1.3
EffectiveEndTime
heure précédant immédiatement *endTime*

Note 1 à l'article: Tous les calculs d'*Agrégat* incluent *startTime*, mais excluent *endTime*. Toutefois, il est parfois nécessaire de renvoyer une limite de fin *Interpolée* comme valeur d'un *Intervalle* avec un horodatage qui se trouve dans l'*Intervalle*. Les *Serveurs* sont supposés utiliser l'heure précédant immédiatement *endTime* si la résolution temporelle du *Serveur* détermine la valeur exacte (à ne pas confondre avec la résolution temporelle du matériel ou du système d'exploitation). Par exemple, si *endTime* est égal à 12:01:00 et que la résolution temporelle est de 1 s, *EffectiveEndTime* est égal à 12:00:59. Voir 5.4.2.4.

Si l'heure recule, les *Serveurs* sont supposés utiliser l'heure qui suit immédiatement *endTime*, la résolution temporelle du *Serveur* déterminant la valeur exacte.

3.1.4
données extrapolées
données construites à partir d'un ensemble de données discrètes, mais qui ne font pas partie de cet ensemble

Note 1 à l'article: S'apparente au processus d'interpolation, qui construit de nouveaux points entre des points connus, mais son résultat fait l'objet d'une plus grande incertitude. Les données *extrapolées* sont utilisées lorsque la période demandée est ultérieure aux données disponibles dans le système sous-jacent. Voir l'exemple présenté dans le Tableau 1.

3.1.5
SlopedInterpolation
interpolation linéaire simple

Note 1 à l'article: Comparer à l'ajustement de courbe à l'aide de polynômes linéaires. Voir l'exemple présenté dans le Tableau 1.

3.1.6
SteppedInterpolation
interpolation qui maintient le dernier point de données constant ou qui interpole la valeur à partir d'un ajustement de courbe horizontale

Note 1 à l'article: Se reporter au Tableau 1 suivant qui répertorie les valeurs brutes et *interpolées*/*extrapolées*.

Tableau 1 – Exemples d'interpolation

Horodatage	Valeur brute	Interpolation inclinée	Interpolation échelonnée
12:00:00	10		
12:00:05		15	10
12:00:08		18	10
12:00:10	20		
12:00:15		25	20
12:00:20	30		
		SlopedExtrapolation	SteppedExtrapolation
12:00:25		35	30
12:00:27		37	30

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

3.1.7
valeurs limites
valeurs en *startTime* et *endTime* nécessaires aux *Agrégats* pour calculer le résultat

Note 1 à l'article: Si les *Données brutes* n'existent pas en *startTime* et *endTime*, une valeur doit être estimée. Il existe deux moyens de déterminer les *Valeurs limites* d'un intervalle. L'une, dite des *Valeurs limites interpolées*, utilise les premiers points de données non "Bad" trouvés avant et après l'horodatage pour estimer la limite. L'autre, dite des *Valeurs limites simples*, utilise les points de données se trouvant immédiatement avant et après les horodatages limites pour estimer la limite, même si ces points sont "Bad". Les 3.1.8 et 3.1.9 décrivent les deux approches différentes plus en détail.

Dans tous les cas, le fanion *TreatUncertainAsBad* (voir 4.2.1.2) est utilisé pour déterminer si les valeurs incertaines sont "Bad" ou non "Bad".

Si une Valeur brute n'a pas été trouvée et qu'il existe une valeur limite non "Bad", la valeur "Interpolated" est attribuée aux bits d'*Agrégat* (voir 5.3.3).

Lors du calcul des *Valeurs limites*, une valeur nulle est attribuée à la partie des *Données brutes* dont le statut est "Bad". Cela signifie que la partie de valeur n'est pas utilisée dans le calcul, et qu'une valeur nulle est renvoyée si la valeur brute est renvoyée. La partie relative au statut est déterminée par les règles spécifiées par la limite ou l'*Agrégat*.

L'approche dite des *Valeurs limites interpolées* (voir 3.1.8) est identique à celle utilisée dans l'OPC HDA (Historical Data Access) classique, et est importante pour les applications telles que la commande de processus avancée, dans lesquelles il est important de détenir à tout moment des valeurs utiles. L'approche dite des *valeurs limites simples* (voir 3.1.9) est une approche nouvelle dans la présente norme. Elle est importante pour les applications qui doivent régulièrement générer des rapports et ne peuvent pas utiliser les valeurs estimées à la place des données "Bad".

3.1.8
valeurs limites interpolées
valeurs limites calculées à l'aide de la valeur "Good" la plus proche

Note 1 à l'article: Les *Valeurs limites interpolées* utilisant *SlopedInterpolation* sont calculées comme suit:

- s'il existe une valeur brute non "Bad" au niveau de l'horodatage, il s'agit de la valeur limite;
- rechercher la première valeur brute non "Bad" avant l'horodatage;
- rechercher la première valeur brute non "Bad" après l'horodatage;
- tracer une droite entre la valeur précédente et la valeur suivante;
- utiliser le point d'intersection des droites avec l'horodatage comme une estimation de la valeur limite.

Le calcul peut être exprimé par la formule suivante:

$$V_{\text{limite}} = (T_{\text{limite}} - T_{\text{précédente}}) \times (V_{\text{suivante}} - V_{\text{précédente}})/(T_{\text{suivante}} - T_{\text{précédente}}) + V_{\text{précédente}}$$

où V_x est une valeur en "x" et T_x est l'horodatage associé à V_x.

S'il existe une valeur non "Bad" avant l'horodatage, le *StatusCode* est Bad_NoData. Le *StatusCode* est *Uncertain_DataSubNormal* s'il existe une valeur "Bad" entre la valeur précédente et la valeur suivante. Si la valeur de l'une ou l'autre des valeurs précédente ou suivante est "Uncertain", le *StatusCode* est *Uncertain_DataSubNormal*. Si la valeur suivante n'existe pas, la valeur précédente doit être extrapolée à l'aide de *SlopedExtrapolation* ou de *SteppedExtrapolation*.

La période recherchée pour déterminer les valeurs "Good" avant et après l'horodatage dépend du *Serveur*, mais si une valeur "Good" est introuvable dans un intervalle de temps raisonnable, le *Serveur* prend pour hypothèse qu'elle n'existe pas. Il convient que le *Serveur* recherche au moins un intervalle de temps au moins égal à la taille de ProcessingInterval.

Les *Valeurs limites interpolées* utilisant SlopedExtrapolation sont calculées comme suit:

- rechercher la première valeur brute non "Bad" avant l'horodatage;
- rechercher la deuxième valeur brute non "Bad" avant l'horodatage;
- tracer une droite entre ces deux valeurs;
- étendre la droite jusqu'à ce qu'elle coupe l'horodatage;
- utiliser le point d'intersection de la droite avec l'horodatage comme une estimation de la valeur limite.

La formule est la même que celle utilisée pour *SlopedInterpolation*.

Le *StatusCode* est toujours *Uncertain_DataSubNormal*. Si une seule valeur brute non "Bad" peut être trouvée avant l'horodatage, *SteppedExtrapolation* est utilisé pour estimer la valeur limite.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Les *Valeurs limites interpolées* utilisant *SteppedInterpolation* sont calculées comme suit:

- s'il existe une valeur brute non "Bad" au niveau de l'horodatage, il s'agit de la valeur limite;
- rechercher la première valeur brute non "Bad" avant l'horodatage;
- utiliser la valeur comme une estimation de la valeur limite.

Le *StatusCode* est *Uncertain_DataSubNormal* s'il existe une valeur "Bad" entre la valeur précédente et l'horodatage. S'il n'existe aucune *Donnée brute* non "Bad" avant l'horodatage, le *StatusCode* est Bad_NoData. Si la valeur précédant l'horodatage est "Uncertain", le *StatusCode* est *Uncertain_DataSubNormal*. La valeur après l'horodatage n'est pas nécessaire lors de l'utilisation de *SteppedInterpolation*. Toutefois, si l'horodatage est après la fin des données, la valeur limite est traitée comme étant extrapolée, et le *StatusCode* est *Uncertain_DataSubNormal*.

SteppedExtrapolation est un terme qui décrit *SteppedInterpolation* lorsque l'horodatage est placé après la dernière valeur dans l'ensemble d'historiques.

3.1.9
valeurs limites simples
valeurs limites calculées à l'aide de la valeur la plus proche

Note 1 à l'article: Les *Valeurs limites simples* utilisant *SlopedInterpolation* sont calculées comme suit:

- s'il existe une valeur brute au niveau de l'horodatage, il s'agit de la valeur limite;
- rechercher la première valeur brute avant l'horodatage;
- rechercher la première valeur brute après l'horodatage;
- si la valeur qui suit l'horodatage est "Bad", la valeur précédente est la valeur limite;
- tracer une droite entre la valeur précédente et la valeur suivante;
- utiliser le point d'intersection des droites avec l'horodatage comme une estimation de la valeur limite.

La formule est la même que celle utilisée pour *SlopedInterpolation* en 3.1.5.

Si une valeur brute au niveau de l'horodatage est "Bad", le *StatusCode* est Bad_NoData. Si la valeur précédant l'horodatage est "Bad", le *StatusCode* est Bad_NoData. Si la valeur précédant l'horodatage est "Uncertain", le *StatusCode* est *Uncertain_DataSubNormal*. Si la valeur qui suit l'horodatage est "Bad" ou "Uncertain", le *StatusCode* est *Uncertain_DataSubNormal*.

Les *Valeurs limites simples* utilisant *SteppedInterpolation* sont calculées comme suit:

- s'il existe une valeur brute au niveau de l'horodatage, il s'agit de la valeur limite;
- rechercher la première valeur brute avant l'horodatage;
- si la valeur qui précède l'horodatage est non "Bad", il s'agit de la valeur limite.

Si une valeur brute au niveau de l'horodatage est "Bad", le *StatusCode* est Bad_NoData. Si la valeur précédant l'horodatage est "Bad", le *StatusCode* est Bad_NoData. Si la valeur précédant l'horodatage est "Uncertain", le *StatusCode* est *Uncertain_DataSubNormal*.

Si le temps limite d'un intervalle se situe au-delà du dernier point de données, le *Serveur* peut extrapoler ou renvoyer un message d'erreur. Si le serveur choisit l'extrapolation, le type d'extrapolation [*SteppedExtrapolation* ou *SloppedExtrapolation*] lui est spécifique.

Dans certains historiques, la dernière valeur brute n'indique pas nécessairement la fin des données. Selon les connaissances de l'historique en matière de mécanisme de collecte de données, c'est-à-dire la fréquence des mises à jour de données et la latence, il peut étendre la dernière valeur jusqu'à une période à couvrir connue. Lors du calcul des *Valeurs limites simples*, l'historique agit comme s'il y avait une autre valeur brute au niveau de cet horodatage.

De la même manière, si la première période d'un intervalle commence avant le premier point de données de l'historique et que la dernière période se situe après le premier point de données de l'historique, l'intervalle est traité comme s'il s'étendait entre le premier point de données de l'historique et la dernière période de l'intervalle, et le bit Partial est attribué au *StatusCode* de l'intervalle (voir 5.3.3.2).

La période recherchée pour déterminer les valeurs avant et après l'horodatage dépend du *Serveur*, mais si une valeur est introuvable dans un intervalle de temps raisonnable, le *Serveur* prend pour hypothèse qu'elle n'existe pas. Il convient que le *Serveur* recherche au moins un intervalle de temps au moins égal à la taille de ProcessingInterval.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

3.2 Termes abrégés

DA Data Access (Accès aux données)

HA Historical Access (Accès à l'historique) (accès aux données ou événements historiques)

HDA Historical Data Access (Accès aux données historiques)

UA Unified Architecture (Architecture unifiée)

4 Modèle d'information d'Agrégat

4.1 Généralités

Les normes IEC 62541-3 et IEC 62541-5 définissent la représentation des données historiques ou en temps réel mises en mémoire tampon d'*Agrégat* dans l'architecture unifiée OPC. Cela inclut la définition des *Agrégats* utilisés dans l'extraction des données traitées et l'extraction de l'historique. La présente définition inclut les types *Référence* et *Objet* normalisés.

4.2 Objets d'Agrégat

4.2.1 Généralités

4.2.1.1 Vue d'ensemble

Les *Serveurs* OPC UA peuvent prendre en charge plusieurs fonctionnalités et capacités différentes. Les *Objets* normalisés suivants permettent de présenter ces capacités en commun, plusieurs concepts définis normalisés pouvant être étendus par les fournisseurs.

4.2.1.2 AggregateConfigurationType

L'*AggregateConfigurationType* définit les caractéristiques générales d'un *Nœud* qui définit la configuration d'*Agrégat* d'une *Variable* ou *Propriété*. L'*Objet AggregateConfiguration* représente le point d'entrée de navigation des informations sur la manière dont le *Serveur* traite la fonctionnalité spécifique à l'*Agrégat*, comme le traitement des données "Uncertain". Il est défini de façon formelle dans le Tableau 2.

Tableau 2 – Définition d'AggregateConfigurationType

Attribut	**Valeur**				
BrowseName	AggregateConfigurationType				
IsAbstract	False				
Références	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Sous-type du *BaseObjectType* défini dans l'IEC 62541-5					
HasProperty	Variable	TreatUncertainAsBad	Boolean	PropertyType	Mandatory
HasProperty	Variable	PercentDataBad	Byte	PropertyType	Mandatory
HasProperty	Variable	PercentDataGood	Byte	PropertyType	Mandatory
HasProperty	Variable	UseSlopedExtrapolation	Boolean	PropertyType	Mandatory

La *Variable TreatUncertainAsBad* indique la manière dont le *Serveur* traite les données renvoyées avec une sévérité de *StatusCode* Uncertain par rapport aux calculs d'*Agrégat*. Les valeurs "True" et "False" indiquent respectivement que le *Serveur* estime la sévérité comme étant équivalente à Bad et Good, sauf indication contraire de la définition d'*Agrégat*. La valeur par défaut est "True". Noter que la valeur est toujours traitée comme "Uncertain" lorsque le *StatusCode* du résultat est calculé.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

La *Variable PercentDataBad* indique le pourcentage minimal de données "Bad" dans un intervalle donné, exigé pour le *StatusCode* afin d'attribuer la valeur Bad à l'intervalle donné de la demande de données traitée (Uncertain est traité comme indiqué ci-dessus). Voir 5.4.3 pour plus de détails sur l'utilisation de cette *Variable* lors de l'attribution de *StatusCodes*. Pour plus de détails sur les *Agrégats* qui utilisent la *Variable PercentDataBad*, voir la définition de chaque *Agrégat*. La valeur par défaut est 100.

La *Variable PercentDataGood* indique le pourcentage minimal de données "Good" dans un intervalle donné, exigé pour le *StatusCode* afin d'attribuer la valeur Good à l'intervalle donné de la demande de données traitée. Voir 5.4.3 pour plus de détails sur l'utilisation de cette *Variable* lors de l'attribution de *StatusCodes*. Pour plus de détails sur les *Agrégats* qui utilisent la *Variable PercentDataGood*, voir la définition de chaque *Agrégat*. La valeur par défaut est 100.

Le *PercentDataGood* et le *PercentDataBad* doivent suivre la relation *PercentDataGood* ≥ (100 – *PercentDataBad*). S'ils sont égaux, le résultat du calcul de *PercentDataGood* est utilisé. Si les valeurs affectées aux variables *PercentDataGood* et *PercentDataBad* ne permettent pas un calcul valide (comme Bad = 80; Good = 0) ou que ces valeurs sont supérieures à 100, le StatusCode du résultat est Bad_AggregateInvalidInputs.

La *Variable UseSlopedExtrapolation* indique la manière dont le *Serveur* interpole les données en l'absence de valeur limite (c'est-à-dire en extrapolant dans le futur à partir de la dernière valeur connue). La valeur "False" indique que le *Serveur* utilise un format *SteppedExtrapolation*, et maintient constante la dernière valeur connue. La valeur "True" indique que le *Serveur* projette la valeur en utilisant le mode *UseSlopedExtrapolation*. La valeur par défaut est False. Pour *SimpleBounds*, cette valeur n'est pas prise en compte.

4.2.2 Objet AggregateFunction

4.2.2.1 Généralités

Cet *Objet* est utilisé comme point d'entrée de navigation des informations relatives aux *Agrégats* pris en charge par un *Serveur*. Le contenu de cet *Objet* est déjà défini par sa définition de type. Toutes les *Instances* de *FolderType* utilisent le *BrowseName* normalisé "AggregateFunctions". La *Référence HasComponent* est utilisée pour associer un *Objet ServerCapabilities* et/ou un *Objet HistoryServerCapabilitiesType* à un *Objet AggregateFunction*. AggregateFunctions est défini de façon formelle dans le Tableau 3.

Tableau 3 – Définition des fonctions d'Agrégat

Attribut	**Valeur**				
BrowseName	AggregateFunctions				
Références	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Modelling Rule**
HasTypeDefinition	Object Type	*FolderType*	Défini dans l'IEC 62541-5		

Chaque *Objet ServerCapabilities* et *HistoryServerCapabilitiesType* doit faire référence à un *Objet AggregateFunction*. De plus, chaque *Objet HistoricalConfiguration* appartenant à un *HistoricalDataNode* peut faire référence à un *Objet AggregateFunction* utilisant la *Référence HasComponent*.

4.2.2.2 AggregateFunctionType

Cet *ObjectType* définit un *Agrégat* pris en charge par un *Serveur* UA. Cet *Objet* est défini de façon formelle dans le Tableau 4.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Tableau 4 – Définition d'AggregateFunctionType

Attribut	**Valeur**				
BrowseName	AggregateFunctionType				
IsAbstract	False				
Références	**Node Class**	**BrowseName**	**DataType**	**Type Definition**	**Modelling Rule**
Sous-type du *BaseObjectType* défini dans l'IEC 62541-5					

Pour l'*AggregateFunctionType*, l'*Attribut Description* (hérité de la *NodeClass Base*) est obligatoire. L'*Attribut Description* offre une description localisée de l'*Agrégat*.

Le Tableau 5 spécifie les *Attributs BrowseName* et *Description* pour les *Objets d'Agrégats* normalisés. La description est le texte "en" localisé. Pour les autres paramètres de lieu, elle doit être traduite.

Tableau 5 – Nœuds d'AggregateType normalisés

BrowseName	**Description**
	Agrégat d'interpolation
Interpolative	Au début de chaque intervalle, extraire la valeur calculée des points de données de part et d'autre de l'horodatage demandé.
Average	Extraire la valeur moyenne des données sur l'intervalle.
TimeAverage	Extraire les données moyennes pondérées dans le temps sur l'intervalle à l'aide des *Valeurs limites interpolées*.
TimeAverage2	Extraire les données moyennes pondérées dans le temps sur l'intervalle à l'aide des *Valeurs limites simples*.
Total	Extraire la totalité (intégrale par rapport au temps) des données sur l'intervalle à l'aide des *Valeurs limites interpolées*.
Total2	Extraire la totalité (intégrale par rapport au temps) des données sur l'intervalle à l'aide des *Valeurs limites simples*.
Minimum	Extraire la valeur brute minimale dans l'intervalle avec l'horodatage du début de l'intervalle.
Maximum	Extraire la valeur brute maximale dans l'intervalle avec l'horodatage du début de l'intervalle.
MinimumActualTime	Extraire la valeur minimale dans l'intervalle et l'horodatage de la valeur minimale.
MaximumActualTime	Extraire la valeur maximale dans l'intervalle et l'horodatage de la valeur maximale.
Range	Extraire la différence entre les valeurs minimale et maximale sur l'intervalle.
Minimum2	Extraire la valeur minimale dans l'intervalle en incluant les *Valeurs limites simples*.
Maximum2	Extraire la valeur maximale dans l'intervalle en incluant les *Valeurs limites simples*.
MinimumActualTime2	Extraire la valeur minimale avec l'horodatage réel en incluant les *Valeurs limites simples*.
MaximumActualTime2	Extraire la valeur maximale avec l'horodatage réel en incluant les *Valeurs limites simples*.
Range2	Extraire la différence entre les valeurs Minimum2 et Maximum2 sur l'intervalle.
Count	Extraire le nombre de valeurs brutes sur l'intervalle.
DurationInStateZero	Extraire la durée pendant laquelle une valeur booléenne ou numérique était à l'état nul à l'aide des *Valeurs limites simples*.
DurationInStateNonZero	Extraire la durée pendant laquelle une valeur booléenne ou numérique était à l'état non nul à l'aide des *Valeurs limites simples*.
NumberOfTransitions	Extraire le nombre de passages d'état nul à non nul d'une valeur booléenne ou numérique dans l'intervalle.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

BrowseName	Description
	Agrégat d'interpolation
Start	Extraire la valeur au début de l'intervalle.
End	Extraire la valeur à la fin de l'intervalle.
Delta	Extraire la différence entre les valeurs Start et End dans l'intervalle.
StartBound	Extraire la valeur au début de l'intervalle à l'aide des *Valeurs limites simples*.
EndBound	Extraire la valeur à la fin de l'intervalle à l'aide des *Valeurs limites simples*.
DeltaBounds	Extraire la différence entre les valeurs StartBound et EndBound dans l'intervalle à l'aide des *Valeurs limites simples*.
DurationGood	Extraire la durée totale dans l'intervalle pendant laquelle les données sont "Good".
DurationBad	Extraire la durée totale dans l'intervalle pendant laquelle les données sont "Bad".
PercentGood	Extraire le pourcentage de données (0 à 100) dans l'intervalle dont le *StatusCode* est "Good".
PercentBad	Extraire le pourcentage de données (0 à 100) dans l'intervalle dont le *StatusCode* est "Bad".
WorstQuality	Extraire le *StatusCode* des données le moins favorable dans l'intervalle.
WorstQuality2	Extraire le *StatusCode* des données le moins favorable en incluant les *Valeurs limites simples*.
AnnotationCount	Extraire le nombre d'*Annotations* dans l'intervalle (s'applique aux *Agrégats* historiques uniquement).
StandardDeviationSample	Extraire l'écart-type pour l'intervalle d'un échantillon de la population (n-1).
VarianceSample	Extraire la variance pour l'intervalle, calculée par StandardDeviationSample.
StandardDeviation Population	Extraire l'écart-type pour l'intervalle d'une population complète (n) qui inclut les *Valeurs limites simples*.
VariancePopulation	Extraire la variance pour l'intervalle, calculée par StandardDeviationPopulation, qui inclut les *Valeurs limites simples*.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

4.3 AggregateFilter MonitoredItem

4.3.1 Valeurs par défaut d'AggregateFilter MonitoredItem

Les valeurs par défaut utilisées pour les *Agrégats MonitoredItem* sont identiques à celles utilisées pour les *Agrégats* historiques. Elles sont définies en 4.2.1.2. Pour plus d'informations sur l'*AggregateFilter MonitoredItem*, voir l'IEC 62541-4.

4.3.2 Agrégats MonitoredItem et Valeurs limites

Lors du calcul des *Agrégats MonitoredItem* qui exigent l'utilisation de *Valeurs limites*, les limites peuvent ne pas être connues. Le calcul est réalisé de la même manière qu'une lecture d'historique avec le bit Partial. L'historique peut attendre un certain temps (pas plus d'un intervalle de traitement, en principe) avant de calculer l'intervalle à autoriser pour une latence de collecte de données et réduire l'utilisation du bit Partial.

Les données d'une lecture d'historique réalisée après la collecte de données et les données issues de *MonitoredItem* sur le même intervalle peuvent ne pas être identiques.

4.4 Fonctions et capacités d'exposition prises en charge

La Figure 1 montre une représentation possible des informations d'*Agrégat* dans l'*AddressSpace*. Dans cet exemple, même si le *Serveur* au plus haut niveau peut prendre en charge la fonctionnalité d'*Agrégat* pour Interpolative, Total, Average et autres, la *DataVariable* X prend uniquement en charge Interpolative, Total et Average, alors que la *DataVariable* Y prend en charge Average, un *Agrégat* défini par le fournisseur et d'autres *Agrégats* (non définis).

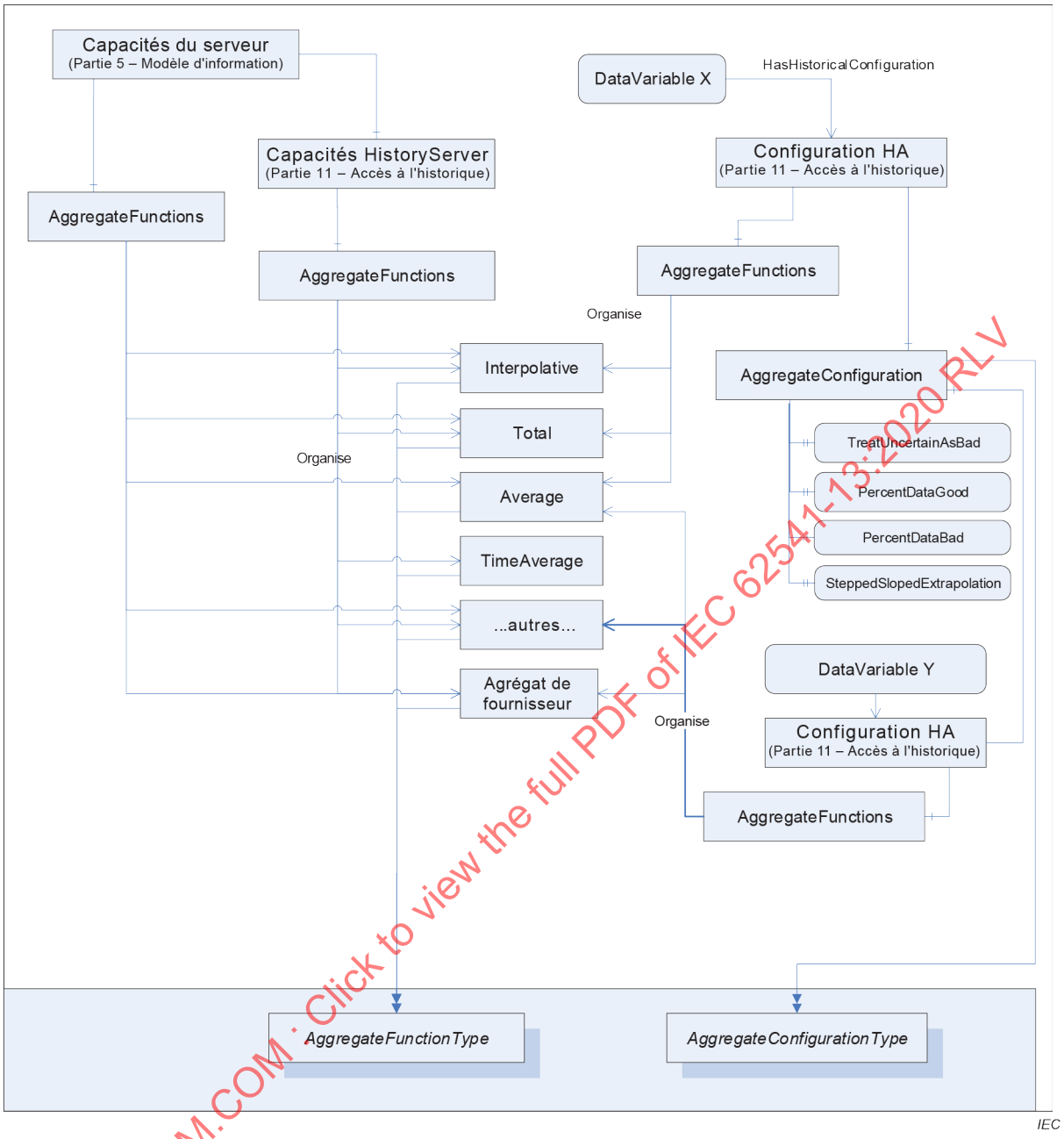


Figure 1 – Représentation des informations de configuration d'Agrégat dans l'AddressSpace

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5 Utilisation spécifique à l'Agrégat des Services

5.1 Généralités

L'IEC 62541-4 spécifie tous les Services nécessaires pour l'OPC UA *Aggregates*. En particulier:

- le Jeu de services "Browse" ou "Query" afin de détecter les *Agrégats* et leur configuration;
- le Service HistoryRead du Jeu de services "Attribute" pour lire l'historique agrégé des HistoricalNodes;
- le Service CreateMonitoredItems permet de spécifier un filtre pour chaque MonitoredItem afin de lire les données agrégées.

5.2 Traitement des données d'Agrégat

5.2.1 Vue d'ensemble

Le service *HistoryRead* défini dans l'IEC 62541-4 peut réaliser plusieurs fonctions différentes. Le paramètre *historyReadDetails* est un *Paramètre extensible* qui indique la fonction à réaliser. La structure *ReadProcessedDetails* est utilisée pour lire les données agrégées des *HistoricalDataNodes*.

Le *Service CreateMonitoredItems* permet de spécifier un filtre pour chaque *MonitoredItem*. Le *MonitoringFilter* est un paramètre extensible dont la structure dépend du type d'élément surveillé. La structure *AggregateFilter* est utilisée pour obtenir les données agrégées pour un abonnement.

5.2.2 Présentation de la structure ReadProcessedDetails

La structure ReadProcessedDetails est décrite de façon formelle dans l'IEC 62541-11. Le Tableau 6 présente les composants de la structure ReadProcessedDetails pour les besoins de la démonstration du présent document.

Tableau 6 – ReadProcessedDetails

Nom	Description
ReadProcessedDetails	Spécifie les détails utilisés pour réaliser une lecture d'historique “traitée”.
startTime	Début de la période de lecture.
endTime	Fin de la période de lecture.
processingInterval	Intervalle entre les valeurs d'*Agrégat* renvoyées.
aggregateType[]	*NodeIds* des *Objets AggregateFunction*. Les *Objets AggregateFunction* indiquent la liste des *Agrégats* à utiliser lors de l'extraction de l'historique traité.
aggregateConfiguration	Structure de configuration d'*Agrégat*.
useServerDefaults	Si la valeur est "True", les valeurs par défaut du *Serveur* sont utilisées, et toutes les valeurs spécifiées pour les autres paramètres ne sont pas prises en compte.
treatUncertainAsBad	Voir 4.2.1.2.
percentDataBad	Voir 4.2.1.2.
percentDataGood	Voir 4.2.1.2.
useSlopedExtrapolation	Voir 4.2.1.2.

5.2.3 Présentation de la structure AggregateFilter

L'*AggregateFilter* définit la fonction d'*Agrégat* qu'il convient d'utiliser pour calculer les valeurs à renvoyer. L'*AggregateFilter* est défini de façon formelle dans l'IEC 62541-4. Le Tableau 7 présente les composants de la structure *AggregateFilter* pour les besoins de la démonstration du présent document.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Tableau 7 – Structure AggregateFilter

Nom	Description
AggregateFilter	
startTime	Début de la période pour calculer l'*Agrégat* la première fois.
aggregateType	*NodeId* des *Objets AggregateFunction* qui indique la liste des *Agrégats* à utiliser pour récupérer les données traitées.
processingInterval	Période à utiliser pour calculer l'*Agrégat*.
aggregateConfiguration	Ce paramètre permet aux *Clients* d'ignorer les paramètres de configuration de l'*Agrégat* fournis par un *Objet AggregateConfiguration*, par élément surveillé.
useServerDefaults	Si la valeur est "True", les valeurs par défaut du *Serveur* sont utilisées, et toutes les valeurs spécifiées pour les autres paramètres ne sont pas prises en compte.
treatUncertainAsBad	Voir 4.2.1.2.
percentDataBad	Voir 4.2.1.2.
percentDataGood	Voir 4.2.1.2.
useSlopedExtrapolation	Voir 4.2.1.2.

5.3 StatusCodes des Agrégats

5.3.1 Vue d'ensemble

Le 5.3 définit les codes et règles supplémentaires qui s'appliquent au *StatusCode* utilisé pour les *Agrégats*.

La structure générale du *StatusCode* est spécifiée dans l'IEC 62541-4. Elle inclut un ensemble de codes de résultat opérationnel communs qui s'appliquent également aux *Agrégats*.

5.3.2 Codes de résultats de niveau opérationnel

Dans les *Agrégats* OPC UA Aggregates, le *StatusCode* permet d'indiquer les conditions dans lesquelles une valeur ou un *Evénement* a été stocké et, de ce fait, peut être utilisé comme un indicateur de validité. Compte tenu de la nature des données agrégées, il est nécessaire d'envoyer au client des informations supplémentaires, outre la qualité de base et le code de résultat d'appel. Par exemple, si le résultat a été *interpolé* ou non, si toutes les données saisies dans un calcul étaient de qualité "Good", etc.

Le Tableau 8 ci-dessous contient les codes avec une sévérité Bad indiquant une défaillance. Le Tableau 9 contient les codes avec la sévérité Uncertain indiquant que la valeur a été extraite dans des conditions inférieures à la normale. Il est important de noter qu'il s'agit de codes spécifiques aux *Agrégats* OPC UA Aggregates, qui viennent à l'appui de ceux qui s'appliquent à tous les types de données, et qui sont donc définis dans l'IEC 62541-4, l'IEC 62541-8 et l'IEC 62541-11.

Tableau 8 – Codes de résultats de niveau opérationnel "Bad"

ID symbolique	Description
Bad_AggregateListMismatch	Le nombre demandé d'*Agrégats* ne correspond pas au nombre demandé de *NodeIds*. Si plusieurs *Agrégats* sont demandés, un *NodeId* correspondant est exigé pour chaque *AggregateFunction.*
Bad_AggregateNotSupported	L'*AggregateFunction* demandée n'est pas prise en charge par le *Serveur* pour le *Nœud* spécifié.
Bad_AggregateInvalidInputs	La valeur d'*Agrégat* n'a pas pu être extraite en raison d'entrées de données non valides, d'erreurs lors de la tentative de conversion des données ou de situations analogues.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Tableau 9 – Codes de résultats de niveau opérationnel "Uncertain"

ID symbolique	Description
Uncertain_DataSubNormal	La valeur est obtenue à partir des valeurs brutes et contient un nombre de valeurs "Good" inférieur au nombre exigé.

5.3.3 Bits d'information d'Agrégat

5.3.3.1 Généralités

Ces bits sont uniquement définis lors de l'obtention des données d'*Agrégat*. Ils indiquent la provenance de la valeur de données et fournissent des informations qui ont un impact sur la manière dont le client utilise la valeur de données. Le Tableau 10 répertorie les paramétrages de bit qui indiquent l'emplacement des données (c'est-à-dire la valeur stockée dans le référentiel de données sous-jacent ou la valeur du résultat de l'agrégation des données). Ces bits s'excluent mutuellement.

Tableau 10 – Emplacement des données

StatusCode	Description
Raw	Valeur de *données brutes*.
Calculated	Valeur de données calculées.
Interpolated	Valeur de données interpolées.

Si des données *Interpolées* sont demandées, et qu'il existe une valeur brute réelle pour cet horodatage, il convient que le *Serveur* définisse le bit "Raw" dans le *StatusCode* de ladite valeur.

Le Tableau 11 répertorie les paramétrages de bit qui donnent des informations importantes supplémentaires relatives aux valeurs de données renvoyées.

Tableau 11 – Informations supplémentaires

StatusCode	Description
Partial	Valeur calculée ne reposant pas sur un intervalle complet. Voir 5.3.3.2.
Extra Data	Si un *Serveur* choisit de définir ce bit, il indique qu'une valeur de *Données brutes* remplace les autres données du même horodatage.
Multiple Values	Plusieurs valeurs correspondent aux critères d'*Agrégat* (c'est-à-dire plusieurs valeurs minimales ou plusieurs qualités inférieures au niveau d'horodatages différents du même *ProcessingInterval*).

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Les conditions dans lesquelles ces bits d'informations sont définis dépendent de la manière dont les données ont été demandées et de l'état du référentiel de données sous-jacent.

5.3.3.2 Bit d'information Partial

Le bit *Partial* est utilisé pour indiquer que l'intervalle n'est pas complet et qu'un client peut recevoir une valeur différente pour l'*Agrégat* s'il extrait de nouveau l'intervalle avec les mêmes paramètres.

Le bit *Partial* est défini dans les exemples ci-après.

Prendre pour hypothèse que, dans le cadre de ces exemples, le premier point et le dernier point stockés dans l'ensemble ont respectivement été stockés à 1:01:10 et 1:31:20. Des données plus anciennes peuvent exister, mais elles ne sont pas disponibles ou ne sont pas en ligne au moment de la requête. Des données plus récentes peuvent être disponibles, mais n'ont pas encore été stockées dans l'ensemble d'historiques.

- Intervalle qui chevauche le début de l'ensemble d'historiques. Si l'heure de début et de fin sont respectivement 1:00:00 et 1:10:00 et que l'intervalle est de 2 min, alors le premier intervalle renverra un bit Partial, puisqu'aucune donnée n'aura été spécifiée pour les 70 premières secondes. Le bit *Partial* est toujours défini pour le premier intervalle si l'heure de début de l'intervalle est antérieure à la première valeur de l'ensemble de données. Les intervalles antérieurs au bit Partial sont étiquetés Bad_NoData.
- Intervalle qui chevauche le dernier point stocké dans l'ensemble d'historiques. Le dernier point de l'ensemble est à 1:31:20, et l'historique n'a pas été arrêté et est toujours en cours. Un intervalle de 6 min qui a commencé à 1:30:00 prend le bit *Partial*, car l'historique attend les données, mais n'a encore rien reçu. Le bit *Partial* est toujours défini pour le dernier intervalle avec les données si l'heure de fin de l'intervalle est postérieure à la dernière valeur de données de l'ensemble de données. Les intervalles se trouvant en totalité après un bit *Partial* sont étiquetés Bad_NoData. Pour ces *Agrégats* avec extrapolation, le bit *Partial* peut être défini. Voir les caractéristiques spécifiques à l'*Agrégat* pour plus de détails.
- Si l'heure de début/de fin ne donne pas un intervalle homogène et que des données supplémentaires sont présentes au-delà de l'heure de fin, le dernier intervalle comporte un bit *Partial*. Si l'heure de début et de fin sont respectivement 1:00:00 et 1:20:00 et que l'intervalle est de 6 min, le dernier intervalle ne dure que 2 min et comporte donc un bit Partial. L'extrapolation ne s'applique pas dans ce cas.

Le bit *Partial* peut être défini avec le bit "Calculated" lorsque ce dernier est toujours défini pour l'*Agrégat* spécifique.

5.4 Détails de l'Agrégat

5.4.1 Généralités

Le 5.4 a pour but de décrire les exigences et le comportement des *Serveurs* OPC UA prenant en charge les *Agrégats*. L'objectif est de normaliser les *Agrégats* de sorte que les utilisateurs puissent correctement prévoir les résultats d'un calcul d'*Agrégat* et comprendre sa signification. Si les utilisateurs exigent de personnaliser la fonctionnalité dans les *Agrégats*, il convient d'écrire ces *Agrégats* comme *Agrégats* personnalisés définis par le fournisseur.

Les *Agrégats* normalisés doivent être aussi cohérents que possible, ce qui signifie que le comportement de chaque *Agrégat* doit être similaire à celui de tous les autres *Agrégats* dans lesquels les paramètres d'entrée, les *Données brutes* et les conditions limites sont analogues. Dans la mesure du possible, il convient que les *Agrégats* traitent de l'entrée et des conditions préalables de la même manière.

Le 5.4 est divisé en deux parties. Le 5.4.2 traite des caractéristiques et du comportement des *Agrégats* qui sont communs à tous les *Agrégats*. Le 5.4.3 traite des caractéristiques et du comportement spécifiques aux *Agrégats*.

5.4.2 Caractéristiques communes

5.4.2.1 Description

Le 5.4.2 traite des caractéristiques et du comportement des *Agrégats* qui sont communs à tous les *Agrégats*.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.2.2 Génération d'intervalles

Pour lire les *Agrégats* historiques, les clients OPC doivent spécifier trois paramètres de temps:

– startTime (Début)
– endTime (Fin)
– *ProcessingInterval* (Int)

Le *Serveur* OPC doit utiliser ces trois paramètres pour générer une séquence d'intervalles de temps et calculer un *Agrégat* pour chaque intervalle. Le 5.4.2.2 spécifie les intervalles de temps générés en fonction des trois paramètres. Le Tableau 12 donne des informations relatives aux intervalles pour chaque combinaison Heure de début – Heure de fin. La plage définie est [Fin – Début|.

Tous les *Agrégats* renvoient un horodatage du début de l'intervalle, sauf indication contraire pour l'*Agrégat* particulier.

Tableau 12 – Informations d'intervalle d'Agrégat historique

Heure de début/fin	Intervalle	Intervalles obtenus
Début = Fin	*Int* = toute valeur	Pas d'intervalle. Renvoie le *StatusCode* Bad_InvalidArgument, en présence ou non de données à l'heure spécifiée.
Début < Fin	*Int* = 0 ou *Int ≥ Plage*	Intervalle, commençant au *Début* et se terminant à la *Fin*. *Début* inclus, *Fin* exclue, c'est-à-dire *[Début, Fin)*.
Début < Fin	*Int ≠* 0, *Int < Plage*, *Int* divise *Plage* de manière égale.	Intervalles *Plage/Int*. Les intervalles sont *[Début, Début + Int), [Début + Int, Début + 2 × Int),..., [Fin – Int, Fin)*.
Début < Fin	*Int ≠* 0, *Int < Plage*, *Int* ne divise pas *Plage* de manière égale.	Intervalles ⌈*Plage/Int*⌉. Les intervalles sont *[Début, Début + Int), [Début + Int, Début + 2 × Int),..., [Début + (⌊Plage/Int⌋ – 1) × Int, Début + ⌊(Plage/Int⌋ × Int), [Début + ⌊Plage/Int⌋ × Int, Fin)*. En d'autres termes, le dernier intervalle contient le "reste" de la plage après avoir enlevé les intervalles ⌊*Plage/Int*⌋ de taille *Int*.
Début > Fin	*Int* = 0 ou *Int ≥ Plage*	Intervalle, commençant au *Début* et se terminant à la *Fin*. *Début* inclus, *Fin* exclue, c'est-à-dire [*Début, Fin*). [a]
Début > Fin	*Int ≠* 0, *Int < Plage*, *Int* divise *Plage* de manière égale.	Intervalles *Plage/Int*. Les intervalles sont [*Début, Début – Int), [Début – Int, Début – 2 × Int),..., [Fin + Int, Fin)*. [a]
Début > Fin	*Int ≠* 0, *Int < Plage*, *Int* ne divise pas *Plage* de manière égale.	Intervalles ⌈*Plage/Int*⌉. Les intervalles sont [*Début, Début – Int), [Début – Int, Début – 2 × Int),..., [Début – (⌊Plage/Int⌋ – 1) × Int, Début – ⌊Plage/Int⌋ × Int), [Début – ⌊Plage/Int⌋ × Int, Fin)*. En d'autres termes, le dernier intervalle contient le "reste" de la plage après avoir enlevé les intervalles ⌊*Plage/Int*⌋ de taille *Int* commençant au Début. [a]
[a] Dans ce cas, l'heure recule sur les intervalles.		

Le calcul de tous les *Agrégats* lorsque l'heure recule est identique au calcul lorsque l'heure avance, sauf que la "première heure" est exclue de l'intervalle et que la "dernière heure" est incluse. Dans la plupart des cas, cela signifie que la valeur est la même, sauf que les horodatages sont décalés d'un *ProcessingInterval*. Par exemple, lorsque l'heure avance, la valeur à l'instant $T = n$ est identique à celle de l'instant $T = n + 1$ lorsque l'heure recule.

Noter que, lors de la détermination des *Agrégats* avec *MonitoredItem*, l'intervalle est simplement le paramètre *ProcessingInterval* défini dans la structure *AggregateFilter*. Voir l'IEC 62541-4 pour plus de détails.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.2.3 Types de données

Le Tableau 13 présente le *DataType* valide pour chaque *Agrégat*. Certains *Agrégats* sont prévus pour des types de données numériques, c'est-à-dire des entiers ou des nombres réels/à virgule flottante. Les dates, chaînes, matrices, etc. ne sont pas prises en charge. Les autres *Agrégats* sont prévus pour des types de données numériques, c'est-à-dire des valeurs booléennes ou des énumérations. De plus, certains *Agrégats* peuvent renvoyer des résultats avec un DataType différent de ceux utilisés pour calculer l'*Agrégat*. Le Tableau 13 présente également le type de données renvoyé pour chaque *Agrégat*.

Tableau 13 – Informations normalisées de type de données d'Agrégat historique

BrowseName	Type de données valide	Type de données obtenu
	Agrégat d'interpolation	
Interpolative	Numérique	Type de données brutes
	Agrégats de moyennage de données	
Average	Numérique	Double
TimeAverage	Numérique	Double
TimeAverage2	Numérique	Double
Total	Numérique	Double
Total2	Numérique	Double
	Agrégats de variation des données	
Minimum	Numérique	Type de données brutes
Maximum	Numérique	Type de données brutes
MinimumActualTime	Numérique	Type de données brutes
MaximumActualTime	Numérique	Type de données brutes
Range	Numérique	Type de données brutes
Minimum2	Numérique	Type de données brutes
Maximum2	Numérique	Type de données brutes
MinimumActualTime2	Numérique	Type de données brutes
MaximumActualTime2	Numérique	Type de données brutes
Range2	Numérique	Type de données brutes
	Agrégats de comptage	
AnnotationCount	Tous	Entier
Count	Tous	Entier
DurationInStateZero	Numérique ou booléen	Durée
DurationInStateNonZero	Numérique ou booléen	Durée
NumberOfTransitions	Numérique ou booléen	Entier
	Agrégats de durée	
Start	Tous	Type de données brutes
End	Tous	Type de données brutes
Delta	Numérique	Type de données brutes
StartBound	Tous	Type de données brutes
EndBound	Tous	Type de données brutes
DeltaBounds	Numérique	Type de données brutes
	Agrégats de qualité des données	

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

BrowseName	Type de données valide	Type de données obtenu
DurationGood	Tous	Durée
DurationBad	Tous	Durée
PercentGood	Tous	Double
PercentBad	Tous	Double
WorstQuality	Tous	*StatusCode*
WorstQuality2	Tous	*StatusCode*
	Agrégats statistiques	
StandardDeviationSample	Numérique	Double
VarianceSample	Numérique	Double
StandardDeviationPopulation	Numérique	Double
VariancePopulation	Numérique	Double

5.4.2.4 Questions liées au calcul du temps

Les questions suivantes peuvent être soulevées lors du calcul d'*Agrégats* dont une partie du calcul inclut le temps.

- Tous les calculs d'*Agrégat* incluent *startTime*, mais excluent *endTime*. Toutefois, il est parfois nécessaire de renvoyer une Limite de fin *Interpolée* comme valeur d'un *Intervalle* avec un horodatage qui se trouve dans l'*Intervalle*. Les *Serveurs* sont supposés utiliser l'heure précédant immédiatement *endTime* si la résolution temporelle du *Serveur* détermine la valeur exacte (à ne pas confondre avec la résolution temporelle du matériel ou du système d'exploitation). Par exemple, si *endTime* est égal à 12:01:00 et que la résolution temporelle est de 1 s, *EffectiveEndTime* est égal à 12:00:59. Si la résolution temporelle du *Serveur* est de 1 ms, *EffectiveEndTime* est de 12:00:59.999.

  Si l'heure recule, les *Serveurs* sont supposés utiliser l'heure qui suit immédiatement *endTime*, la résolution temporelle du *Serveur* déterminant la valeur exacte.

- Si l'*Intervalle* contient un point de données et qu'il est égal à *StartTime*, la durée utilisée dans les calculs est une unité de la résolution temporelle du *Serveur*.

5.4.3 Traitement des données agrégées spécifiques

5.4.3.1 Généralités

Lors de l'accès aux données agrégées à l'aide du *service HistoryRead* ou *CreateMonitoredItems*, les règles suivantes sont utilisées pour traiter les cas d'utilisation d'*Agrégat* spécifiques.

Si *ProcessingInterval* est égal à 0, le *Serveur* doit créer une valeur d'*Agrégat* pour l'ensemble de l'intervalle de temps. Cela permet des *Agrégats* sur de longues périodes. Une valeur dont l'horodatage est égal à *endTime* est exclue de cet *Agrégat*, comme elle l'est d'un intervalle avec cette heure de fin. Si le *ProcessingInterval* dont la valeur est nulle est transmis dans le *MonitoredItemFilter*, une valeur différente de zéro adaptée doit lui être attribuée.

L'horodatage renvoyé avec l'*Agrégat* doit être l'heure du début de l'intervalle, sauf lorsque l'*Agrégat* spécifie un horodatage différent.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Si une autre valeur que l'horodatage source est définie pour l'horodatage demandé, l'opération doit renvoyer le *StatusCode Bad_TimestampToReturnInvalid*. Si un horodatage demandé n'est pas pris en charge d'une autre manière pour un *HistoricalDataNode*, l'opération doit renvoyer le *StatusCode Bad_TimestampNotSupported*. Pour *MonitoredItem*, le *Serveur* ne doit pas renvoyer les données antérieures si un horodatage demandé n'est pas pris en charge par l'ensemble d'historiques.

5.4.3.2 Calcul de StatusCode

5.4.3.2.1 Généralités

Les *StatusCodes* d'une valeur d'*Agrégat* doivent tenir compte des valeurs utilisées pour les calculer. De plus, les paramètres de configuration *PercentDataGood* et *PercentDataBad* permettent au client de contrôler la manière de procéder au calcul, si cela est pris en charge par le *Serveur*.

Si un *Agrégat* repose sur des valeurs brutes (Average, par exemple), le calcul est réalisé avec des valeurs de comptage. Si un *Agrégat* repose sur des valeurs brutes, mais qu'il peut également renvoyer une *Valeur limite*, les *Valeurs limites* sont incluses dans le comptage lors du calcul du *StatusCode*. Si un *Agrégat* procède à un calcul pondéré dans le temps (TimeAverage ou TimeAverage2, par exemple), le calcul du *StatusCode* doit également être pondéré dans le temps.

Pour les besoins du calcul des *StatusCodes* pondérés dans le temps, chaque intervalle doit être divisé en zones de données "Good" et "Bad". La création de ces zones exige de calculer les *Valeurs limites* pour chaque intervalle, le type de valeur limite dépendant de l'*Agrégat*.

Si *TreatUncertainAsBad* = False, des zones "Uncertain" sont incluses avec les zones "Good" lors du calcul des rapports ci-dessus. Si *TreatUncertainAsBad* = True, les zones "Uncertain" sont incluses dans les zones "Bad". Le *StatusCode* de la valeur est toujours traité comme "Uncertain" lorsque le *StatusCode* du résultat est calculé. Si aucune zone "Bad" ne se trouve dans l'intervalle, le *StatusCode* de l'intervalle est "Good". Pour tous les intervalles contenant des zones dont les *StatusCodes* sont "Bad", la durée totale de toutes les zones *Bad* est calculée, puis divisée par la largeur de l'intervalle. Le rapport obtenu est multiplié par 100 et comparé au paramètre *PercentDataBad*. Le *StatusCode* de l'intervalle est "Bad" si le rapport est supérieur ou égal au paramètre *PercentDataBad*. Pour tous les intervalles qui ne sont pas "Bad", la durée totale de toutes les zones "Good" est calculée, puis divisée par la largeur de l'intervalle. Le rapport obtenu est multiplié par 100 et comparé au paramètre *PercentDataGood*. Le *StatusCode* de l'intervalle est "Good" si le rapport est supérieur ou égal au paramètre *PercentDataGood*. Si, pour un intervalle, aucun rapport ne s'applique, cet intervalle est *Uncertain_DataSubNormal*.

Si l'intervalle ne contient pas de données, qu'il se trouve à l'intérieur de la plage [StartOfData, EndOfData] et que l'*Agrégat* renvoie un type de données brutes, le *StatusCode* de l'intervalle est Bad_NoData, sauf si les caractéristiques spécifiques à l'*Agrégat* en décident autrement.

La largeur d'un intervalle est le *ProcessingInterval,* sauf s'il s'agit d'un intervalle partiel (c'est-à-dire qu'une valeur est attribuée au bit Partial). Dans ce cas, la largeur est la durée utilisée lors du calcul de l'intervalle partiel.

Les 5.4.3.2.2 et 5.4.3.2.3 incluent des diagrammes qui représentent une série de demandes et de données. La couleur de l'axe temporel indique le statut des différentes zones. Le rouge indique "Bad", le vert "Good" et l'orange "Uncertain". Ces exemples prennent pour hypothèse que *TreatUncertainAsBad* = False.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.2.2 Interpolation inclinée et Valeurs limites simples

La Figure 2 suivante représente une série de données pour la *Variable* avec Stepped = False et un *Agrégat* qui utilise des *Valeurs limites simples*. L'Heure de début de la demande traitée est antérieure au premier point de la série, et son Heure de fin n'est pas un entier multiple du *ProcessingInterval*.

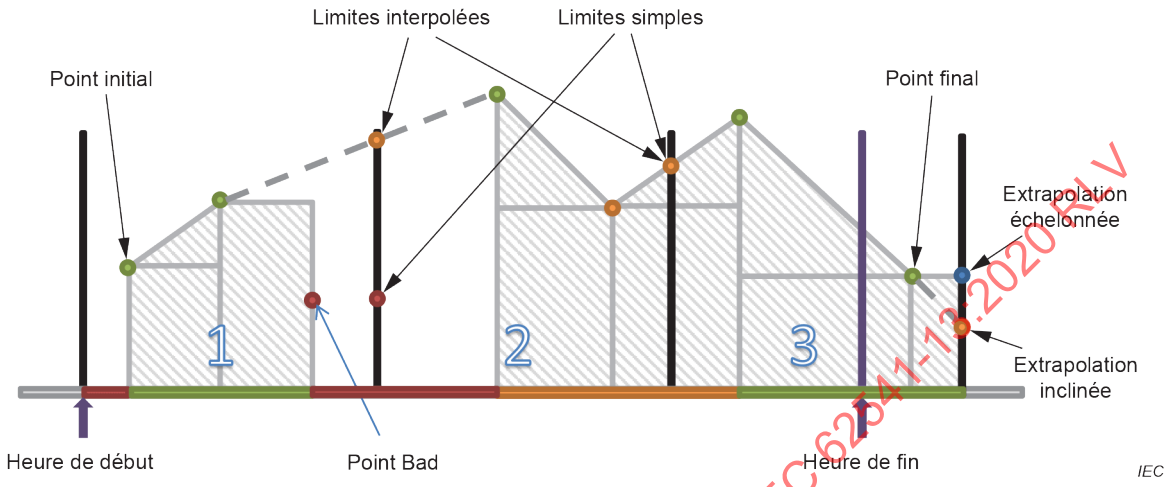


Figure 2 – Variable avec Stepped = False et Valeurs limites simples

Le premier intervalle contient quatre zones:

- la période qui précède le premier point de données;
- la période entre le premier et le deuxième point, où *SlopedInterpolation* peut être utilisé;
- la période entre le deuxième et le troisième point, où *SteppedInterpolation* est utilisé;
- la période qui suit le point "Bad", où il n'existe aucune donnée.

Une zone est "Uncertain" si elle se termine dans une valeur "Bad" ou "Uncertain" et que *SlopedInterpolation* est utilisé. Le point final n'a aucun impact sur la zone si *SteppedInterpolation* est utilisé.

Le deuxième intervalle contient trois zones:

- la période qui précède le point de données "Good", où il n'existe aucune donnée;
- la période entre le premier et le deuxième point, où *SlopedInterpolation* peut être utilisé;
- la période entre le deuxième point et la limite calculée avec *SlopedInterpolation*.

Le troisième intervalle contient trois zones:

- la période entre la limite simple et le premier point de données;
- la période entre le premier point et un point interpolé qui tombe sur l'heure de fin;
- la période qui suit l'heure de fin, qui n'est pas prise en compte.

Il s'agit d'une zone partielle, les données après l'heure de fin n'étant pas utilisées. Toutefois, si l'interpolation inclinée est utilisée et que le point qui suit le point d'extrémité est "Uncertain", la zone entre le dernier point et l'heure de fin est "Uncertain".

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.2.3 Interpolation échelonnée et Valeurs limites interpolées

La Figure 3 représente une série de données pour la *Variable* avec Stepped = True et un *Agrégat* qui utilise des *Valeurs limites interpolées*. L'Heure de début de la demande traitée est antérieure au premier point de la série, et son Heure de fin n'est pas un entier multiple du *ProcessingInterval*.

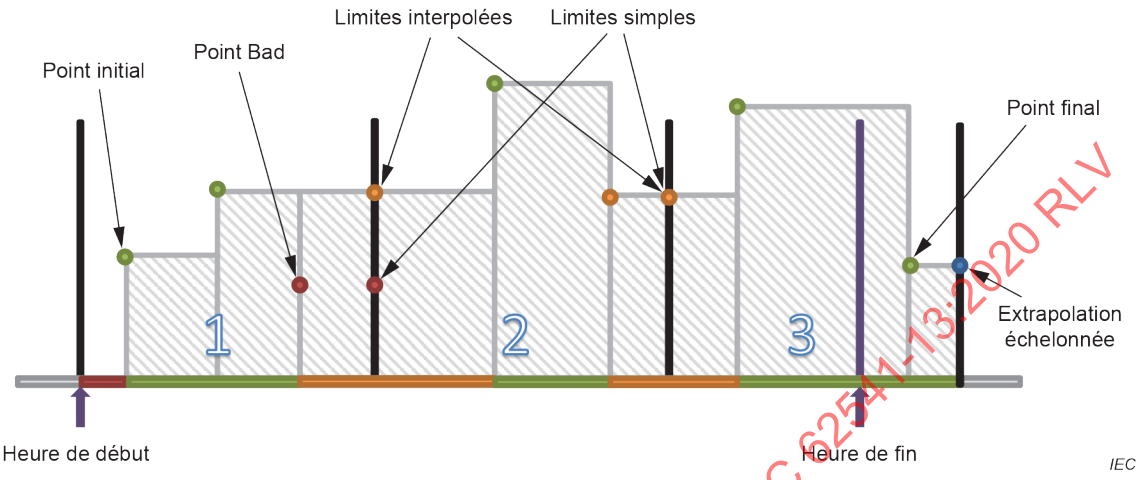


Figure 3 – Variable avec Stepped = True et Valeurs limites interpolées

Le premier intervalle contient trois zones:

- la période qui précède le premier point de données;
- la période entre le premier et le deuxième point, où *SteppedInterpolation* est utilisé;
- la période entre le deuxième point et la limite de fin interpolée.

Le point "Bad" n'est pas pris en compte en raison de la limite de fin interpolée, mais cela crée des zones "Uncertain". Si *SlopedInterpolation* a été utilisé, la zone "Uncertain" commence au deuxième point. Dans ce cas, elle commence uniquement lorsque la première valeur "Bad" n'est pas prise en compte.

Le deuxième intervalle contient trois zones:

- la période entre la limite de début et le premier point de données;
- la période entre le premier et le deuxième point, où *SteppedInterpolation* est utilisé;
- la période entre le deuxième point et la limite de fin interpolée.

Le troisième intervalle contient trois zones:

- la période entre la limite interpolée et le premier point de données;
- la période entre le premier point et un point interpolé qui tombe sur l'heure de fin;
- la période qui suit l'heure de fin, qui n'est pas prise en compte.

Il s'agit d'une zone partielle, et les données qui suivent l'heure de fin ne sont pas utilisées.

5.4.3.3 Description

Le 5.4.3.3 traite des caractéristiques spécifiques à l'*Agrégat* et du comportement spécifique à un *Agrégat* particulier.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Chaque paragraphe comporte un tableau qui exprime de façon formelle le comportement de l'*Agrégat* (y compris les exceptions). La signification de chacune des zones du tableau est décrite dans le Tableau 14.

Description du Tableau 14:

- la première colonne est le nom commun de l'élément;
- la deuxième colonne est une description de l'élément et une liste de choix valides pour l'élément, comprenant une description de chaque choix;
- la deuxième partie du tableau décrit la manière dont le statut associé au calcul de l'*Agrégat* est déterminé;
- la dernière partie du tableau précise le comportement prévu de l'*Agrégat* dans certains cas particuliers communs. Ces comportements exigeant des descriptions textuelles, cette partie ne contient pas de liste de choix valides.

Tableau 14 – Description du tableau d'Agrégat

Caractéristiques de l'Agrégat	
Type	Type d'*Agrégat*. <Interpolated | Calculated | Raw> Interpolated: Voir la définition d'Interpolé. Calculated: Déterminé par un calcul défini. Raw: Choisit une valeur brute dans un intervalle.
Type de données	Type de données du résultat. <Double | Int32 | Same as Source>
Limites d'utilisation	Manière dont l'*Agrégat* traite les limites. <None | Interpolated | Simple> None: Les limites ne s'appliquent pas à l'*Agrégat*. Interpolated: Utilise les limites interpolées. Simple: Utilise les limites simples.
Horodatage	Horodatage de la valeur d'*Agrégat* obtenue: <*startTime* | *endTime* | Raw> *startTime*: Heure de début de l'intervalle. *endTime*: Heure de fin de l'intervalle. Raw: Heure associée à une valeur dans l'intervalle.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Caractéristiques de l'Agrégat	
Calcul des StatusCodes	
Méthode de calcul	Manière de calculer le code de statut: <PercentValues | PercentTime | Custom> PercentValues: En fonction du pourcentage des nombres de valeurs. PercentTime: En fonction du pourcentage d'intervalle de temps. Custom: Spécifique à l'*Agrégat* (description incluse).
Bit partiel	Pour les intervalles partiels, si l'*Agrégat* définit ce bit. <Set Sometimes | Not Set> Il peut également décrire des cas particuliers pour la définition de ce bit.
Bit calculé	Décrit l'utilisation du bit calculé. <Set Always | Set Sometimes | Not Set> Set Always: Le bit est toujours défini. Set Sometimes: Le bit est parfois défini (décrire quand). Not Set: Le bit n'est jamais défini.
Bit interpolé	Décrit l'utilisation du bit interpolé. <Set Always | Set Sometimes | Not Set> Set Always: Le bit est toujours défini. Set Sometimes: Le bit est parfois défini (décrire quand). Not Set: Le bit n'est jamais défini.
Bit brut	Décrit l'utilisation du bit brut. <Set Always | Set Sometimes | Not Set> Set Always: Le bit est toujours défini. Set Sometimes: Le bit est parfois défini (décrire quand). Not Set: Le bit n'est jamais défini.
Bit multivaleur	Décrit l'utilisation du bit à plusieurs valeurs. <Set Sometimes | Not Set> Set Sometimes: Le bit est utilisé (voir l'IEC 62541-11) Not Set: Le bit n'est jamais défini.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Caractéristiques de l'Agrégat	
Cas spéciaux communs du StatusCode	
Avant le début des données	Si l'ensemble de l'intervalle précède le début des données.
Après la fin des données	Si l'ensemble de l'intervalle suit la fin des données (comme déterminé par l'Historique).
Limite de début introuvable	Si la limite de début est introuvable pour le tout premier intervalle et que ce dernier n'est pas partiel, détermine le traitement particulier qu'il convient d'appliquer, le cas échéant.
Limite de fin introuvable	Si la limite de fin est introuvable pour le dernier intervalle, lequel n'est pas partiel, détermine le traitement particulier qu'il convient d'appliquer, le cas échéant.
Limite "Bad"	Si la valeur limite est "Bad", elle détermine le traitement particulier qu'il convient d'appliquer, le cas échéant.
Limite "Uncertain"	Si la valeur limite est "Uncertain", elle détermine le traitement particulier qu'il convient d'appliquer, le cas échéant.

5.4.3.4 Interpolative

L'*Agrégat* "Interpolative" défini dans le Tableau 15 renvoie la *Valeur limite interpolée* (voir 3.1.8) pour la *startTime* de chaque intervalle.

Lors de la recherche des valeurs "Good" avant ou après la valeur limite, la période recherchée est spécifique au *Serveur*, mais il convient que le *Serveur* recherche un intervalle de temps dont la taille est au moins égale au *ProcessingInterval*.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Tableau 15 – Récapitulatif de l'Agrégat "Interpolative"

Caractéristiques de l'Agrégat interpolé	
Type	Interpolated
Type de données	Same as Source
Limites d'utilisation	Interpolated
Horodatage	StartTime
Calcul des StatusCodes	
Méthode de calcul	Custom "Good" si aucune valeur "Bad" n'est prise en compte et si des valeurs "Good" sont utilisées, "Uncertain" si les valeurs "Bad" ne sont pas prises en compte ou si des valeurs "Uncertain" sont utilisées. En l'absence de valeur de début, *Bad_NoData*. Voir la description des Limites interpolées (voir 3.1.8) pour plus de détails
Bit partiel	Not Set
Bit calculé	Not Set
Bit interpolé	Set Sometimes Toujours défini, sauf lorsque le bit brut est défini
Bit brut	Set Sometimes Si une valeur existe avec l'heure exacte du début d'intervalle
Bit multivaleur	Not Set
Cas spéciaux communs du StatusCode	
Avant le début des données	Renvoie *Bad_NoData*
Après la fin des données	Renvoie une valeur extrapolée (voir 3.1.8) (inclinée ou échelonnée en fonction des paramètres) Le code de statut est *Uncertain_DataSubNormal*
Limite de début introuvable	*Bad_NoData*
Limite de fin introuvable	*Voir "Après la fin des données"*
Limite "Bad"	Ne renvoie pas une limite "Bad", sauf comme indiqué ci-dessus
Limite "Uncertain"	Renvoie *Uncertain_DataSubNormal* si une ou plusieurs valeurs "Bad" n'ont pas été prises en compte dans le calcul de la valeur limite

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.5 Average

L'*Agrégat* "Average" défini dans le Tableau 16 ajoute les valeurs de toutes les *Données brutes* "Good" pour chaque intervalle, et divise la somme par le nombre de valeurs "Good". Si les valeurs autres que "Good" ne sont pas prises en compte dans le calcul, le *StatusCode* de l'*Agrégat* est déterminé par calcul (voir 5.3). Il ne s'agit pas d'un *Agrégat* fondé sur le temps. Par conséquent, PercentGood/PercentBad s'applique au nombre de valeurs dans l'intervalle.

Tableau 16 – Récapitulatif de l'Agrégat "Average"

Caractéristiques de l'Agrégat "Average"	
Type	Calculated
Type de données	Double
Limites d'utilisation	None
Horodatage	StartTime
Calcul des StatusCodes	
Méthode de calcul	PercentValues
Bit partiel	Not Set
Bit calculé	Set Always
Bit interpolé	Not Set
Bit brut	Not Set
Bit multivaleur	Not Set
Cas spéciaux communs du StatusCode	
Avant le début des données	Bad_NoData
Après la fin des données	Bad_NoData
Limite de début introuvable	Limites non utilisées
Limite de fin introuvable	Limites non utilisées
Limite "Bad"	Limites non utilisées
Limite "Uncertain"	Limites non utilisées

5.4.3.6 TimeAverage

L'*Agrégat* "TimeAverage" défini dans le Tableau 17 utilise les *Valeurs limites interpolées* (voir 3.1.8) pour déterminer la valeur d'un point au début et à la fin d'un intervalle. Une ligne droite est tracée entre chaque valeur dans l'*intervalle*, entre la valeur limite de début et la valeur limite de fin (voir les exemples). La zone sous les lignes est divisée par la longueur du *ProcessingInterval* pour donner la moyenne. Noter que ce calcul utilise toujours une ligne inclinée entre les points. TimeAverage2 utilise une ligne échelonnée ou inclinée, en fonction de la valeur de la *Propriété* Echelonnée de la *Variable*.

Si l'intervalle contient une ou plusieurs valeurs "Bad", elles ne sont pas prises en compte dans le calcul et la valeur *Uncertain_DataSubNormal* est attribuée au *StatusCode*. Les lignes inclinées sont tracées entre les valeurs "Good" lors du calcul de la zone.

La résolution temporelle utilisée dans ce calcul est spécifique au *Serveur*.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Tableau 17 – Récapitulatif de l'Agrégat "TimeAverage"

Caractéristiques de l'Agrégat "TimeAverage"	
Type	Calculated
Type de données	Double
Limites d'utilisation	Interpolated
Horodatage	StartTime
Calcul des StatusCodes	
Méthode de calcul	Custom "Good" si aucune valeur "Bad" n'est prise en compte et si des valeurs "Good" sont utilisées, "Uncertain" si les valeurs "Bad" ne sont pas prises en compte ou si des valeurs "Uncertain" sont utilisées.
Bit partiel	Set Sometimes Si l'intervalle n'est pas complet
Bit calculé	Set Always
Bit interpolé	Not Set
Bit brut	Not Set
Bit multivaleur	Not Set
Cas spéciaux communs du StatusCode	
Avant le début des données	Bad_NoData
Après la fin des données	Valeur extrapolée, statut "Uncertain"
Limite de début introuvable	Calculer l'intervalle partiel
Limite de fin introuvable	Extrapoler les données, statut "Uncertain"
Limite "Bad"	Non applicable
Limite "Uncertain"	Non applicable

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.7 TimeAverage2

L'*Agrégat* "TimeAverage2" défini dans le Tableau 18 utilise les *Valeurs limites simples* (voir 3.1.9) pour déterminer la valeur d'un point au début et à la fin d'un intervalle. Une ligne droite est tracée entre chaque valeur dans l'intervalle, entre la valeur limite de début et la valeur limite de fin (voir les exemples). La zone sous les lignes est divisée par la longueur du *ProcessingInterval* pour donner la moyenne. Noter que ce calcul utilise une ligne échelonnée ou inclinée, en fonction de la valeur de la *Propriété* Echelonnée de la *Variable*. TimeAverage utilise toujours une ligne inclinée entre les points.

La résolution temporelle utilisée dans ce calcul est spécifique au *Serveur*.

Si l'intervalle contient des données autres que "Good", elles ne sont pas prises en compte pour le calcul et l'intervalle de temps est réduit d'une durée égale à celle de ces mêmes données, c'est-à-dire que si une valeur est "Bad" pendant 1 min dans un intervalle de 5 min, TimeAverage2 est représenté par la zone sous la période de 4 min des valeurs "Good", divisée par 4 min. Si un sous-intervalle se termine par une valeur "Bad", seule la valeur de départ "Good" est utilisée pour calculer la zone du sous-intervalle qui précède la valeur "Bad".

Le *StatusCode* de l'*Agrégat* est déterminé par calcul (voir 5.3).

Tableau 18 – Récapitulatif de l'Agrégat "TimeAverage2"

Caractéristiques de l'Agrégat "TimeAverage2"	
Type	Calculated
Type de données	Double
Limites d'utilisation	Simple
Horodatage	StartTime
Calcul des StatusCodes	
Méthode de calcul	PercentTime
Bit partiel	Set Sometimes Si l'intervalle n'est pas complet
Bit calculé	Set Always
Bit interpolé	Not Set
Bit brut	Not Set
Bit multivaleur	Not Set
Cas spéciaux communs du StatusCode	
Avant le début des données	Bad_NoData
Après la fin des données	Bad_NoData
Limite de début introuvable	La limite est Bad_NoData et est traitée comme une autre valeur "Bad" dans l'intervalle
Limite de fin introuvable	La limite est Bad_NoData et est traitée comme une autre valeur "Bad" dans l'intervalle
Limite "Bad"	Traitée comme une autre valeur "Bad" dans l'intervalle
Limite "Uncertain"	Traitée comme une autre valeur "Uncertain" dans l'intervalle

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.8 Total

L'*Agrégat* "Total" défini dans le Tableau 19 permet de procéder au calcul suivant pour chaque intervalle:

Total = TimeAverage × *ProcessingInterval* (secondes)

où: TimeAverage est le résultat de l'*Agrégat* "TimeAverage", utilisant le *ProcessingInterval* fourni pour l'appel Total.

Les unités obtenues sont normalisées en secondes, c'est-à-dire [TimeAverage Units] × secondes.

Le *StatusCode* de l'*Agrégat* est déterminé par calcul (voir 5.3).

Tableau 19 – Récapitulatif de l'Agrégat "Total"

Caractéristiques de l'Agrégat "Total"	
Type	Calculated
Type de données	Double
Limites d'utilisation	Interpolated
Horodatage	StartTime
Calcul des StatusCodes	
Méthode de calcul	Custom "Good" si aucune valeur "Bad" n'est prise en compte et si des valeurs "Good" sont utilisées, "Uncertain" si les valeurs "Bad" ne sont pas prises en compte ou si des valeurs "Uncertain" sont utilisées.
Bit partiel	Set Sometimes Si l'intervalle n'est pas complet
Bit calculé	Set Always
Bit interpolé	Not Set
Bit brut	Not Set
Bit multivaleur	Not Set
Cas spéciaux communs du StatusCode	
Avant le début des données	Bad_NoData
Après la fin des données	Valeur extrapolée, statut "Uncertain"
Limite de début introuvable	Calculer l'intervalle partiel
Limite de fin introuvable	Extrapoler les données, statut "Uncertain"
Limite "Bad"	Non applicable
Limite "Uncertain"	Non applicable

5.4.3.9 Total2

L'*Agrégat* "Total2" défini dans le Tableau 20 permet de procéder au calcul suivant pour chaque intervalle:

Total2 = TimeAverage2 × *ProcessingInterval* des données "Good" (secondes)

où TimeAverage2 est le résultat de l'*Agrégat* "TimeAverage2", utilisant le *ProcessingInterval* fourni pour l'appel Total2.

L'intervalle des données "Good" est la somme de tous les sous-intervalles contenant des données autres que "Bad", c'est-à-dire que si une valeur est "Bad" pendant 1 min dans un intervalle de 5 min, l'intervalle des données "Good" s'étend sur une période de 4 min.

Les unités obtenues sont normalisées en secondes, c'est-à-dire [TimeAverage2 Unités] × secondes.

Le *StatusCode* de l'*Agrégat* est déterminé par calcul (voir 5.3).

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Tableau 20 – Récapitulatif de l'Agrégat "Total2"

Caractéristiques de l'Agrégat "Total2"	
Type	Calculated
Type de données	Double
Limites d'utilisation	Simple
Horodatage	StartTime
Calcul des StatusCodes	
Méthode de calcul	PercentTime
Bit partiel	Set Sometimes Si l'intervalle n'est pas complet
Bit calculé	Set Always
Bit interpolé	Not Set
Bit brut	Not Set
Bit multivaleur	Not Set
Cas spéciaux communs du StatusCode	
Avant le début des données	Bad_NoData
Après la fin des données	Bad_NoData
Limite de début introuvable	La valeur limite est Bad_NoData, traitée comme une autre valeur de qualité "Bad" dans le calcul (non prise en compte)
Limite de fin introuvable	La valeur limite est Bad_NoData, traitée comme une autre valeur de qualité "Bad" dans le calcul (non prise en compte)
Limite "Bad"	La valeur est traitée comme une autre valeur de qualité "Bad" dans le calcul (non prise en compte)
Limite "Uncertain"	La valeur est traitée comme une autre valeur de qualité différente de "Good" dans le calcul (non prise en compte)

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.10 Minimum

L'*Agrégat* "Minimum" défini dans le Tableau 21 permet d'extraire la valeur brute "Good" minimale dans l'intervalle et de la renvoyer avec l'horodatage de début d'intervalle. Noter que dans le cas où la même valeur minimale se retrouve dans plusieurs horodatages, le bit MultipleValues est défini.

Sauf indication contraire, les *StatusCodes* sont Good, Calculated. Si la valeur minimale se situe au début de l'intervalle, le code de statut est Good, Raw. Si seules les valeurs de qualité "Bad" sont disponibles, le statut *Bad_NoData* est renvoyé.

L'horodatage de l'*Agrégat* est toujours le début de l'intervalle de chaque *ProcessingInterval*.

Tableau 21 – Récapitulatif de l'Agrégat "Minimum"

Caractéristiques de l'Agrégat "Minimum"	
Type	Calculated
Type de données	Same as Source
Limites d'utilisation	None
Horodatage	StartTime
Calcul des StatusCodes	
Méthode de calcul	Custom En l'absence de valeur "Bad", le statut est "Good". En présence de valeurs "Bad", le statut est *Uncertain_SubNormal*. Si une valeur "Uncertain" est inférieure à la valeur "Good" minimale, le statut est *Uncertain_SubNormal*.
Bit partiel	Set Sometimes Si l'intervalle n'est pas complet
Bit calculé	Set Sometimes Si la valeur "Minimum" n'est pas sur la StartTime de l'intervalle ou si le statut était défini sur *Uncertain_SubNormal* en raison de la présence de valeurs autres que "Good" dans l'intervalle
Bit interpolé	Not Set
Bit brut	Set Sometimes Si la valeur "Minimum" est sur la StartTime de l'intervalle
Bit multivaleur	Set Sometimes Si plusieurs valeurs "Good" existent avec la valeur "Minimum"
Cas spéciaux communs du StatusCode	
Avant le début des données	Bad_NoData
Après la fin des données	Bad_NoData
Limite de début introuvable	Non applicable
Limite de fin introuvable	Non applicable
Limite "Bad"	Non applicable
Limite "Uncertain"	Non applicable

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.11 Maximum

L'*Agrégat* "Maximum" défini dans le Tableau 22 permet d'extraire la valeur brute "Good" maximale dans l'intervalle et de la renvoyer avec l'horodatage de début d'intervalle. Noter que dans le cas où la même valeur maximale se retrouve dans plusieurs horodatages, le bit MultipleValues est défini.

Sauf indication contraire, les *StatusCodes* sont Good, Calculated. Si la valeur minimale se situe au début de l'intervalle, le code de statut est Good, Raw. Si seules les valeurs de qualité "Bad" sont disponibles, le statut *Bad_NoData* est renvoyé.

L'horodatage de l'*Agrégat* est toujours le début de l'intervalle de chaque *ProcessingInterval*.

Tableau 22 – Récapitulatif de l'Agrégat "Maximum"

Caractéristiques de l'Agrégat "Maximum"	
Type	Calculated
Type de données	Same as Source
Limites d'utilisation	None
Horodatage	StartTime
Calcul des StatusCodes	
Méthode de calcul	Custom En l'absence de valeur "Bad", le statut est "Good". En présence de valeurs "Bad", le statut est *Uncertain_SubNormal*. Si une valeur "Uncertain" est supérieure à la valeur "Good" maximale, le statut est *Uncertain_SubNormal*.
Bit partiel	Set Sometimes Si l'intervalle n'est pas complet
Bit calculé	Set Sometimes Si la valeur "Maximum" n'est pas sur la *StartTime* de l'intervalle ou si le statut était défini sur *Uncertain_SubNormal* en raison de la présence de valeurs autres que "Good" dans l'intervalle
Bit interpolé	Not Set
Bit brut	Set Sometimes Si la valeur "Maximum" est sur la *StartTime* de l'intervalle
Bit multivaleur	Set Sometimes Si plusieurs valeurs "Good" existent avec la valeur "Maximum"
Cas spéciaux communs du StatusCode	
Avant le début des données	Bad_NoData
Après la fin des données	Bad_NoData
Limite de début introuvable	Non applicable
Limite de fin introuvable	Non applicable
Limite "Bad"	Non applicable
Limite "Uncertain"	Non applicable

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.12 MinimumActualTime

L'*Agrégat* "MinimumActualTime" défini dans le Tableau 23 permet d'extraire la valeur brute "Good" minimale dans l'intervalle et de la renvoyer avec l'horodatage auquel elle s'est produite. Noter que si la même valeur minimale existe dans plusieurs horodatages, la plus ancienne est extraite et les *Bits d'Agrégat* sont définis sur MultipleValues.

Tableau 23 – Récapitulatif de l'Agrégat "MinimumActualTime"

Caractéristiques de l'Agrégat "MinimumActualTime"	
Type	Calculated
Type de données	Same as Source
Limites d'utilisation	None
Horodatage	Durée de la valeur minimale
Calcul des StatusCodes	
Méthode de calcul	Custom En l'absence de valeur "Bad", le statut est "Good". En présence de valeurs "Bad", le statut est *Uncertain_SubNormal*. Si une valeur "Uncertain" est inférieure à la valeur "Good" minimale, le statut est *Uncertain_SubNormal*.
Bit partiel	Set Sometimes Si l'intervalle n'est pas complet
Bit calculé	Set Sometimes Si le statut était défini sur *Uncertain_SubNormal* en raison de valeurs autres que "Good" dans l'intervalle
Bit interpolé	Not Set
Bit brut	Set Sometimes Si une valeur minimale "Good" est renvoyée
Bit multivaleur	Set Sometimes Si plusieurs valeurs "Good" existent avec la valeur "Minimum"
Cas spéciaux communs du StatusCode	
Avant le début des données	Bad_NoData
Après la fin des données	Bad_NoData
Limite de début introuvable	Non applicable
Limite de fin introuvable	Non applicable
Limite "Bad"	Non applicable
Limite "Uncertain"	Non applicable

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.13 MaximumActualTime

L'*Agrégat* "MaximumActualTime" défini dans le Tableau 24 est identique à l'*Agrégat* "MinimumActualTime", sauf qu'il s'agit de la valeur brute maximale dans l'intervalle. Noter que si la même valeur maximale existe dans plusieurs horodatages, la plus ancienne est extraite et les *Bits d'Agrégat* sont définis sur MultipleValues.

Tableau 24 – Récapitulatif de l'Agrégat "MaximumActualTime"

Caractéristiques de l'Agrégat "MaximumActualTime"	
Type	Calculated
Type de données	Same as Source
Limites d'utilisation	None
Horodatage	Durée de la valeur maximale
Calcul des StatusCodes	
Méthode de calcul	Custom En l'absence de valeur "Bad", le statut est "Good". En présence de valeurs "Bad", le statut est *Uncertain_SubNormal*. Si une valeur "Uncertain" est supérieure à la valeur "Good" maximale, le statut est *Uncertain_SubNormal*.
Bit partiel	Set Sometimes Si l'intervalle n'est pas complet
Bit calculé	Set Sometimes Si le statut était défini sur *Uncertain_SubNormal* en raison de valeurs autres que "Good" dans l'intervalle
Bit interpolé	Not Set
Bit brut	Set Sometimes Si une valeur maximale "Good" est renvoyée
Bit multivaleur	Set Sometimes Si plusieurs valeurs "Good" existent avec la valeur "Maximum"
Cas spéciaux communs du StatusCode	
Avant le début des données	Bad_NoData
Après la fin des données	Bad_NoData
Limite de début introuvable	Non applicable
Limite de fin introuvable	Non applicable
Limite "Bad"	Non applicable
Limite "Uncertain"	Non applicable

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.14 Range

L'*Agrégat* "Range" défini dans le Tableau 25 recherche les différences entre les valeurs brutes "Good" maximales et minimales dans l'intervalle. Si l'intervalle ne contient qu'une seule valeur *"Good"*, la plage est nulle. Noter que la plage est toujours nulle ou positive. Si les valeurs autres que "Good" ne sont pas prises en compte lors de la recherche des valeurs minimales ou maximales ou qu'il existe des valeurs "Bad", le statut est *Uncertain_DataSubNormal*.

Tableau 25 – Récapitulatif de l'Agrégat "Range"

Caractéristiques de l'Agrégat "Range"	
Type	Calculated
Type de données	Same as Source
Limites d'utilisation	None
Horodatage	StartTime
Calcul des StatusCodes	
Méthode de calcul	Custom En l'absence de valeur "Bad", le statut est "Good". En présence de valeurs "Bad", le statut est *Uncertain_SubNormal*. Si une valeur "Uncertain" est supérieure à la valeur maximale ou inférieure à la valeur "Good" minimale, le statut est *Uncertain_SubNormal*.
Bit partiel	Set Sometimes Si l'intervalle n'est pas complet
Bit calculé	Set Always
Bit interpolé	Not Set
Bit brut	Not Set
Bit multivaleur	Not Set
Cas spéciaux communs du StatusCode	
Avant le début des données	Bad_NoData
Après la fin des données	Bad_NoData
Limite de début introuvable	Non applicable
Limite de fin introuvable	Non applicable
Limite "Bad"	Non applicable
Limite "Uncertain"	Non applicable

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.15 Minimum2

L'*Agrégat* "Minimum2" défini dans le Tableau 26 permet d'extraire la valeur "Good" minimale de chaque intervalle pour "Minimum", sauf que les *Valeurs limites simples* sont incluses. Les *Valeurs limites simples* de l'intervalle sont déterminées conformément à la définition des *Valeurs limites simples* (voir 3.1.9). Les valeurs "Bad" ne sont pas prises en compte dans le calcul. Le *StatusCode* de l'*Agrégat* est déterminé par calcul (voir 5.3) pour les *Agrégats* fondés sur le temps. Si une valeur limite est renvoyée, le statut indique "Raw", "Calculated" ou "Interpolated".

Si la valeur de "TreatUncertainAsBad" est "False" et qu'une valeur brute "Uncertain" est la valeur minimale, alors cette valeur "Uncertain" est utilisée. Sinon, les valeurs "Uncertain" ne sont pas prises en compte.

Si l'interpolation inclinée est utilisée et que la Limite de fin est la valeur minimale, la Limite de fin est utilisée comme valeur minimale avec l'horodatage défini sur la *startTime* de l'intervalle. La Limite de fin n'est pas prise en compte dans tous les autres cas.

Tableau 26 – Récapitulatif de l'Agrégat "Minimum2"

Caractéristiques de l'Agrégat "Minimum2"	
Type	Calculated
Type de données	Same as Source
Limites d'utilisation	Simple
Horodatage	StartTime
Calcul des StatusCodes	
Méthode de calcul	PercentTime
Bit partiel	Set Sometimes Si l'intervalle n'est pas complet
Bit calculé	Set Sometimes Défini sauf si StartBound est la valeur "Minimum"
Bit interpolé	Set Sometimes Si une limite interpolée est la valeur "Minimum"
Bit brut	Set Sometimes Si une valeur brute est la valeur "Minimum"
Bit multivaleur	Set Sometimes Si plusieurs valeurs "Good" existent avec la même valeur
Cas spéciaux communs du StatusCode	
Avant le début des données	Bad_NoData
Après la fin des données	Bad_NoData
Limite de début introuvable	Traiter la valeur de début comme Bad_NoData et calculer l'*Agrégat*
Limite de fin introuvable	Traiter la valeur de fin comme Bad_NoData et calculer l'*Agrégat*
Limite "Bad"	Utiliser comme une valeur et calculer l'*Agrégat* comme défini
Limite "Uncertain"	Utiliser comme une valeur et calculer l'*Agrégat* comme défini

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.16 Maximum2

L'*Agrégat* "Maximum2" défini dans le Tableau 27 permet d'extraire la valeur "Good" maximale de chaque intervalle pour "Maximum", sauf que les *Valeurs limites simples* sont incluses. Les *Valeurs limites simples* de l'intervalle sont déterminées conformément à la définition des *Valeurs limites simples* (voir 3.1.9). Les valeurs "Bad" ne sont pas prises en compte dans le calcul. Le *StatusCode* de l'*Agrégat* est déterminé par calcul (voir 5.3) pour les *Agrégats* fondés sur le temps. Si une valeur limite est renvoyée, le statut indique "Raw", "Calculated" ou "Interpolated".

Si la valeur de *TreatUncertainAsBad* est "False" et qu'une valeur brute "Uncertain" est la valeur maximale, alors cette valeur "Uncertain" est utilisée. Sinon, les valeurs "Uncertain" ne sont pas prises en compte.

Si l'interpolation inclinée est utilisée et que la Limite de fin est la valeur maximale, la Limite de fin est utilisée comme valeur maximale avec l'horodatage défini sur la *startTime* de l'intervalle. La Limite de fin n'est pas prise en compte dans tous les autres cas.

Tableau 27 – Récapitulatif de l'Agrégat "Maximum2"

Caractéristiques de l'Agrégat "Maximum2"	
Type	Calculated
Type de données	Same as Source
Limites d'utilisation	Simple
Horodatage	StartTime
Calcul des StatusCodes	
Méthode de calcul	PercentTime
Bit partiel	Set Sometimes Si l'intervalle n'est pas complet
Bit calculé	Set Sometimes Défini sauf si StartBound est la valeur "Maximum"
Bit interpolé	Set Sometimes Si une limite interpolée est la valeur "Maximum"
Bit brut	Set Sometimes Si une valeur brute est la valeur "Maximum"
Bit multivaleur	Set Sometimes Si plusieurs valeurs "Good" existent avec la même valeur
Cas spéciaux communs du StatusCode	
Avant le début des données	Bad_NoData
Après la fin des données	Bad_NoData
Limite de début introuvable	Traiter la valeur de début comme Bad_NoData et calculer l'*Agrégat*
Limite de fin introuvable	Traiter la valeur de fin comme Bad_NoData et calculer l'*Agrégat*
Limite "Bad"	Utiliser comme une valeur et calculer l'*Agrégat* comme défini
Limite "Uncertain"	Utiliser comme une valeur et calculer l'*Agrégat* comme défini

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.17 MinimumActualTime2

L'*Agrégat* "MinimumActualTime2" défini dans le Tableau 28 permet d'extraire la valeur "Good" minimale de chaque intervalle défini pour "MinimumActualTime", sauf que les *Valeurs limites simples* sont incluses. Les *Valeurs limites simples* de l'intervalle sont déterminées conformément à la définition des *Valeurs limites simples* (voir 3.1.9). Les valeurs "Bad" ne sont pas prises en compte dans le calcul. Le *StatusCode* de l'*Agrégat* est déterminé par calcul (voir 5.3) pour les *Agrégats* fondés sur le temps. Si une valeur limite est renvoyée, le statut indique "Raw", "Calculated" ou "Interpolated".

Si la valeur de "TreatUncertainAsBad" est "False" et qu'une valeur brute "Uncertain" est la valeur minimale, alors cette valeur "Uncertain" est utilisée. Sinon, les valeurs "Uncertain" ne sont pas prises en compte.

Si l'interpolation inclinée est utilisée et que la Limite de fin est la valeur minimale, la Limite de fin est utilisée comme valeur minimale avec l'horodatage défini sur l'*EffectiveEndTime* de l'intervalle. La Limite de fin n'est pas prise en compte dans tous les autres cas.

Tableau 28 – Récapitulatif de l'Agrégat "MinimumActualTime2"

Caractéristiques de l'Agrégat "MinimumActualTime2"	
Type	Calculated
Type de données	Same as Source
Limites d'utilisation	Simple
Horodatage	Durée de la valeur minimale
Calcul des StatusCodes	
Méthode de calcul	PercentTime
Bit partiel	Set Sometimes Si l'intervalle n'est pas complet
Bit calculé	Not Set
Bit interpolé	Set Sometimes Si une limite interpolée est la valeur "Minimum"
Bit brut	Set Sometimes Si une valeur brute est la valeur "Minimum"
Bit multivaleur	Set Sometimes Si plusieurs valeurs "Good" existent avec la même valeur
Cas spéciaux communs du StatusCode	
Avant le début des données	Bad_NoData
Après la fin des données	Bad_NoData
Limite de début introuvable	Traiter la valeur de début comme Bad_NoData et calculer l'*Agrégat*
Limite de fin introuvable	Traiter la valeur de fin comme Bad_NoData et calculer l'*Agrégat*
Limite "Bad"	Utiliser comme une valeur et calculer l'*Agrégat* comme défini
Limite "Uncertain"	Utiliser comme une valeur et calculer l'*Agrégat* comme défini

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

5.4.3.18 MaximumActualTime2

L'*Agrégat* "MaximumActualTime2" défini dans le Tableau 29 permet d'extraire la valeur "Good" maximale de chaque intervalle pour "MaximumActualTime", sauf que les *Valeurs limites simples* sont incluses. Les *Valeurs limites simples* de l'intervalle sont déterminées conformément à la définition des *Valeurs limites simples* (voir 3.1.9). Les valeurs "Bad" ne sont pas prises en compte dans le calcul. Le *StatusCode* de l'*Agrégat* est déterminé par calcul (voir 5.3) pour les *Agrégats* fondés sur le temps. Si une valeur limite est renvoyée, le statut indique "Raw", "Calculated" ou "Interpolated".

Si la valeur de *TreatUncertainAsBad* est "False" et qu'une valeur brute "Uncertain" est la valeur maximale, alors cette valeur "Uncertain" est utilisée. Sinon, les valeurs "Uncertain" ne sont pas prises en compte.

Si l'interpolation inclinée est utilisée et que la Limite de fin est la valeur maximale, la Limite de fin est utilisée comme valeur maximale avec l'horodatage défini sur l'EffectiveEndTime de l'intervalle. La Limite de fin n'est pas prise en compte dans tous les autres cas.

Tableau 29 – Récapitulatif de l'Agrégat "MaximumActualTime2"

Caractéristiques de l'Agrégat "MaximumActualTime2"	
Type	Calculated
Type de données	Same as Source
Limites d'utilisation	Simple
Horodatage	Durée de la valeur maximale
Calcul des StatusCodes	
Méthode de calcul	PercentTime
Bit partiel	Set Sometimes Si l'intervalle n'est pas complet
Bit calculé	Not Set
Bit interpolé	Set Sometimes Si une limite interpolée est la valeur "Maximum"
Bit brut	Set Sometimes Si une valeur brute est la valeur "Maximum"
Bit multivaleur	Set Sometimes Si plusieurs valeurs sont égales à la valeur "Maximum"
Cas spéciaux communs du StatusCode	
Avant le début des données	Bad_NoData
Après la fin des données	Bad_NoData
Limite de début introuvable	Traiter la valeur de début comme Bad_NoData et calculer l'*Agrégat*
Limite de fin introuvable	Traiter la valeur de fin comme Bad_NoData et calculer l'*Agrégat*
Limite "Bad"	Utiliser comme une valeur et calculer l'*Agrégat* comme défini
Limite "Uncertain"	Utiliser comme une valeur et calculer l'*Agrégat* comme défini

5.4.3.19 Range2

L'*Agrégat* "Range2" défini dans le Tableau 30 détermine la différence entre les valeurs maximale et minimale dans l'intervalle, renvoyées par les *Agrégats* "Minimum2" et "Maximum2". Noter que la plage est toujours nulle ou positive.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV

Tableau 30 – Récapitulatif de l'Agrégat "Range2"

Caractéristiques de l'Agrégat "Range2"	
Type	Calculated
Type de données	Same as Source
Limites d'utilisation	Simple (utilisé dans les calculs de "Minimum2" et "Maximum2")
Horodatage	StartTime
Calcul des StatusCodes	
Méthode de calcul	Custom Si "Minimum2" ou "Maximum2" est "Bad", le statut est *Bad_NoData*. Si "Minimum2" ou "Maximum2" est "Uncertain", le statut est *Uncertain_DataSubNormal*. Sinon, "Good"
Bit partiel	Set Sometimes Si l'intervalle n'est pas complet
Bit calculé	Set Always
Bit interpolé	Not Set
Bit brut	Not Set
Bit multivaleur	Not Set
Cas spéciaux communs du StatusCode	
Avant le début des données	Bad_NoData
Après la fin des données	Bad_NoData
Limite de début introuvable	Traité par "Minimum2" et "Maximum2"
Limite de fin introuvable	Traité par "Minimum2" et "Maximum2"
Limite "Bad"	Traité par "Minimum2" et "Maximum2"
Limite "Uncertain"	Traité par "Minimum2" et "Maximum2"

5.4.3.20 AnnotationCount

L'*Agrégat* "AnnotationCount" défini dans le Tableau 31 renvoie un comptage de toutes les *Annotations* de l'intervalle.

Les *StatusCodes* sont Good, Calculated.

IECNORM.COM : Click to view the full PDF of IEC 62541-13:2020 RLV