



**SLOVENSKI STANDARD**  
**oSIST prEN IEC 62541-13:2024**  
**01-marec-2024**

---

**Enotna arhitektura OPC - 13. del: Zborniki**

OPC Unified Architecture - Part 13: Aggregates

OPC Unified Architecture - Teil 13: Aggregation von Daten

Architecture unifiée OPC - Partie 13: Agrégats

**Ta slovenski standard je istoveten z: prEN IEC 62541-13:2024**

---

**ICS:**

25.040.40	Merjenje in krmiljenje industrijskih postopkov	Industrial process measurement and control
35.240.50	Uporabniške rešitve IT v industriji	IT applications in industry

**oSIST prEN IEC 62541-13:2024**                      **en,fr,de**





PROJECT NUMBER: <b>IEC 62541-13 ED3</b>	
DATE OF CIRCULATION: <b>2024-01-26</b>	CLOSING DATE FOR VOTING: <b>2024-04-19</b>
SUPERSEDES DOCUMENTS: <b>65E/984/RR</b>	

IEC SC 65E : DEVICES AND INTEGRATION IN ENTERPRISE SYSTEMS	
SECRETARIAT: United States of America	SECRETARY: Mr Donald (Bob) Lattimer
OF INTEREST TO THE FOLLOWING COMMITTEES:	PROPOSED HORIZONTAL STANDARD: <input type="checkbox"/> Other TC/SCs are requested to indicate their interest, if any, in this CDV to the secretary.
FUNCTIONS CONCERNED: <input type="checkbox"/> EMC <input type="checkbox"/> ENVIRONMENT <input type="checkbox"/> QUALITY ASSURANCE <input type="checkbox"/> SAFETY	
<input checked="" type="checkbox"/> SUBMITTED FOR CENELEC PARALLEL VOTING <b>Attention IEC-CENELEC parallel voting</b> The attention of IEC National Committees, members of CENELEC, is drawn to the fact that this Committee Draft for Vote (CDV) is submitted for parallel voting. The CENELEC members are invited to vote through the CENELEC online voting system.	<input type="checkbox"/> NOT SUBMITTED FOR CENELEC PARALLEL VOTING

This document is still under study and subject to change. It should not be used for reference purposes.

Recipients of this document are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

Recipients of this document are invited to submit, with their comments, notification of any relevant "In Some Countries" clauses to be included should this proposal proceed. Recipients are reminded that the CDV stage is the final stage for submitting ISC clauses. (SEE [AC/22/2007](#) OR [NEW GUIDANCE DOC](#)).

TITLE:

**OPC Unified Architecture - Part 13: Aggregates**

PROPOSED STABILITY DATE: 2026

NOTE FROM TC/SC OFFICERS:

**Copyright © 2023 International Electrotechnical Commission, IEC.** All rights reserved. It is permitted to download this electronic file, to make a copy and to print out the content for the sole purpose of preparing National Committee positions. You may not copy or "mirror" the file or printed version of the document, or any part of it, for any other purpose without permission in writing from IEC.

## CONTENTS

1			
2			
3	FOREWORD .....		iv
4	1 Scope .....		1
5	2 Normative references .....		1
6	3 Terms, definitions, and abbreviated terms .....		1
7	3.1 Terms and definitions .....		1
8	3.2 Abbreviated terms .....		4
9	4 Aggregate information model .....		4
10	4.1 General .....		4
11	4.2 Aggregate Objects .....		4
12	4.2.1 General .....		4
13	4.2.2 AggregateFunction Object .....		5
14	4.3 MonitoredItem AggregateFilter .....		7
15	4.3.1 MonitoredItem AggregateFilter Defaults .....		7
16	4.3.2 MonitoredItem Aggregates and Bounding Values .....		8
17	4.4 Exposing Supported Functions and Capabilities .....		8
18	5 Aggregate specific usage of Services .....		9
19	5.1 General .....		9
20	5.2 Aggregate data handling .....		9
21	5.2.1 Overview .....		9
22	5.2.2 ReadProcessedDetails structure overview .....		9
23	5.2.3 AggregateFilter structure overview .....		9
24	5.3 Aggregates StatusCodes .....		10
25	5.3.1 Overview .....		10
26	5.3.2 Operation level result codes .....		10
27	5.3.3 Aggregate Information Bits .....		10
28	5.4 Aggregate details .....		12
29	5.4.1 General .....		12
30	5.4.2 Common characteristics .....		12
31	5.4.3 Specific aggregated data handling .....		15
32	Annex A (informative) .....		46
33	A.1 Aggregate Examples .....		46
34	A.1.1 Aggregate Examples CSV .....		46

## FIGURES

38	Figure 1 – Representation of Aggregate Configuration information in the AddressSpace .....	9
39	Figure 2 – Variable with Stepped = False and Simple Bounding Values .....	16
40	Figure 3 – Variable with Stepped = True and Interpolated Bounding Values .....	17

## TABLES

44	Table 1 – Interpolation examples .....	2
45	Table 2 – AggregateConfigurationType Definition .....	4

46	Table 3 – Aggregate Functions Definition .....	5
47	Table 4 – AggregateFunctionType Definition .....	6
48	Table 5 – Standard AggregateType Nodes .....	7
49	Table 6 – ReadProcessedDetails .....	9
50	Table 7 – AggregateFilter structure .....	10
51	Table 8 – Bad operation level result codes .....	10
52	Table 9 – Uncertain operation level result codes .....	10
53	Table 10 – Data location .....	11
54	Table 11 – Additional information .....	11
55	Table 12 – History Aggregate interval information .....	13
56	Table 13 – Standard History Aggregate Data Type information.....	14
57	Table 14 – Aggregate table description .....	18
58	Table 15 – Interpolative Aggregate summary.....	19
59	Table 16 – Average Aggregate summary .....	20
60	Table 17 – TimeAverage Aggregate summary .....	20
61	Table 18 – TimeAverage2 Aggregate summary .....	22
62	Table 19 – Total Aggregate summary .....	22
63	Table 20 – Total2 Aggregate summary .....	23
64	Table 21 – Minimum Aggregate summary.....	24
65	Table 22 – Maximum Aggregate summary .....	24
66	Table 23 – MinimumActualTime Aggregate summary .....	25
67	Table 24 – MaximumActualTime Aggregate summary.....	26
68	Table 25 – Range Aggregate summary.....	27
69	Table 26 – Minimum2 Aggregate summary.....	28
70	Table 27 – Maximum2 Aggregate summary .....	28
71	Table 28 – MinimumActualTime2 Aggregate summary.....	29
72	Table 29 – MaximumActualTime2 Aggregate summary.....	30
73	Table 30 – Range2 Aggregate summary.....	30
74	Table 31 – AnnotationCount Aggregate summary .....	31
75	Table 32 – Count Aggregate summary.....	32
76	Table 33 – DurationInStateZero Aggregate summary .....	32
77	Table 34 – DurationInStateNonZero Aggregate summary .....	33
78	Table 35 – NumberOfTransitions Aggregate summary .....	34
79	Table 36 – Start Aggregate summary .....	35
80	Table 37 – End Aggregate summary.....	35
81	Table 38 – Delta Aggregate summary.....	36
82	Table 39 – StartBound Aggregate summary .....	36
83	Table 40 – EndBound Aggregate summary.....	37
84	Table 41 – DeltaBounds Aggregate summary .....	37
85	Table 42 – DurationGood Aggregate summary .....	38
86	Table 43 – DurationBad Aggregate summary .....	39
87	Table 44 – PercentGood Aggregate summary .....	40
88	Table 45 – PercentBad Aggregate summary.....	40

89	Table 46 – WorstQuality Aggregate summary .....	41
90	Table 47 – WorstQuality2 Aggregate summary .....	42
91	Table 48 – StandardDeviationSample Aggregate summary .....	42
92	Table 49 – VarianceSample Aggregate summary .....	43
93	Table 50 – StandardDeviationPopulation Aggregate summary .....	44
94	Table 51 – VariancePopulation Aggregate summary .....	44
95		
96		

**iTeh Standards**  
**(<https://standards.iteh.ai>)**  
**Document Preview**

[oSIST prEN IEC 62541-13:2024](https://standards.iteh.ai/catalog/standards/sist/3f0c48a0-f5f6-4f47-8717-1de7904dab18/osist-pren-iec-62541-13-2024)

<https://standards.iteh.ai/catalog/standards/sist/3f0c48a0-f5f6-4f47-8717-1de7904dab18/osist-pren-iec-62541-13-2024>

## 97 INTERNATIONAL ELECTROTECHNICAL COMMISSION

98

99

100

101

102

103

104

**OPC UNIFIED ARCHITECTURE –****Part 13: Aggregates****FOREWORD**

105 1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all  
 106 national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-  
 107 operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to  
 108 other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available  
 109 Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical  
 110 committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work.  
 111 International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation.  
 112 IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions  
 113 determined by agreement between the two organizations.

114 2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international  
 115 consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC  
 116 National Committees.

117 3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees  
 118 in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate,  
 119 IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.

120 4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently  
 121 to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication  
 122 and the corresponding national or regional publication shall be clearly indicated in the latter.

123 5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment  
 124 services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by  
 125 independent certification bodies.

126 6) All users should ensure that they have the latest edition of this publication.

127 7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members  
 128 of its technical committees and IEC National Committees for any personal injury, property damage or other damage of  
 129 any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the  
 130 publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.

131 8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable  
 132 for the correct application of this publication.

133 9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights.  
 134 IEC shall not be held responsible for identifying any or all such patent rights.

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

138 This fourth edition cancels and replaces the third edition published in 2020. This edition constitutes a  
 139 technical revision.

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

141 a) Multiple fixes for the computation of aggregates

- 142 • The Raw status bit is always be set for non-bad StatusCodes for the Start and End  
 143 aggregates.
- 144 • Entries in the Interpolative examples Tables A2.2 Historian1, Historian2, and Historian3  
 145 have been changed from Good to Good, Raw status codes when the timestamp matches with  
 146 the timestamp of the data source.
- 147 • Missing tables were added for DurationInStateZero and DurationInStateNonZero.
- 148 • The value of zero has been removed for results with a StatusCode of bad.
- 149 • Data Type was listed as "Status Code" when it should be "Double" for both Standard  
 150 Deviation and both Variance Aggregates.
- 151 • Rounding Error in TimeAverage and TimeAverage2 have been corrected.

- 152 • The status codes have been corrected for the last two intervals and the value has been
- 153 corrected in the last interval.
- 154 • The wording has been changed to be more consistent with the certification testing tool.
- 155 • UsedSlopedExtrapolation set to true for Historian2 and all examples locations needed new
- 156 values or status' are modified.
- 157 • Values affected by percent good and percent bad have been updated.
- 158 • PercentGood/PercentBad are now accounted for in the calculation.
- 159 • TimeAverage must use SlopedInterpolation but the Time aggregate is incorrectly allowed to
- 160 used Stepped Interpolation.
- 161 • Partial bit is now correctly calculated.
- 162 • Unclear sentence was removed.
- 163 • Examples have been moved to a CSV.
- 164 • The value and status code for Historian 3 have been updated.
- 165 • TimeAverage2 Historian1 now takes uncertain regions into account when calculating
- 166 StatusCodes.
- 167 • TimeAverage2 Historian2 now takes uncertain regions into account when calculating
- 168 StatusCodes.
- 169 • Total2 Historian1 now takes uncertain regions into account when calculating StatusCodes
- 170 • Total2 Historian2 now takes uncertain regions into account when calculating StatusCodes
- 171 • Maximum2 Historian1 now takes uncertain regions into account when calculating
- 172 StatusCodes
- 173 • MaximumActualTime2 Historian1 now takes uncertain regions into account when calculating
- 174 StatusCodes
- 175 • Minimum2 Historian1 now takes uncertain regions into account when calculating
- 176 StatusCodes
- 177 • MinimumActualTime2 Historian1 now has the StatusCodes calculated while using the
- 178 TreatUncertainAsBad flag.
- 179 • Range2 Historian1 now looks at TreatUncertainAsBad in the calculation of the StatusCodes.
- 180 • Clarifications were made to the text defining how PercentGood/PercentBad are used. The
- 181 TimeAverage2 and Total2 aggregates had their table values and StatusCodes corrected.

oSIST prEN IEC 62541-13:2024

183 The text of this International Standard is based on the following documents: 18/osist-pren-iec-62541-13-2024

CDV	Report on voting
65E/XX/CDV	65E/XX/RVC

184  
185 Full information on the voting for the approval of this International Standard can be found in the report  
186 on voting indicated in the above table.

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

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

190 *Italics* are used to denote a defined term or definition that appears in the "Terms and definition" clause  
191 in one of the parts of the IEC 62541 series.

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

194 The *italicized terms and names* are, with a few exceptions, written in camel-case (the practice of  
195 writing compound words or phrases in which the elements are joined without spaces, with each  
196 element's initial letter capitalized within the compound). For example, the defined term is



197 *AddressSpace* instead of Address Space. This makes it easier to understand that there is a single  
198 definition for *AddressSpace*, not separate definitions for Address and Space.

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

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

- 204 • reconfirmed,
- 205 • withdrawn,
- 206 • replaced by a revised edition, or
- 207 • amended.

208

**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.**

209

## iTeh Standards (<https://standards.iteh.ai>) Document Preview

[oSIST prEN IEC 62541-13:2024](https://standards.iteh.ai/catalog/standards/sist/3f0c48a0-f5f6-4f47-8717-1de7904dab18/osist-pren-iec-62541-13-2024)

<https://standards.iteh.ai/catalog/standards/sist/3f0c48a0-f5f6-4f47-8717-1de7904dab18/osist-pren-iec-62541-13-2024>

210  
211

# OPC Unified Architecture Specification

212  
213  
214  
215

## Part 13: Aggregates

### 216 1 Scope

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

### 219 2 Normative references

220 The following documents, in whole or in part, are normatively referenced in this document and  
221 are indispensable for its application. For dated references, only the edition cited applies. For  
222 undated references, the latest edition of the referenced document (including any amendments  
223 and errata) applies.

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

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

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

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

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

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

### 230 3 Terms, definitions, and abbreviated terms

#### 231 3.1 Terms and definitions

232 For the purposes of this document, the terms and definitions given in IEC 62541-1, IEC 62541-  
233 3, IEC 62541-4, and IEC 62541-11 as well as the following apply.

##### 234 3.1.1

##### 235 **ProcessingInterval**

236 timespan for which derived values are produced based on a specified *Aggregate*

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

##### 241 3.1.2

##### 242 **Interpolated data**

243 data that is calculated from data samples

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

##### 246 3.1.3

##### 247 **EffectiveEndTime**

248 time immediately before *endTime*

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

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

256 **3.1.4**  
 257 **Extrapolated data**  
 258 data constructed from a discrete data set but is outside of the discrete data set

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

262 **3.1.5**  
 263 **SlopedInterpolation**  
 264 simple linear interpolation

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

266 **3.1.6**  
 267 **SteppedInterpolation**  
 268 Interpolation holding the last data point constant or interpolating the value based on a horizontal  
 269 line fit

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

271 **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

272  
 273 **3.1.7**  
 274 **bounding values**  
 275 values at the *startTime* and *endTime* needed for *Aggregates* to compute the result

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

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

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

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

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

292 **3.1.8**  
 293 **interpolated bounding values**  
 294 *bounding values* determined by a calculation using the nearest Good value

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

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

301 The calculation can be expressed with the following formula:

$$302 \quad V_{\text{bound}} = (T_{\text{bound}} - T_{\text{before}}) \times (V_{\text{after}} - V_{\text{before}}) / (T_{\text{after}} - T_{\text{before}}) + V_{\text{before}}$$

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

304 If no non-Bad values exist before the timestamp the *StatusCode* is *Bad\_NoData*. The *StatusCode* is  
 305 *Uncertain\_DataSubNormal* if any Bad values exist between the before value and after value. If either the before  
 306 value or the after value are Uncertain the *StatusCode* is *Uncertain\_DataSubNormal*. If the after value does not exist  
 307 the before value shall be extrapolated using *SlopedExtrapolation* or *SteppedExtrapolation*.

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

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

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

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

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

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

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

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

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

### 331 3.1.9

#### 332 simple bounding values

333 *bounding values* determined by a calculation using the nearest value

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

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

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

342 If a Raw value at the timestamp is Bad the *StatusCode* is *Bad\_NoData*. If the value before the timestamp is Bad the  
 343 *StatusCode* is *Bad\_NoData*. If the value before the timestamp is Uncertain the *StatusCode* is  
 344 *Uncertain\_DataSubNormal*. If the value after the timestamp is Bad or Uncertain the *StatusCode* is  
 345 *Uncertain\_DataSubNormal*.

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

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

350 If a Raw value at the timestamp is Bad the *StatusCode* is *Bad\_NoData*. If the value before the timestamp is Bad the  
 351 *StatusCode* is *Bad\_NoData*. If the value before the timestamp is Uncertain the *StatusCode* is  
 352 *Uncertain\_DataSubNormal*.

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

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

360 In the same way, if the earliest time of an interval starts before the first data point in history and the latest time is  
 361 after the first data point in history, then the interval will be treated as if the interval extends from the first data point  
 362 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).

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

### 366 3.2 Abbreviated terms

367	DA	Data Access
368	HA	Historical Access (access to historical data or events)
369	HDA	Historical Data Access
370	UA	Unified Architecture

## 371 4 Aggregate information model

### 372 4.1 General

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

### 377 4.2 Aggregate Objects

#### 378 4.2.1 General

##### 379 4.2.1.1 Overview

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

##### 383 4.2.1.2 AggregateConfigurationType

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

388 **Table 2 – AggregateConfigurationType Definition**

Attribute	Value				
BrowseName	AggregateConfigurationType				
IsAbstract	False				
References	NodeClass	BrowseName	Data Type	TypeDefinition	ModellingRule
Subtype of the <i>BaseObjectType</i> 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
Conformance Units					
Aggregate Master Configuration					

389  
 390 The *TreatUncertainAsBad Variable* indicates how the *Server* treats data returned with a  
 391 *StatusCode* severity Uncertain with respect to *Aggregate* calculations. A value of True indicates  
 392 the *Server* considers the severity equivalent to *Bad*, a value of False indicates the *Server*  
 393 considers the severity equivalent to *Good*, unless the *Aggregate* definition says otherwise. The  
 394 default value is True. Note that the value is still treated as Uncertain when the *StatusCode* for  
 395 the result is calculated.

396 The *PercentDataBad Variable* indicates the minimum percentage of Bad data in a given interval  
 397 required for the *StatusCode* for the given interval for processed data request to be set to *Bad*.  
 398 (Uncertain is treated as defined above.) Refer to 5.4.3 for details on using this *Variable* when

399 assigning *StatusCodes*. For details on which *Aggregates* use the *PercentDataBad Variable*, see  
400 the definition of each *Aggregate*. The default value is 100.

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

406 The following calculations are used to determine the *StatusCode* which will be used to calculate  
407 the value of the aggregate. Refer to 5.4.3 for details on using these *Variables* when assigning  
408 *StatusCodes*. The *PercentDataGood* and *PercentDataBad* shall follow the following relationship  
409  $PercentDataGood \geq (100 - PercentDataBad)$ . If they are equal the result of the  
410 *PercentDataGood* calculation is used. If the values entered for *PercentDataGood* and  
411 *PercentDataBad* do not result in a valid calculation (e.g. Bad = 80; Good = 0) the result will  
412 have a *StatusCode* of *Bad\_AggregateInvalidInputs*. The *StatusCode*  
413 *Bad\_AggregateInvalidInputs* will be returned if the value of *PercentDataGood* or  
414 *PercentDataBad* exceed 100.

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

## 421 4.2.2 AggregateFunction Object

### 422 4.2.2.1 General

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

429 **Table 3 – Aggregate Functions Definition**

Attribute	Value				
BrowseName	AggregateFunctions				
References	Node Class	BrowseName	Data Type	Type Definition	Modelling Rule
HasTypeDefinition	Object Type	FolderType	Defined in IEC 62541-5		
Conformance Units					
Historical Access Aggregates					

430

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

### 435 4.2.2.2 AggregateFunctionType

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