DIN EN IEC 62541-5

DIN

ICS 25.040.40; 35.200; 35.240.50

Entwurf

Einsprüche bis 2019-10-09
Vorgesehen als Ersatz für
DIN EN 62541-5:2015-11

OPC Unified Architecture – Teil 5: Informationsmodell (IEC 65E/603/CDV:2018); Englische Fassung prEN IEC 62541-5:2018

OPC Unified Architecture –
Part 5: Information Model
(IEC 65E/603/CDV:2018);
English version prEN IEC 62541-5:2018

Architecture unifiée OPC –
Partie 5: Modèle d'informations
(IEC 65E/603/CDV:2018);
Version anglaise prEN IEC 62541-5:2018

Anwendungswarnvermerk

Dieser Norm-Entwurf mit Erscheinungsdatum 2019-08-09 wird der Öffentlichkeit zur Prüfung und Stellungnahme vorgelegt.

Weil die beabsichtigte Norm von der vorliegenden Fassung abweichen kann, ist die Anwendung dieses Entwurfs besonders zu vereinbaren.

Stellungnahmen werden erbeten

- vorzugsweise online im Norm-Entwurfs-Portal von DIN unter www.din.de/go/entwuerfe bzw. für Norm-Entwürfe der DKE auch im Norm-Entwurfs-Portal der DKE unter www.entwuerfe.normenbibliothek.de, sofern dort wiedergegeben;
- oder als Datei per E-Mail an dke@vde.com möglichst in Form einer Tabelle. Die Vorlage dieser Tabelle kann im Internet unter www.din.de/go/stellungnahmen-norm-entwuerfe oder für Stellungnahmen zu Norm-Entwürfen der DKE unter www.dke.de/stellungnahme abgerufen werden;
- oder in Papierform an die DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik in DIN und VDE, Stresemannallee 15, 60596 Frankfurt am Main.

Die Empfänger dieses Norm-Entwurfs werden gebeten, mit ihren Kommentaren jegliche relevanten Patentrechte, die sie kennen, mitzuteilen und unterstützende Dokumentationen zur Verfügung zu stellen.

Gesamtumfang 158 Seiten

DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik in DIN und VDE

AVA Documents Document Sharing Hub

Full document details: E DIN EN IEC 62541-5:2019-09

Nationales Vorwort

Die Englische Fassung des europäischen Dokuments prEN IEC 62541-5:2018 „OPC Unified Architecture – Teil 5: Informationsmodell“ (Entwurf in der Umfrage) ist unverändert in diesen Norm-Entwurf übernommen worden.

Die Internationale Elektrotechnische Kommission (IEC) und das Europäische Komitee für Elektrotechnische Normung (CENELEC) haben vereinbart, dass ein auf IEC-Ebene erarbeiteter Entwurf für eine Internationale Norm zeitgleich (parallel) bei IEC und CENELEC zur Umfrage (CDV-Stadium) und Abstimmung als FDIS (en: Final Draft International Standard) bzw. Schluss-Entwurf für eine Europäische Norm gestellt wird, um eine Beschleunigung und Straffung der Normungsarbeit zu erreichen. Dem entsprechend ist das internationale Dokument IEC 65E/603/CDV:2018 „OPC Unified Architecture – Part 5: Information Model“ unverändert in den Entwurf prEN IEC 62541-5:2018 übernommen worden.

Das internationale Dokument wurde vom SC 65E „Devices and integration in enterprise systems“ der Internationalen Elektrotechnischen Kommission (IEC) erarbeitet.

Bei der Abstimmung zu dem Europäischen Schluss-Entwurf bei CENELEC und dem Internationalen Schluss-Entwurf bei IEC [Final Draft International Standard (FDIS)] sind jeweils nur „JA/NEIN“-Entscheidungen möglich, wobei „NEIN“-Entscheidungen fundiert begründet werden müssen. Dokumente, die bei CENELEC als Europäische Norm angenommen und ratifiziert werden, sind unverändert als Deutsche Normen zu übernehmen.

Für dieses Dokument ist das nationale Arbeitsgremium K 931 „Systemaspekte der Automatisierung“ der DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik in DIN und VDE (www.dke.de) zuständig.

Für den Fall einer undatierten Verweisung im normativen Text (Verweisung auf ein Dokument ohne Angabe des Ausgabedatums und ohne Hinweis auf eine Abschnittsnummer, eine Tabelle, ein Bild usw.) bezieht sich die Verweisung auf die jeweils aktuellste Ausgabe des in Bezug genommenen Dokuments.

Für den Fall einer datierten Verweisung im normativen Text bezieht sich die Verweisung immer auf die in Bezug genommene Ausgabe des Dokuments.

Der Zusammenhang der zitierten Dokumente mit den entsprechenden deutschen Dokumenten ergibt sich, soweit ein Zusammenhang besteht, grundsätzlich über die Nummer der entsprechenden IEC-Publikation. Beispiel: IEC 60068 ist als EN 60068 als Europäische Norm durch CENELEC übernommen und als DIN EN 60068 ins Deutsche Normenwerk aufgenommen.

Das Präsidium des DIN hat mit Präsidialbeschluss 1/2004 festgelegt, dass DIN-Normen, deren Inhalt sich auf internationale Arbeitsergebnisse der Informationsverarbeitung gründet, unter bestimmten Bedingungen allein in englischer Sprache veröffentlicht werden dürfen. Diese Bedingungen sind für den vorliegenden Norm-Entwurf erfüllt.

Da sich die Benutzer des vorliegenden Norm-Entwurfs der englischen Sprache als Fachsprache bedienen, wird die englische Fassung der prEN IEC 62541-5 veröffentlicht. Zu deren Abschnitt 1, der den Anwendungsbereich festlegt, und Unterabschnitt 3.1, der die Begriffe festlegt, wurde eine Übersetzung angefertigt und als informativer Nationaler Anhang NA des vorliegenden Norm-Entwurfs hinzugefügt. Für einige der verwendeten Begriffe existieren keine gebräuchlichen deutschen Benennungen, da sich die deutschen Anwender in der Regel ebenfalls der englischen Benennungen bedienen. Dieser Norm-Entwurf steht nicht in unmittelbarem Zusammenhang mit Rechtsvorschriften und ist nicht als Sicherheitsnorm anzusehen.

Es ist vorgesehen, auch bei der entsprechenden zukünftigen Deutschen Norm auf die deutsche Sprachfassung zu verzichten und diese in der englischsprachigen Fassung zu veröffentlichen.

Es wird auf die Möglichkeit hingewiesen, dass einige Elemente dieses Dokuments Patentrechte berühren können. IEC ist nicht dafür verantwortlich, einige oder alle diesbezüglichen Patentrechte zu identifizieren.

Documents
Document Sharing Hub

Full document details: E DIN EN IEC 62541-5:2019-09

Das Original-Dokument enthält Bilder in Farbe, die in der Papierversion in einer Graustufen-Darstellung wiedergegeben werden. Elektronische Versionen dieses Dokuments enthalten die Bilder in der originalen Farbdarstellung.

Änderungen

Gegenüber DIN EN 62541-5:2015-11 wurden folgende Änderungen vorgenommen:

a) Anhang F zur Benutzerautorisierung hinzugefügt. Beschreibt das Rolleninformationsmodell, das auch die Konfiguration von Rollen ermöglicht;

b) neue Datentypen hinzugefügt: „Union“, „Decimal“, „OptionSet“, „DateString“, „TimeString“, „DurationString“, „NormalizedString“, „DecimalString“ und „AudioDataType“;

c) Verfahren zur Anforderung einer Zustandsänderung in einem Server hinzugefügt;

d) Verfahren zur Einstellung der Subskription in den Permanentmodus hinzugefügt;

e) Verfahren zur Anforderung der erneuten Sendung von od-Daten von einer Subskription hinzugefügt;

f) Konzept zur Ermöglichung der zeitweiligen Erstellung von Dateien zum Schreiben oder Lesen vom Server in C.4 hinzugefügt;

g) neuen Variablentyp zur Unterstützung der Auswahllisten hinzugefügt;

h) optionale Eigenschaften zum FiniteStateMachineType hinzugefügt, um aktuell verfügbare Zustände und Übergänge anzuzeigen;

i) Eigenschaft UrisVersion zum ServerTyp hinzugefügt. Diese Versionsinformation kann für den sitzungslosen Dienstaufruf genutzt werden.

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

Nationaler Anhang NA
(informativ)

Auszugsweise Übersetzung

Reihenfolge und Inhalt der folgenden Abschnitte sind identisch mit denen in den jeweiligen Abschnitten der Englischen Fassung.

1 Anwendungsbereich

Dieses Dokument legt das Informationsmodell der OPC Unified Architecture (vereinheitlichte Architektur der plattformunabhängigen Kommunikation) fest. Das Informationsmodell legt die genormten *Knoten* (en: *Nodes*) eines *AdressRaums* (en: *AddressSpace*) eines *Servers* fest. Diese *Knoten* sind sowohl genormte Typen als auch genormte Instanzen, welche für Diagnostik oder als Eintrittspunkt zu serverspezifischen *Knoten* verwendet werden. Folglich legt das Informationsmodell den *AdressRaum* eines leeren OPC-UA-*Servers* fest. Es wird jedoch nicht erwartet, dass alle Server alle diese *Knoten* bereitstellen.

3 Begriffe und Konventionen

3.1 Begriffe

Für die Anwendung dieses Dokuments gelten die Begriffe nach IEC TR 62541-1, IEC 62541-3 und die folgenden Begriffe.

ISO und IEC stellen terminologische Datenbanken für die Anwendung in der Normung unter den folgenden Adressen bereit:

- IEC Electropedia: verfügbar unter http://www.electropedia.org/
- ISO Online Browsing Platform: verfügbar unter http://www.iso.org/obp

3.1.1
ClientBenutzerID
(en: ClientUserID)
Zeichenkette, die den Benutzer des Clients identifiziert, der eine Aktion anfordert

Anmerkung 1 zum Begriff: Die *ClientBenutzerID* wird direkt oder indirekt vom *Token BenutzerIdentität* (en: *UserIdentityToken*) erhalten, der vom *Client* beim Aufruf des *Dienstes AktiviereSitzung* (en: ActivateSession *Service*) übergeben wird. Für Einzelheiten siehe 6.4.3.

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

CONTENTS

FIGURES 7
TABLES 8
FOREWORD 14
1 Scope 1
2 Normative references 1
3 Terms, definitions and conventions 2
3.1 Terms and definitions 2
3.2 Abbreviations and symbols 2
3.3 Conventions for Node descriptions 2
4 NodeIds and BrowseNames 3
4.1 NodeIds 3
4.2 BrowseNames 4
5 Common Attributes 4
5.1 General 4
5.2 Objects 4
5.3 Variables 4
5.4 VariableTypes 5
5.5 Methods 5
6 Standard ObjectTypes 5
6.1 General 5
6.2 BaseObjectType 5
6.3 ObjectTypes for the Server Object 6
6.3.1 ServerType 6
6.3.2 ServerCapabilitiesType 9
6.3.3 ServerDiagnosticsType 11
6.3.4 SessionsDiagnosticsSummaryType 12
6.3.5 SessionDiagnosticsObjectType 12
6.3.6 VendorServerInfoType 13
6.3.7 ServerRedundancyType 13
6.3.8 TransparentRedundancyType 13
6.3.9 NonTransparentRedundancyType 14
6.3.10 NonTransparentNetworkRedundancyType 14
6.3.11 OperationLimitsType 15
6.3.12 AddressSpaceFileType 16
6.3.13 NamespaceMetadataType 16
6.3.14 NamespacesType 18
6.4 ObjectTypes used as EventTypes 18
6.4.1 General 18
6.4.2 BaseEventType 18
6.4.3 AuditEventType 21
6.4.4 AuditSecurityEventType 21
6.4.5 AuditChannelEventType 22
6.4.6 AuditOpenSecureChannelEventType 22
6.4.7 AuditSessionEventType 23

AVA Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

6.4.8 AuditCreateSessionEventType 23
6.4.9 AuditUrlMismatchEventType 24
6.4.10 AuditActivateSessionEventType 24
6.4.11 AuditCancelEventType 25
6.4.12 AuditCertificateEventType 25
6.4.13 AuditCertificateDataMismatchEventType 25
6.4.14 AuditCertificateExpiredEventType 26
6.4.15 AuditCertificateInvalidEventType 26
6.4.16 AuditCertificateUntrustedEventType 26
6.4.17 AuditCertificateRevokedEventType 27
6.4.18 AuditCertificateMismatchEventType 27
6.4.19 AuditNodeManagementEventType 27
6.4.20 AuditAddNodesEventType 28
6.4.21 AuditDeleteNodesEventType 28
6.4.22 AuditAddReferencesEventType 29
6.4.23 AuditDeleteReferencesEventType 29
6.4.24 AuditUpdateEventType 29
6.4.25 AuditWriteUpdateEventType 30
6.4.26 AuditHistoryUpdateEventType 30
6.4.27 AuditUpdateMethodEventType 31
6.4.28 SystemEventType 31
6.4.29 DeviceFailureEventType 31
6.4.30 SystemStatusChangeEventType 32
6.4.31 BaseModelChangeEventType 32
6.4.32 GeneralModelChangeEventType 32
6.4.33 SemanticChangeEventType 33
6.4.34 EventQueueOverflowEventType 33
6.4.35 ProgressEventType 33
6.5 ModellingRuleType 34
6.6 FolderType 34
6.7 DataTypeEncodingType 34
6.8 AggregateFunctionType 34
7 Standard VariableTypes 35
7.1 General 35
7.2 BaseVariableType 35
7.3 PropertyType 35
7.4 BaseDataVariableType 36
7.5 ServerVendorCapabilityType 36
7.6 ServerStatusType 36
7.7 BuildInfoType 37
7.8 ServerDiagnosticsSummaryType 37
7.9 SamplingIntervalDiagnosticsArrayType 38
7.10 SamplingIntervalDiagnosticsType 38
7.11 SubscriptionDiagnosticsArrayType 39
7.12 SubscriptionDiagnosticsType 39
7.13 SessionDiagnosticsArrayType 40
7.14 SessionDiagnosticsVariableType 41
7.15 SessionSecurityDiagnosticsArrayType 43
7.16 SessionSecurityDiagnosticsType 43

AVA Documents Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

7.17 OptionSetType 44
7.18 SelectionListType 45
7.19 AudioVariableType 45
8 Standard Objects and their Variables 46
8.1 General 46
8.2 Objects used to organise the AddressSpace structure 46
8.2.1 Overview 46
8.2.2 Root 46
8.2.3 Views 47
8.2.4 Objects 47
8.2.5 Types 48
8.2.6 ObjectTypes 48
8.2.7 VariableTypes 49
8.2.8 ReferenceTypes 50
8.2.9 DataTypes 51
8.2.10 EventTypes 52
8.3 Server Object and its containing Objects 52
8.3.1 General 52
8.3.2 Server Object 53
8.4 ModellingRule Objects 54
8.4.1 ExposesItsArray 54
8.4.2 Mandatory 54
8.4.3 Optional 54
8.4.4 OptionalPlaceholder 54
8.4.5 MandatoryPlaceholder 55
9 Standard Methods 55
9.1 GetMonitoredItems 55
9.2 ResendData 55
9.3 SetSubscriptionDurable 56
9.4 RequestServerStateChange 56
10 Standard Views 57
11 Standard ReferenceTypes 57
11.1 References 57
11.2 HierarchicalReferences 58
11.3 NonHierarchicalReferences 58
11.4 HasChild 58
11.5 Aggregates 58
11.6 Organizes 59
11.7 HasComponent 59
11.8 HasOrderedComponent 59
11.9 HasProperty 59
11.10 HasSubtype 60
11.11 HasModellingRule 60
11.12 HasTypeDefinition 60
11.13 HasEncoding 60
11.14 HasEventSource 61
11.15 HasNotifier 61
11.16 GeneratesEvent 61

AVA Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

11.17 AlwaysGeneratesEvent 61
12 Standard DataTypes 62
12.1 Overview 62
12.2 DataTypes defined in IEC 62541-3 62
12.3 DataTypes defined in IEC 62541-4 67
12.4 BuildInfo 68
12.5 RedundancySupport 68
12.6 ServerState 69
12.7 RedundantServerDataType 69
12.8 SamplingIntervalDiagnosticsDataType 69
12.9 ServerDiagnosticsSummaryDataType 70
12.10 ServerStatusDataType 71
12.11 SessionDiagnosticsDataType 71
12.12 SessionSecurityDiagnosticsDataType 73
12.13 ServiceCounterDataType 73
12.14 StatusResult 74
12.15 SubscriptionDiagnosticsDataType 74
12.16 ModelChangeStructureDataType 75
12.17 SemanticChangeStructureDataType 76
12.18 BitFieldMaskDataType 76
12.19 NetworkGroupDataType 77
12.20 EndpointUrlListDataType 77
12.21 KeyValuePair 77
12.22 EndpointType 78
Annex A (informative) Design decisions when modelling the server information 79
A.1 Overview 79
A.2 ServerType and Server Object 79
A.3 Typed complex Objects beneath the Server Object 79
A.4 Properties versus DataVariables 79
A.5 Complex Variables using complex DataTypes 80
A.6 Complex Variables having an array 80
A.7 Redundant information 80
A.8 Usage of the BaseDataVariableType 80
A.9 Subtyping 81
A.10 Extensibility mechanism 81
Annex B (normative) StateMachines 82
B.1 General 82
B.2 Examples of finite state machines 82
B.2.1 Simple state machine 82
B.2.2 State machine containing substates 83
B.3 Definition of state machine 84
B.4 Representation of state machines in the AddressSpace 84
B.4.1 Overview 84
B.4.2 StateMachineType 85
B.4.3 StateVariableType 86
B.4.4 TransitionVariableType 87
B.4.5 FiniteStateMachineType 87
B.4.6 FiniteStateVariableType 89
B.4.7 FiniteTransitionVariableType 90

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

B.4.8 StateType 90
B.4.9 InitialStateType 91
B.4.10 TransitionType 92
B.4.11 FromState 92
B.4.12 ToState 92
B.4.13 HasCause 93
B.4.14 HasEffect 93
B.4.15 HasSubStateMachine 94
B.4.16 TransitionEventType 94
B.4.17 AuditUpdateStateEventType 95
B.4.18 Special Restrictions on subtyping StateMachines 95
B.4.19 Specific StatusCodes for StateMachines 95
B.5 Examples of StateMachines in the AddressSpace 96
B.5.1 StateMachineType using inheritance 96
B.5.2 StateMachineType with a sub-machine using inheritance 97
B.5.3 StateMachineType using containment 98
B.5.4 Example of a StateMachine having Transition to SubStateMachine 99
Annex C (normative) File Transfer 101
C.1 Overview 101
C.2 File ..... **Error! Bookmark not defined.**
C.2.1 FileType 101
C.2.2 Open 102
C.2.3 Close 103
C.2.4 Read 103
C.2.5 Write 104
C.2.6 GetPosition 104
C.2.7 SetPosition 105
C.3 File System 106
C.3.1 FileDirectoryType 106
C.3.2 FileSystem Object 106
C.3.3 CreateDirectory 107
C.3.4 CreateFile 107
C.3.5 Delete 108
C.3.6 MoveOrCopy 109
C.4 Temporary File Transfer 110
C.4.1 TemporaryFileTransferType 110
C.4.2 File Transfer Sequences 110
C.4.3 GenerateFileForRead 111
C.4.4 GenerateFileForWrite 112
C.4.5 CloseAndCommit 112
C.4.6 FileTransferStateMachineType 113
C.4.7 Reset 115
Annex D (normative) DataTypeDictionary 116
D.1 Overview 116
D.2 Data Type Model 116
D.3 DataTypeDictionary, DataTypeDescription, DataTypeEncoding and DataTypeSystem 117
D.4 AddressSpace Organization 119
D.5 Node Definitions 120

AVA Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

D.5.1 HasDescription 120
D.5.2 DataTypeDictionaryType 121
D.5.3 DataTypeDescriptionType 121
D.5.4 DataTypeSystemType 122
D.5.5 OPC Binary 122
D.5.6 XML Schema 122
Annex E (normative) OPC Binary Type Description System 123
E.1 Concepts 123
E.2 Schema Description 124
E.2.1 TypeDictionary 124
E.2.2 TypeDescription 124
E.2.3 OpaqueType 124
E.2.4 EnumeratedType 125
E.2.5 StructuredType 125
E.2.6 FieldType 126
E.2.7 EnumeratedValue 127
E.2.8 ByteOrder 127
E.2.9 ImportDirective 127
E.3 Standard Type Descriptions 128
E.4 Type Description Examples 128
E.5 OPC Binary XML Schema 130
E.6 OPC Binary Standard TypeDictionary 131
Annex F (normative) User Authorization 134
F.1 Overview 134
F.2 RoleSetType ..... **Error! Bookmark not defined.**
F.2.1 AddRole Method ..... **Error! Bookmark not defined.**
F.2.2 RemoveRole Method ..... **Error! Bookmark not defined.**
F.3 RoleType ..... **Error! Bookmark not defined.**
F.3.1 IdentityMappingRuleType ..... **Error! Bookmark not defined.**
F.3.2 AddIdentity Method ..... **Error! Bookmark not defined.**
F.3.3 RemoveIdentity Method ..... **Error! Bookmark not defined.**
F.3.4 AddApplication Method ..... **Error! Bookmark not defined.**
F.3.5 RemoveApplication Method ..... **Error! Bookmark not defined.**
F.3.6 AddEndpoint Method ..... **Error! Bookmark not defined.**
F.3.7 RemoveEndpoint Method ..... **Error! Bookmark not defined.**
F.4 RoleMappingRuleChangedAuditEventType 139

FIGURES

Figure 1 – Standard AddressSpace Structure 46
Figure 2 – Views Organization 47
Figure 3 – Objects Organization 48
Figure 4 – ObjectTypes Organization 49
Figure 5 – VariableTypes Organization 50
Figure 6 – ReferenceType Definitions 51
Figure 7 – EventTypes Organization 52
Figure 8 – Excerpt of Diagnostic Information of the Server 53

AVA Documents Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

Figure B.1 – Example of a simple state machine 83
Figure B.2 – Example of a state machine having a sub-machine 83
Figure B.3 – The StateMachine Information Model 85
Figure B.6 – Example of an initial State in a sub-machine 91
Figure B.7 – Example of a StateMachineType using inheritance 96
Figure B.8 – Example of a StateMachineType with a SubStateMachine using inheritance 97
Figure B.9 – Example of a StateMachineType using containment 98
Figure B.10 – Example of a state machine with transitions from sub-states 99
Figure B.11 – Example of a StateMachineType having Transition to SubStateMachine 100
Figure C.1 – FileSystem Example 106
Figure C.2 – Read File Transfer Example Sequence 110
Figure C.3 – Write File Transfer Example Sequence 111
Figure C.4 – File Transfer States 113
Figure C.5 – FileTransferStateMachineType 114
Figure D.1 – DataType Model 116
Figure D.2 – Example of DataType Modelling 119
Figure D.3 – DataTypes Organization 120
Figure E.1 – OPC Binary Dictionary Structure 123

TABLES

Table 1 – Examples of DataTypes 2
Table 2 – Type Definition Table 3
Table 3 – Common Node Attributes 4
Table 4 – Common Object Attributes 4
Table 5 – Common Variable Attributes 5
Table 6 – Common VariableType Attributes 5
Table 7 – Common Method Attributes 5
Table 7 – BaseObjectType Definition 6
Table 8 – ServerType Definition 7
Table 9 – ServerCapabilitiesType Definition 9
Table 10 – ServerDiagnosticsType Definition 11
Table 11 – SessionsDiagnosticsSummaryType Definition 12
Table 12 – SessionDiagnosticsObjectType Definition 12
Table 13 – VendorServerInfoType Definition 13
Table 14 – ServerRedundancyType Definition 13
Table 15 – TransparentRedundancyType Definition 13
Table 16 – NonTransparentRedundancyType Definition 14
Table 17 – NonTransparentNetworkRedundancyType Definition 14
Table 18 – OperationLimitsType Definition 15
Table 19 – AddressSpaceFileType Definition 16
Table 20 – NamespaceMetadataType Definition 17
Table 21 – NamespacesType Definition 18

AVA Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

Table 22 – BaseEventType Definition 19
Table 23 – AuditEventType Definition 21
Table 24 – AuditSecurityEventType Definition 21
Table 25 – AuditChannelEventType Definition 22
Table 26 – AuditOpenSecureChannelEventType Definition 22
Table 27 – AuditSessionEventType Definition 23
Table 28 – AuditCreateSessionEventType Definition 23
Table 29 – AuditUrlMismatchEventType Definition 24
Table 30 – AuditActivateSessionEventType Definition 24
Table 31 – AuditCancelEventType Definition 25
Table 32 – AuditCertificateEventType Definition 25
Table 33 – AuditCertificateDataMismatchEventType Definition 26
Table 34 – AuditCertificateExpiredEventType Definition 26
Table 35 – AuditCertificateInvalidEventType Definition 26
Table 36 – AuditCertificateUntrustedEventType Definition 27
Table 37 – AuditCertificateRevokedEventType Definition 27
Table 38 – AuditCertificateMismatchEventType Definition 27
Table 39 – AuditNodeManagementEventType Definition 28
Table 40 – AuditAddNodesEventType Definition 28
Table 41 – AuditDeleteNodesEventType Definition 28
Table 42 – AuditAddReferencesEventType Definition 29
Table 43 – AuditDeleteReferencesEventType Definition 29
Table 44 – AuditUpdateEventType Definition 29
Table 45 – AuditWriteUpdateEventType Definition 30
Table 46 – AuditHistoryUpdateEventType Definition 30
Table 47 – AuditUpdateMethodEventType Definition 31
Table 48 – SystemEventType Definition 31
Table 49 – DeviceFailureEventType Definition 31
Table 50 – SystemStatusChangeEventType Definition 32
Table 51 – BaseModelChangeEventType Definition 32
Table 52 – GeneralModelChangeEventType Definition 32
Table 53 – SemanticChangeEventType Definition 33
Table 54 – EventQueueOverflowEventType Definition 33
Table 55 – ProgressEventType Definition 33
Table 56 – ModellingRuleType Definition 34
Table 57 – FolderType Definition 34
Table 58 – DataTypeEncodingType Definition 34
Table 59 – AggregateFunctionType Definition 35
Table 60 – BaseVariableType Definition 35
Table 61 – PropertyType Definition 35
Table 62 – BaseDataVariableType Definition 36
Table 63 – ServerVendorCapabilityType Definition 36
Table 64 – ServerStatusType Definition 37

AVA Documents
Document Sharing Hub

Table 65 – BuildInfoType Definition 37
Table 66 – ServerDiagnosticsSummaryType Definition 38
Table 67 – SamplingIntervalDiagnosticsArrayType Definition 38
Table 68 – SamplingIntervalDiagnosticsType Definition 38
Table 69 – SubscriptionDiagnosticsArrayType Definition 39
Table 70 – SubscriptionDiagnosticsType Definition 40
Table 71 – SessionDiagnosticsArrayType Definition 40
Table 72 – SessionDiagnosticsVariableType Definition 42
Table 73 – SessionSecurityDiagnosticsArrayType Definition 43
Table 74 – SessionSecurityDiagnosticsType Definition 44
Table 75 – OptionSetType Definition 44
Table 76 – SelectionListType Definition 45
Table 77 – AudioVariableType Definition 46
Table 78 – Root Definition 47
Table 79 – Views Definition 47
Table 80 – Objects Definition 48
Table 81 – Types Definition 48
Table 82 – ObjectTypes Definition 49
Table 83 – VariableTypes Definition 50
Table 84 – ReferenceTypes Definition 51
Table 85 – DataTypes Definition 51
Table 86 – EventTypes Definition 52
Table 87 – Server Definition ..... **Error! Bookmark not defined.**
Table 88 – ExposesItsArray Definition 54
Table 89 – Mandatory Definition 54
Table 90 – Optional Definition 54
Table 91 – OptionalPlaceholder Definition 54
Table 92 – MandatoryPlaceholder Definition 55
Table 93 – GetMonitoredItems Method AddressSpace Definition 55
Table 94 – ResendData Method AddressSpace Definition 56
Table 95 – SetSubscriptionDurable Method AddressSpace Definition 56
Table 96 – RequestServerStateChange Method AddressSpace Definition 57
Table 97 – References ReferenceType 58
Table 98 – HierarchicalReferences ReferenceType 58
Table 99 – NonHierarchicalReferences ReferenceType 58
Table 100 – HasChild ReferenceType 58
Table 101 – Aggregates ReferenceType 59
Table 102 – Organizes ReferenceType 59
Table 103 – HasComponent ReferenceType 59
Table 104 – HasOrderedComponent ReferenceType 59
Table 105 – HasProperty ReferenceType 60
Table 106 – HasSubtype ReferenceType 60
Table 107 – HasModellingRule ReferenceType 60

AVA Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

Table 108 – HasTypeDefinition ReferenceType 60
Table 109 – HasEncoding ReferenceType 61
Table 110 – HasEventSource ReferenceType 61
Table 111 – HasNotifier ReferenceType 61
Table 112 – GeneratesEvent ReferenceType 61
Table 113 – AlwaysGeneratesEvent ReferenceType 62
Table 114 – IEC 62541-3 DataType Definitions 63
Table 115 – BaseDataType Definition 64
Table 116 – Structure Definition 64
Table 117 – Enumeration Definition 65
Table 118 – ByteString Definition 65
Table 119 – Number Definition 65
Table 120 – Double Definition 65
Table 121 – Integer Definition 65
Table 122 – DateTime Definition 66
Table 123 – String Definition 66
Table 124 – UInteger Definition 66
Table 125 – Image Definition 66
Table 126 – UInt64 Definition 66
Table 127 – DataTypeDefinition Definition 67
Table 128 – EnumValueType Definition 67
Table 129 – IEC 62541-4 DataType Definitions 67
Table 130 – UserIdentityToken Definition 68
Table 131 – BuildInfo Structure 68
Table 132 – BuildInfo Definition 68
Table 133 – RedundancySupport Values 68
Table 134 – RedundancySupport Definition 68
Table 135 – ServerState Values 69
Table 136 – ServerState Definition 69
Table 137 – RedundantServerDataType Structure 69
Table 138 – RedundantServerDataType Definition 69
Table 139 – SamplingIntervalDiagnosticsDataType Structure 70
Table 140 – SamplingIntervalDiagnosticsDataType Definition 70
Table 141 – ServerDiagnosticsSummaryDataType Structure 70
Table 142 – ServerDiagnosticsSummaryDataType Definition 70
Table 143 – ServerStatusDataType Structure 71
Table 144 – ServerStatusDataType Definition 71
Table 145 – SessionDiagnosticsDataType Structure 71
Table 146 – SessionDiagnosticsDataType Definition 72
Table 147 – SessionSecurityDiagnosticsDataType Structure 73
Table 148 – SessionSecurityDiagnosticsDataType Definition 73
Table 149 – ServiceCounterDataType Structure 73
Table 150 – ServiceCounterDataType Definition 73

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

Table 151 – StatusResult Structure 74
Table 152 – StatusResult Definition 74
Table 153 – SubscriptionDiagnosticsDataType Structure 75
Table 154 – SubscriptionDiagnosticsDataType Definition 75
Table 155 – ModelChangeStructureDataType Structure 76
Table 156 – ModelChangeStructureDataType Definition 76
Table 157 – SemanticChangeStructureDataType Structure 76
Table 158 – SemanticChangeStructureDataType Definition 76
Table 159 – BitFieldMaskDataType Definition 77
Table 160 – NetworkGroupDataType Structure 77
Table 161 – NetworkGroupDataType Definition 77
Table 162 – EndpointUrlListDataType Structure 77
Table 163 – EndpointUrlListDataType Definition 77
Table 164 – KeyValuePair Structure 78
Table 166 – EndpointType Structure 78
Table B.1 – StateMachineType Definition 86
Table B.2 – StateVariableType Definition 86
Table B.3 – TransitionVariableType Definition 87
Table B.4 – FiniteStateMachineType Definition 88
Table B.5 – FiniteStateVariableType Definition 90
Table B.6 – FiniteTransitionVariableType Definition 90
Table B.7 – StateType Definition 91
Table B.8 – InitialStateType Definition 92
Table B.9 – TransitionType Definition 92
Table B.10 – FromState ReferenceType 92
Table B.11 – ToState ReferenceType 93
Table B.12 – HasCause ReferenceType 93
Table B.13 – HasEffect ReferenceType 94
Table B.14 – HasSubStateMachine ReferenceType 94
Table B.15 – TransitionEventType 94
Table B.16 – AuditUpdateStateEventType 95
Table B.17 – Specific StatusCodes for StateMachines 95
Table C.1 – FileType 101
Table C.2 – Open Method AddressSpace Definition 103
Table C.3 – Close Method AddressSpace Definition 103
Table C.4 – Read Method AddressSpace Definition 104
Table C.5 – Write Method AddressSpace Definition 104
Table C.6 – GetPosition Method AddressSpace Definition 105
Table C.7 – SetPosition Method AddressSpace Definition 105
Table C.8 – FileDirectoryType 106
Table C.9 – CreateDirectory Method AddressSpace Definition 107
Table C.10 – CreateFile Method AddressSpace Definition 108
Table C.11 – Delete Method AddressSpace Definition 109

AVA Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

Table C.12 – MoveOrCopy Method AddressSpace Definition 109
Table C.13 – TemporaryFileTransferType 110
Table C.14 – GenerateFileForRead Method AddressSpace Definition 112
Table C.15 – GenerateFileForWrite Method AddressSpace Definition 112
Table C.16 – CloseAndCommit Method AddressSpace Definition 113
Table C.17 – FileTransferStateMachineType 114
Table C.18 – FileTransferStateMachineType transitions 115
Table D.1 – HasDescription ReferenceType 121
Table D.2 – DataTypeDictionaryType Definition 121
Table D.3 – DataTypeDescriptionType Definition 121
Table D.4 – DataTypeSystemType Definition 122
Table D.5 – OPC Binary Definition 122
Table D.6 – XML Schema Definition 122
Table E.1 – TypeDictionary Components 124
Table E.2 – TypeDescription Components 124
Table E.3 – OpaqueType Components 125
Table E.4 – EnumeratedType Components 125
Table E.5 – StructuredType Components 126
Table E.6 – FieldType Components 126
Table E.7 – EnumeratedValue Components 127
Table E.8 – ImportDirective Components 128
Table E.9 – Standard Type Descriptions 128
Table 18 – RoleSetType Definition ..... **Error! Bookmark not defined.**
Table 19 – RoleType Definition ..... **Error! Bookmark not defined.**
Table 20 – IdentityMappingRuleType ..... **Error! Bookmark not defined.**
Table 21 – RoleMappingRuleChangedAuditEventType Definition 140

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

OPC UNIFIED ARCHITECTURE –

Part 5: Information Model

FOREWORD

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

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

IEC 62541-5 has been prepared by subcommittee 65E: Devices and integration in enterprise systems, of IEC technical committee 65: Industrial-process measurement, control and automation.

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

Enquiry draft	Report on voting
65E/XX/DTR	65E/XX/RVC

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

This third edition cancels and replaces the second edition of IEC 62541, published in 2015.

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

a) Added Annex F on User Authentication. Describes the Role Information Model that also allows configuration of Roles.
b) Added new data types: “Union”, “Decimal”, “OptionSet”, “DateString”, “TimeString”, “DurationString”, NormalizedString”, “DecimalString”, and “AudioDataType”.

AVA Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

c) Added Method to request a state change in a Server.

d) Added Method to set Subscription to persistent mode.

e) Added Method to request resending od data from a Subscription.

f) Added concept allowing to temporary create file to write to or read from server in C.4.

g) Added new Variable type tosupport Selection Lists

h) Added optional properties to FiniteStateMachineType to expose currently available states and transitions.

i) Added UrisVersion Property to ServerType. This version information can be used for session-less service invocation

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

Throughout this document and the referenced 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, Part 5*, 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.

The National Committees are requested to note that for this publication the stability date is 2021.

THIS TEXT IS INCLUDED FOR THE INFORMATION OF THE NATIONAL COMMITTEES AND WILL BE DELETED AT THE PUBLICATION STAGE.

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

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

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

OPC Unified Architecture, Part 5 Specification

Part 5: Information Model

1 Scope

This specification defines the Information Model of the OPC Unified Architecture. The Information Model describes standardised *Nodes* of a *Server*'s *AddressSpace*. These *Nodes* are standardised types as well as standardised instances used for diagnostics or as entry points to server-specific *Nodes*. Thus, the Information Model defines the *AddressSpace* of an empty OPC UA *Server*. However, it is not expected that all *Servers* will provide all of these *Nodes*.

2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. 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 62541-6, *OPC Unified Architecture – Part 6: Mappings*

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

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

IEC 62541-10, *OPC Unified Architecture – Part 10: Programs*

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

RFC 2045: Multipurpose Internet Mail Extensions (MIME) Part One:
http://www.ietf.org/rfc/rfc2045.txt

RFC 2046: Multipurpose Internet Mail Extensions (MIME) Part Two:
https://www.ietf.org/rfc/rfc2046.txt

RFC 2047: Multipurpose Internet Mail Extensions (MIME) Part Three:
http://www.ietf.org/rfc/rfc2047.txt

ISO/IEC/IEEE 60559:2011: Information technology – Microprocessor Systems – Floating-Point arithmetic
https://www.iso.org/standard/57469.html

XML Schema Part 1: Structures
http://www.w3.org/TR/xmlschema-1/

XML Schema Part 2: Datatypes
http://www.w3.org/TR/xmlschema-2/

XPATH: XML Path Language
http://www.w3.org/TR/xpath/

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

UTF-8: UTF-8, a transformation format of ISO 10646

http://www.ietf.org/rfc/rfc3629.txt

3 Terms, definitions and conventions

3.1 Terms and definitions

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

3.1.1 ClientUserId

String that identifies the user of the client requesting an action

Note 1 to entry: The *ClientUserId* is obtained directly or indirectly from the *UserIdentityToken* passed by the *Client* in the *ActivateSession Service* call. See 6.4.3 for details.

3.2 Abbreviations and symbols

UA Unified Architecture

XML Extensible Markup Language

3.3 Conventions for Node descriptions

Node definitions are specified using tables (see Table 2).

Attributes are defined by providing the *Attribute* name and a value, or a description of the value.

References are defined by providing the *ReferenceType* name, the *BrowseName* of the *TargetNode* and its *NodeClass*.

- If the *TargetNode* is a component of the *Node* being defined in the table the *Attributes* of the composed *Node* are defined in the same row of the table.
- The *DataType* is only specified for *Variables*; "[<number>]" indicates a single-dimensional array, for multi-dimensional arrays the expression is repeated for each dimension (e.g. [2][3] for a two-dimensional array). For all arrays the *ArrayDimensions* is set as identified by <number> values. If no <number> is set, the corresponding dimension is set to 0, indicating an unknown size. If no number is provided at all the *ArrayDimensions* can be omitted. If no brackets are provided, it identifies a scalar *DataType* and the *ValueRank* is set to the corresponding value (see IEC 62541-3). In addition, *ArrayDimensions* is set to null or is omitted. If it can be Any or ScalarOrOneDimension, the value is put into "{<value>}", so either "{Any}" or "{ScalarOrOneDimension}" and the *ValueRank* is set to the corresponding value (see IEC 62541-3) and the *ArrayDimensions* is set to null or is omitted. Examples are given in Table 1.

Table 1 – Examples of DataTypes

Notation	Data-Type	Value-Rank	Array-Dimensions	Description
Int32	Int32	−1	omitted or null	A scalar Int32.
Int32[]	Int32	1	omitted or {0}	Single-dimensional array of Int32 with an unknown size.
Int32[][]	Int32	2	omitted or {0,0}	Two-dimensional array of Int32 with unknown sizes for both dimensions.
Int32[3][]	Int32	2	{3,0}	Two-dimensional array of Int32 with a size of 3 for the first dimension and an unknown size for the second dimension.
Int32[5][3]	Int32	2	{5,3}	Two-dimensional array of Int32 with a size of 5 for the first dimension and a size of 3 for the second dimension.
Int32{Any}	Int32	−2	omitted or null	An Int32 where it is unknown if it is scalar or array with any number of dimensions.
Int32{ScalarOrOneDimension}	Int32	−3	omitted or null	An Int32 where it is either a single-dimensional array or a scalar.

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

- The TypeDefinition is specified for *Objects* and *Variables*.
- The TypeDefinition column specifies a symbolic name for a *NodeId*, i.e. the specified *Node* points with a *HasTypeDefinition Reference* to the corresponding *Node*.
- The *ModellingRule* of the referenced component is provided by specifying the symbolic name of the rule in the *ModellingRule* column. In the *AddressSpace*, the *Node* shall use a *HasModellingRule Reference* to point to the corresponding *ModellingRule Object*.

If the *NodeId* of a *DataType* is provided, the symbolic name of the *Node* representing the *DataType* shall be used.

Nodes of all other *NodeClasses* cannot be defined in the same table; therefore only the used *ReferenceType*, their *NodeClass* and their *BrowseName* are specified. A reference to another part of this document points to their definition.

Table 2 illustrates the table. If no components are provided, the DataType, TypeDefinition and ModellingRule columns may be omitted and only a Comment column is introduced to point to the *Node* definition.

Table 2 – Type Definition Table

Attribute	**Value**				
Attribute name	Attribute value. If it is an optional Attribute that is not set "--" will be used.				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
ReferenceType name	*NodeClass* of the Target*Node*.	*BrowseName* of the target *Node*. If the *Reference* is to be instantiated by the server, then the value of the target Node's BrowseName is "--".	*DataType* of the referenced *Node*, only applicable for *Variables*.	*TypeDefinition* of the referenced *Node*, only applicable for *Variables* and *Objects*.	Referenced *ModellingRule* of the referenced *Object*.
NOTE Notes referencing footnotes of the table content.					

Components of *Nodes* can be complex that is containing components by themselves. The *TypeDefinition*, *NodeClass*, *DataType* and *ModellingRule* can be derived from the type definitions, and the symbolic name can be created as defined in 4.1. Therefore those containing components are not explicitly specified; they are implicitly specified by the type definitions.

4 NodeIds and BrowseNames

4.1 NodeIds

The *NodeIds* of all *Nodes* described in this standard are only symbolic names. IEC 62541-6 defines the actual *NodeIds*.

The symbolic name of each *Node* defined in this standard is its *BrowseName*, or, when it is part of another *Node*, the *BrowseName* of the other *Node*, a ".", and the *BrowseName* of itself. In this case "part of" means that the whole has a *HasProperty* or *HasComponent Reference* to its part. Since all *Nodes* not being part of another *Node* have a unique name in this standard, the symbolic name is unique. For example, the *ServerType* defined in 6.3.1 has the symbolic name "ServerType". One of its *InstanceDeclarations* would be identified as "ServerType.ServerCapabilities". Since this *Object* is complex, another *InstanceDeclaration* of the *ServerType* is "ServerType.ServerCapabilities.MinSupportedSampleRate". The *Server Object* defined in 8.3.2 is based on the ServerType and has the symbolic name "Server". Therefore, the instance based on the *InstanceDeclaration* described above has the symbolic name "Server.ServerCapabilities.MinSupportedSampleRate".

The NamespaceIndex for all *NodeIds* defined in this standard is 0. The namespace for this NamespaceIndex is specified in IEC 62541-3.

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

Note that this standard not only defines concrete *Nodes*, but also requires that some Nodes have to be generated, for example one for each Session running on the *Server*. The *NodeIds* of those *Nodes* are server-specific, including the Namespace. However the NamespaceIndex of those *Nodes* cannot be the NamespaceIndex 0, because they are not defined by the OPC Foundation but generated by the *Server*.

4.2 BrowseNames

The text part of the *BrowseNames* for all *Nodes* defined in this standard is specified in the tables defining the *Nodes*. The NamespaceIndex for all *BrowseNames* defined in this standard is 0.

5 Common Attributes

5.1 General

For all *Nodes* specified in this standard, the *Attributes* named in Table 3 shall be set as specified in Table 3.

Table 3 – Common Node Attributes

Attribute	Value
DisplayName	The *DisplayName* is a *LocalizedText*. Each server shall provide the *DisplayName* identical to the *BrowseName* of the *Node* for the LocaleId “en”. Whether the server provides translated names for other LocaleIds is server-specific.
Description	Optionally a server-specific description is provided.
NodeClass	Shall reflect the *NodeClass* of the *Node*.
NodeId	The *NodeId* is described by *BrowseNames* as defined in 4.1 and defined in IEC 62541-6.
WriteMask	Optionally the *WriteMask Attribute* can be provided. If the *WriteMask Attribute* is provided, it shall set all non-server-specific *Attributes* to not writable. For example, the *Description Attribute* may be set to writable since a *Server* may provide a server-specific description for the *Node*. The *NodeId* shall not be writable, because it is defined for each *Node* in this standard.
UserWriteMask	Optionally the *UserWriteMask Attribute* can be provided. The same rules as for the *WriteMask Attribute* apply.
RolePermissions	Optionally server-specific role permissions can be provided.
UserRolePermissions	Optionally the role permissions of the current Session can be provided. The value is server-specifc and depend on the *RolePermissions Attribute* (if provided) and the current *Session*.
AccessRestrictions	Optionally server-specific access restrictions can be provided.

5.2 Objects

For all *Objects* specified in this standard, the *Attributes* named in Table 4 shall be set as specified in Table 4.

Table 4 – Common Object Attributes

Attribute	Value
EventNotifier	Whether the *Node* can be used to subscribe to *Events* or not is server-specific.

5.3 Variables

For all *Variables* specified in this standard, the *Attributes* named in Table 5 shall be set as specified in Table 5.

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

Table 5 – Common Variable Attributes

Attribute	Value
MinimumSamplingInterval	Optionally, a server-specific minimum sampling interval is provided.
AccessLevel	The access level for *Variables* used for type definitions is server-specific, for all other *Variables* defined in this standard, the access level shall allow reading; other settings are server-specific.
UserAccessLevel	The value for the *UserAccessLevel Attribute* is server-specific. It is assumed that all *Variables* can be accessed by at least one user.
Value	For *Variables* used as *InstanceDeclarations,* the value is server-specific; otherwise it shall represent the value described in the text.
ArrayDimensions	If the *ValueRank* does not identify an array of a specific dimension (i.e. *ValueRank* <= 0) the *ArrayDimensions* can either be set to null or the *Attribute* is missing. This behaviour is server-specific. If the *ValueRank* specifies an array of a specific dimension (i.e. *ValueRank* > 0) then the *ArrayDimensions Attribute* shall be specified in the table defining the *Variable*.
Historizing	The value for the *Historizing Attribute* is server-specific.
AccessLevelEx	If the *AccessLevelEx Attribute* is provided, it shall have the bits 8, 9, and 10 set to 0, meaning that read and write operations on an individual *Variable* are atomic, and arrays can be partly written.

5.4 VariableTypes

For all *VariableTypes* specified in this standard, the *Attributes* named in Table 6 shall be set as specified in Table 6.

Table 6 – Common VariableType Attributes

Attributes	Value
Value	Optionally a server-specific default value can be provided.
ArrayDimensions	If the *ValueRank* does not identify an array of a specific dimension (i.e. *ValueRank* <= 0) the *ArrayDimensions* can either be set to null or the *Attribute* is missing. This behaviour is server-specific. If the *ValueRank* specifies an array of a specific dimension (i.e. *ValueRank* > 0) then the *ArrayDimensions Attribute* shall be specified in the table defining the *VariableType*.

5.5 Methods

For all *Methods* specified in this standard, the *Attributes* named in Table 7 shall be set as specified in Table 7.

Table 7 – Common Method Attributes

Attributes	Value
Executable	All *Methods* defined in this specification shall be executable (*Executable Attribute* set to “True”), unless it is defined differently in the *Method* definition.
UserExecutable	The value of the *UserExecutable Attribute* is server-specific. It is assumed that all *Methods* can be executed by at least one user.

6 Standard ObjectTypes

6.1 General

Typically, the components of an *ObjectType* are fixed and can be extended by subtyping. However, since each *Object* of an *ObjectType* can be extended with additional components, this standard allows extending the standard *ObjectTypes* defined in this document with additional components. Thereby, it is possible to express the additional information in the type definition that would already be contained in each *Object*. Some *ObjectTypes* already provide entry points for server-specific extensions. However, it is not allowed to restrict the components of the standard *ObjectTypes* defined in this standard. An example of extending the *ObjectTypes* is putting the standard *Property NodeVersion* defined in IEC 62541-3 into the *BaseObjectType*, stating that each *Object* of the *Server* will provide a *NodeVersion*.

6.2 BaseObjectType

The *BaseObjectType* is used as type definition whenever there is an *Object* having no more concrete type definitions available. *Servers* should avoid using this *ObjectType* and use a more specific type, if possible. This *ObjectType* is the base *ObjectType* and all other

AVA Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

ObjectTypes shall either directly or indirectly inherit from it. However, it might not be possible for *Servers* to provide all *HasSubtype References* from this *ObjectType* to its subtypes, and therefore it is not required to provide this information.

There are no *References* except for *HasSubtype References* specified for this *ObjectType*. It is formally defined in Table 8.

Table 8 – BaseObjectType Definition

Attribute	**Value**				
BrowseName	BaseObjectType				
IsAbstract	False				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**Modelling Rule**
HasSubtype	ObjectType	ServerType	Defined in 6.3.1		
HasSubtype	ObjectType	ServerCapabilitiesType	Defined in 6.3.2		
HasSubtype	ObjectType	ServerDiagnosticsType	Defined in 6.3.3		
HasSubtype	ObjectType	SessionsDiagnosticsSummaryType	Defined in 6.3.4		
HasSubtype	ObjectType	SessionDiagnosticsObjectType	Defined in 6.3.5		
HasSubtype	ObjectType	VendorServerInfoType	Defined in 6.3.6		
HasSubtype	ObjectType	ServerRedundancyType	Defined in 6.3.7		
HasSubtype	ObjectType	BaseEventType	Defined in 6.4.2		
HasSubtype	ObjectType	ModellingRuleType	Defined in 6.5		
HasSubtype	ObjectType	FolderType	Defined in 6.6		
HasSubtype	ObjectType	DataTypeEncodingType	Defined in 6.7		

6.3 ObjectTypes for the Server Object

6.3.1 ServerType

This *ObjectType* defines the capabilities supported by the OPC UA *Server*. It is formally defined in Table 9.

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

178 **Table 9 – ServerType Definition**

Attribute	Value			
BrowseName	ServerType			
IsAbstract	False			
References	**NodeClass**	**BrowseName**	**DataType / TypeDefinition**	**Modelling Rule**
Subtype of the BaseObjectType defined in 6.2				
HasProperty	Variable	ServerArray	String[] PropertyType	Mandatory
HasProperty	Variable	NamespaceArray	String[] PropertyType	Mandatory
HasProperty	Variable	UrisVersion	VersionTime PropertyType	Optional
HasComponent	Variable	ServerStatus[1]	ServerStatusDataType ServerStatusType	Mandatory
HasProperty	Variable	ServiceLevel	Byte PropertyType	Mandatory
HasProperty	Variable	Auditing	Boolean PropertyType	Mandatory
HasProperty	Variable	EstimatedReturnTime	DateTime PropertyType	Optional
HasProperty	Variable	LocalTime	TimeZoneDataType PropertyType	Optional
HasComponent	Object	ServerCapabilities[1]	- ServerCapabilitiesType	Mandatory
HasComponent	Object	ServerDiagnostics[1]	- ServerDiagnosticsType	Mandatory
HasComponent	Object	VendorServerInfo	- VendorServerInfoType	Mandatory
HasComponent	Object	ServerRedundancy[1]	- ServerRedundancyType	Mandatory
HasComponent	Object	Namespaces	- NamespacesType	Optional
HasComponent	Method	GetMonitoredItems	Defined in 9.1	Optional
HasComponent	Method	ResendData	Defined in 9.2	Optional
HasComponent	Method	SetSubscriptionDurable	Defined in 9.3	Optional
HasComponent	Method	RequestServerStateChange	Defined in 9.4	Optional
NOTE Containing *Objects* and *Variables* of these *Objects* and *Variables* are defined by their *BrowseName* defined in the corresponding *TypeDefinitionNode*. The *NodeId* is defined by the composed symbolic name described in 4.1.				

179

180 *ServerArray* defines an array of *Server* URIs. This *Variable* is also referred to as the *server*
181 *table*. Each URI in this array represents a globally-unique logical name for a *Server* within the
182 scope of the network in which it is installed. Each OPC UA *Server* instance has a single URI
183 that is used in the *server table* of other OPC UA *Servers*. Index 0 is reserved for the URI of
184 the local *Server*. Values above 0 are used to identify remote *Servers* and are specific to a
185 *Server*. IEC 62541-4 describes discovery mechanism that can be used to resolve URIs into
186 URLs. The *Server* URI is case sensitive.

187 The URI of the *ServerArray* with Index 0 shall be identical to the URI of the *NamespaceArray*
188 with Index 1, since both represent the local *Server*.

189 The indexes into the *server table* are referred to as *server indexes* or *server names*. They are
190 used in OPC UA *Services* to identify *TargetNodes* of *References* that reside in remote
191 *Servers*. Clients may read the entire table or they may read individual entries in the table. The
192 *Server* shall not modify or delete entries of this table while any client has an open session to
193 the *Server*, because clients may cache the *server table*. A *Server* may add entries to the
194 *server table* even if clients are connected to the *Server*.

195 *NamespaceArray* defines an array of namespace URIs. This *Variable* is also referred as
196 *namespace table*. The indexes into the *namespace table* are referred to as
197 *NamespaceIndexes*. *NamespaceIndexes* are used in *NodeIds* in OPC UA *Services*, rather
198 than the longer namespace URI. Index 0 is reserved for the OPC UA namespace, and index 1
199 is reserved for the local *Server*. Clients may read the entire *namespace table* or they may
200 read individual entries in the *namespace table*. The *Server* shall not modify or delete entries
201 of the *namespace table* while any client has an open session to the *Server*, because clients

AVA Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

may cache the *namespace table*. A *Server* may add entries to the *namespace table* even if clients are connected to the *Server*. It is recommended that *Servers* not change the indexes of the *namespace table* but only add entries, because the client may cache *NodeIds* using the indexes. Nevertheless, it might not always be possible for *Servers* to avoid changing indexes in the *namespace table*. Clients that cache *NamespaceIndexes* of *NodeIds* should always check when starting a session to verify that the cached *NamespaceIndexes* have not changed.

UrisVersion defines the version of the *ServerArray* and the *NamespaceArray*. Everytime the *ServerArray* or the *NamespaceArray* is changed, the value of the *UrisVersion* shall be updated to a value greater than the previous value. The *UrisVersion Property* is used in combination with the *SessionlessInvoke Service* defined in IEC 62541-4. If a *Server* supports this *Service*, the *Server* shall support this *Property*. It is the responsibility of the *Server* to provide a consistent set of values for the *ServerArray*, *NamespaceArray* and the *UrisVersion Properties*. The *VersionTime DataType* is defined in IEC 62541-4.

ServerStatus contains elements that describe the status of the *Server*. See 12.10 for a description of its elements.

ServiceLevel describes the ability of the *Server* to provide its data to the client. The value range is from 0 to 255, where 0 indicates the worst and 255 indicates the best. IEC 62541-4 defines required sub-ranges for different scenarios. The intent is to provide the clients an indication of availability among redundant *Servers*.

Auditing is a Boolean specifying if the *Server* is currently generating audit events. It is set to TRUE if the *Server* generates audit events, otherwise to false. The *Profiles* defined in IEC 62541-7 specify what kind of audit events are generated by the *Server*.

EstimatedReturnTime indicates the time at which the *Server* is expected to have a *ServerStatus.State* of RUNNING_0. A *Client* that observes a shutdown or a *ServiceLevel* of 0 should either wait until after this time to attempt to reconnect to this *Server* or enter into slow retry logic. For example, most *Clients* will attempt to reconnect after a failure immediately and then progressively increase the delay between attempts until some maximum delay. This time can be used to trigger the *Client* to start its reconnect logic with some delay.

LocalTime is a structure containing the Offset and the DaylightSavingInOffset flag. The Offset specifies the time difference (in minutes) between the *Server* time in UTC and the local time at the *Server* location. If DaylightSavingInOffset is TRUE, then Standard/Daylight savings time (DST) at the *Server* location is in effect and Offset includes the DST correction. If FALSE then the Offset does not include DST correction and DST may or may not be in effect.

ServerCapabilities defines the capabilities supported by the OPC UA *Server*. See 6.3.2 for its description.

ServerDiagnostics defines diagnostic information about the OPC UA *Server*. See 6.3.3 for its description.

VendorServerInfo represents the browse entry point for vendor-defined *Server* information. This *Object* is required to be present even if there are no vendor-defined *Objects* beneath it. See 6.3.6 for its description.

ServerRedundancy describes the redundancy capabilities provided by the *Server*. This *Object* is required even if the *Server* does not provide any redundancy support. If the *Server* supports redundancy, then a subtype of *ServerRedundancyType* is used to describe its capabilities. Otherwise, it provides an *Object* of type *ServerRedundancyType* with the *Property* RedundancySupport set to none. See 6.3.7 for the description of *ServerRedundancyType*.

Namespaces provides a list of *NamespaceMetadataType Objects* with additional information about the namespaces used in the *Server*. See 6.3.14 for the description of *NamespaceMetadataType*.

The *GetMonitoredItems Method* is used to identify the *MonitoredItems* of a *Subscription*. It is defined in 9.1; the intended usage is defined in IEC 62541-4.

AVA Documents Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

The *ResendData Method* is used to get the latest values of the data monitored items of a *Subscription*. It is defined in 9.2; the intended usage is defined in IEC 62541-4.

The *SetSubscriptionDurable Method* is used to set a *Subscription* into a mode where *MonitoredItem* data and event queues are stored and delivered even if an OPC UA *Client* was disconnected for a longer time or the OPC UA *Server* was restarted. It is defined in 9.3; the intended usage is defined in IEC 62541-4.

The *RequestServerStateChange Method* allows a *Client* to request a state change in the *Server*. It is defined in 9.4; the intended usage is defined in IEC 62541-4.

6.3.2 ServerCapabilitiesType

This *ObjectType* defines the capabilities supported by the OPC UA *Server*. It is formally defined in Table 10.

Table 10 – ServerCapabilitiesType Definition

Attribute	**Value**			
BrowseName	ServerCapabilitiesType			
IsAbstract	False			
References	**NodeClass**	**BrowseName**	**DataType / TypeDefinition**	**ModellingRule**
Subtype of the BaseObjectType defined in 6.2				
HasProperty	Variable	ServerProfileArray	String[] PropertyType	Mandatory
HasProperty	Variable	LocaleIdArray	LocaleId[] PropertyType	Mandatory
HasProperty	Variable	MinSupportedSampleRate	Duration PropertyType	Mandatory
HasProperty	Variable	MaxBrowseContinuationPoints	UInt16 PropertyType	Mandatory
HasProperty	Variable	MaxQueryContinuationPoints	UInt16 PropertyType	Mandatory
HasProperty	Variable	MaxHistoryContinuationPoints	UInt16 PropertyType	Mandatory
HasProperty	Variable	SoftwareCertificates	SignedSoftwareCertificate[] PropertyType	Mandatory
HasProperty	Variable	MaxArrayLength	UInt32 PropertyType	Optional
HasProperty	Variable	MaxStringLength	UInt32 PropertyType	Optional
HasProperty	Variable	MaxByteStringLength	UInt32 PropertyType	Optional
HasComponent	Object	OperationLimits	-- OperationLimitsType	Optional
HasComponent	Object	ModellingRules	-- FolderType	Mandatory
HasComponent	Object	AggregateFunctions	-- FolderType	Mandatory
HasComponent	Object	RoleSet	RoleSetType	Optional

ServerProfileArray lists the *Profiles* that the *Server* supports. See IEC 62541-7 for the definitions of *Server Profiles*. This list should be limited to the *Profiles* the *Server* supports in its current configuration.

LocaleIdArray is an array of LocaleIds that are known to be supported by the *Server*. The *Server* might not be aware of all LocaleIds that it supports because it may provide access to underlying servers, systems or devices that do not report the LocaleIds that they support.

MinSupportedSampleRate defines the minimum supported sample rate, including 0, which is supported by the *Server*.

MaxBrowseContinuationPoints is an integer specifying the maximum number of parallel continuation points of the Browse *Service* that the *Server* can support per session. The value specifies the maximum the *Server* can support under normal circumstances, so there is no

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

guarantee the *Server* can always support the maximum. The client should not open more Browse calls with open continuation points than exposed in this *Variable*. The value 0 indicates that the *Server* does not restrict the number of parallel continuation points the client should use.

MaxQueryContinuationPoints is an integer specifying the maximum number of parallel continuation points of the QueryFirst *Services* that the *Server* can support per session. The value specifies the maximum the *Server* can support under normal circumstances, so there is no guarantee the *Server* can always support the maximum. The client should not open more QueryFirst calls with open continuation points than exposed in this *Variable*. The value 0 indicates that the *Server* does not restrict the number of parallel continuation points the client should use.

MaxHistoryContinuationPoints is an integer specifying the maximum number of parallel continuation points of the HistoryRead *Services* that the *Server* can support per session. The value specifies the maximum the *Server* can support under normal circumstances, so there is no guarantee the *Server* can always support the maximum. The client should not open more HistoryRead calls with open continuation points than exposed in this *Variable*. The value 0 indicates that the *Server* does not restrict the number of parallel continuation points the client should use.

SoftwareCertificates is an array of *SignedSoftwareCertificates* containing all *SoftwareCertificates* supported by the *Server*. A *SoftwareCertificate* identifies capabilities of the *Server*. It contains the list of *Profiles* supported by the *Server*. *Profiles* are described in IEC 62541-7.

The *MaxArrayLength Property* indicates the maximum length of a one or multidimensional array supported by Variables of the *Server*. In a multidimensional array it indicates the overall length. For example, a three-dimensional array of 2x3x10 has the array length of 60. The *Server* might further restrict the length for individual Variables without notice to the client. *Servers* may use the *Property MaxArrayLength* defined in IEC 62541-3 on individual *DataVariables* to specify the size on individual values. The individual *Property* may have a larger or smaller value than *MaxArrayLength*.

The *MaxStringLength Property* indicates the maximum number of bytes in *Strings* supported by *Variables* of the *Server*. *Servers* may override this setting by adding the *MaxStringLength Property* defined in IEC 62541-3 to an individual *DataVariable*. If a *Server* does not impose a maximum number of bytes or is not able to determine the maximum number of bytes this *Property* shall not be provided.

The *MaxByteStringLength Property* indicates the maximum number of bytes in a *ByteString* supported by *Variables* of the *Server*. It also specifies the default maximum size of a *FileType Object's* read and write buffers. *Servers* may override this setting by adding the *MaxByteStringLength Property* defined in IEC 62541-3 to an individual *DataVariable* or *FileType Object.* If a *Server* does not impose a maximum number of bytes or is not able to determine the maximum number of bytes this *Property* shall not be provided.

OperationLimits is an entry point to access information on operation limits of the *Server*, for example the maximum length of an array in a read *Service* call.

ModellingRules is an entry point to browse to all *ModellingRules* supported by the *Server*. All *ModellingRules* supported by the *Server* should be able to be browsed starting from this *Object*.

AggregateFunctions is an entry point to browse to all *AggregateFunctions* supported by the *Server*. All *AggregateFunctions* supported by the *Server* should be able to be browsed starting from this *Object*. AggregateFunctions are Objects of *AggregateFunctionType*.

The *RoleSet Object* is used to publish all *Roles* supported by the *Server*. The *RoleSetType* is specified in F.2.

When vendors expose their own capabilities they should add additional *Nodes* to the standard *ServerCapabilities Object* instance.

AVA Documents Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

6.3.3 ServerDiagnosticsType

This *ObjectType* defines diagnostic information about the OPC UA *Server*. This *ObjectType* is formally defined in Table 11.

Table 11 – ServerDiagnosticsType Definition

Attribute	Value			
BrowseName	ServerDiagnosticsType			
IsAbstract	False			
References	**Node Class**	**BrowseName**	**DataType / TypeDefinition**	**Modelling Rule**
Subtype of the BaseObjectType defined in 6.2				
HasComponent	Variable	ServerDiagnosticsSummary	ServerDiagnosticsSummaryDataType ServerDiagnosticsSummaryType	Mandatory
HasComponent	Variable	SamplingIntervalDiagnosticsArray	SamplingIntervalDiagnosticsDataType[] SamplingIntervalDiagnosticsArrayType	Optional
HasComponent	Variable	SubscriptionDiagnosticsArray	SubscriptionDiagnosticsDataType[] SubscriptionDiagnosticsArrayType	Mandatory
HasComponent	Object	SessionsDiagnosticsSummary	-- SessionsDiagnosticsSummaryType	Mandatory
HasProperty	Variable	EnabledFlag	Boolean PropertyType	Mandatory

ServerDiagnosticsSummary contains diagnostic summary information for the *Server*, as defined in 12.9.

SamplingIntervalDiagnosticsArray is an array of diagnostic information per sampling rate as defined in 12.8. There is one entry for each sampling rate currently used by the *Server*. Its *TypeDefinitionNode* is the *VariableType SamplingIntervalDiagnosticsArrayType*, providing a *Variable* for each entry in the array, as defined in 7.9.

The sampling interval diagnostics are only collected by *Servers* which use a fixed set of sampling intervals. In these cases, length of the array and the set of contained *Variables* will be determined by the *Server* configuration and the *NodeId* assigned to a given sampling interval diagnostics variable shall not change as long as the *Server* configuration does not change. A *Server* may not expose the SamplingIntervalDiagnosticsArray if it does not use fixed sampling rates.

SubscriptionDiagnosticsArray is an array of Subscription diagnostic information per subscription, as defined in 12.15. There is one entry for each Notification channel actually established in the *Server*. Its *TypeDefinitionNode* is the *VariableType* SubscriptionDiagnosticsArrayType, providing a *Variable* for each entry in the array as defined in 7.11. Those *Variables* are also used as *Variables* referenced by other *Variables*.

SessionsDiagnosticsSummary contains diagnostic information per session, as defined in 6.3.4.

EnabledFlag identifies whether or not diagnostic information is collected by the *Server*. It can also be used by a client to enable or disable the collection of diagnostic information of the *Server*. The following settings of the Boolean value apply: TRUE indicates that the *Server* collects diagnostic information, and setting the value to TRUE leads to resetting and enabling the collection. FALSE indicates that no diagnostic information is collected, and setting the value to FALSE disables the collection without resetting the diagnostic values.

When diagnostics are turned off, the *Server* can return Bad_NodeIdUnknown for all static diagnostic *Nodes* except the *EnabledFlag Property*. Dynamic diagnostic *Nodes* (such as the *Session Nodes*) will not appear in the *AddressSpace*.

If collection of diagnostic information is not supported at all, the EnabledFlag Property will be read only.

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

6.3.4 SessionsDiagnosticsSummaryType

This *ObjectType* defines diagnostic information about the sessions of the OPC UA *Server*. This *ObjectType* is formally defined in Table 12.

Table 12 – SessionsDiagnosticsSummaryType Definition

Attribute		Value		
BrowseName		SessionsDiagnosticsSummaryType		
IsAbstract		False		
References	**NodeClass**	**BrowseName**	**DataType / TypeDefinition**	**Modelling Rule**
Subtype of the BaseObjectType defined in 6.2				
HasComponent	Variable	SessionDiagnosticsArray	SessionDiagnosticsDataType[] SessionDiagnosticsArrayType	Mandatory
HasComponent	Variable	SessionSecurityDiagnosticsArray	SessionSecurityDiagnosticsDataType[] SessionSecurityDiagnosticsArrayType	Mandatory
HasComponent	Object	<ClientName>	-- SessionDiagnosticsObjectType	Optional Placeholder
NOTE This row represents no *Node* in the *AddressSpace*. It is a placeholder pointing out that instances of the *ObjectType* will have those *Objects*.				

SessionDiagnosticsArray provides an array with an entry for each session in the *Server* having general diagnostic information about a session.

SessionSecurityDiagnosticsArray provides an array with an entry for each active session in the *Server* having security-related diagnostic information about a session. Since this information is security-related, it should not be made accessible to all users, but only to authorised users.

For each session of the *Server*, this *Object* also provides an *Object* representing the session, indicated by *<ClientName>*. The BrowseName could be derived from the *sessionName* defined in the *CreateSession Service* (IEC 62541-4) or some other server-specific mechanisms. It is of the *ObjectType* SessionDiagnosticsObjectType, as defined in 6.3.5.

6.3.5 SessionDiagnosticsObjectType

This *ObjectType* defines diagnostic information about a session of the OPC UA *Server*. This *ObjectType* is formally defined in Table 13.

Table 13 – SessionDiagnosticsObjectType Definition

Attribute		Value		
BrowseName		SessionDiagnosticsObjectType		
IsAbstract		False		
References	**NodeClass**	**BrowseName**	**DataType / TypeDefinition**	**Modelling Rule**
Subtype of the BaseObjectType defined in 6.2				
HasComponent	Variable	SessionDiagnostics	SessionDiagnosticsDataType SessionDiagnosticsVariableType	Mandatory
HasComponent	Variable	SessionSecurityDiagnostics	SessionSecurityDiagnosticsDataType SessionSecurityDiagnosticsType	Mandatory
HasComponent	Variable	SubscriptionDiagnosticsArray	SubscriptionDiagnosticsDataType[] SubscriptionDiagnosticsArrayType	Mandatory

SessionDiagnostics contains general diagnostic information about the session; the *SessionSecurityDiagnostics Variable* contains security-related diagnostic information. Because the information of the second *Variable* is security-related, it should not be made accessible to all users, but only to authorised users.

SubscriptionDiagnosticsArray is an array of Subscription diagnostic information per opened subscription, as defined in 12.15. Its *TypeDefinitionNode* is the *VariableType* SubscriptionDiagnosticsArrayType providing a *Variable* for each entry in the array, as defined in 7.11.

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

6.3.6 VendorServerInfoType

This *ObjectType* defines a placeholder *Object* for vendor-specific information about the OPC UA *Server*. This *ObjectType* defines an empty *ObjectType* that has no components. It shall be subtyped by vendors to define their vendor-specific information. This *ObjectType* is formally defined in Table 14.

Table 14 – VendorServerInfoType Definition

Attribute	Value				
BrowseName	VendorServerInfoType				
IsAbstract	False				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the BaseObjectType defined in 6.2					

6.3.7 ServerRedundancyType

This *ObjectType* defines the redundancy capabilities supported by the OPC UA *Server*. It is formally defined in Table 15.

Table 15 – ServerRedundancyType Definition

Attribute	Value				
BrowseName	ServerRedundancyType				
IsAbstract	False				
References	**NodeClass**	**BrowseName**	**DataType**	**Type Definition**	**Modelling Rule**
Subtype of the BaseObjectType defined in 6.2					
HasProperty	Variable	RedundancySupport	RedundancySupport	PropertyType	Mandatory
HasSubtype	ObjectType	TransparentRedundancyType	Defined in 6.3.8		
HasSubtype	ObjectType	NonTransparentRedundancyType	Defined in 6.3.9		

RedundancySupport indicates what redundancy is supported by the *Server*. Its values are defined in 12.5. It shall be set to NONE_0 for all instances of the *ServerRedundancyType* using the *ObjectType* directly (no subtype).

6.3.8 TransparentRedundancyType

This *ObjectType* is a subtype of *ServerRedundancyType* and is used to identify the capabilities of the OPC UA *Server* for server-controlled redundancy with a transparent switchover for the client. It is formally defined in Table 16.

Table 16 – TransparentRedundancyType Definition

Attribute	Value				
BrowseName	TransparentRedundancyType				
IsAbstract	False				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Modelling Rule**
Subtype of the ServerRedundancyType defined in 6.3.7, i.e. inheriting the InstanceDeclarations of that Node.					
HasProperty	Variable	CurrentServerId	String	PropertyType	Mandatory
HasProperty	Variable	RedundantServerArray	RedundantServerDataType[]	PropertyType	Mandatory

RedundancySupport is inherited from the *ServerRedundancyType*. It shall be set to TRANSPARENT_4 for all instances of the *TransparentRedundancyType*.

Although, in a transparent switchover scenario, all redundant *Servers* serve under the same URI to the *Client*, it may be required to track the exact data source on the *Client*. Therefore, *CurrentServerId* contains an identifier of the currently-used *Server* in the *Redundant Set*. This *Server* is valid only inside a *Session*; if a *Client* opens several *Sessions*, different *Servers* of the redundant set of *Servers* may serve it in different *Sessions*. The value of the *CurrentServerId* may change due to *Failover* or load balancing, so a *Client* that needs to track its data source shall subscribe to this *Variable*.

AVA Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

As diagnostic information, the *RedundantServerArray* contains an array of available *Servers* in the *Redundant Set*; including their service levels (see 12.7). This array may change during a *Session*.

6.3.9 NonTransparentRedundancyType

This *ObjectType* is a subtype of *ServerRedundancyType* and is used to identify the capabilities of the OPC UA *Server* for non-transparent redundancy. It is formally defined in Table 17.

Table 17 – NonTransparentRedundancyType Definition

Attribute	Value				
BrowseName	NonTransparentRedundancyType				
IsAbstract	False				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**Modelling Rule**
Subtype of the ServerRedundancyType defined in 6.3.7, which means it inherits the InstanceDeclarations of that Node.					
HasProperty	Variable	ServerUriArray	String[]	PropertyType	Mandatory
HasSubtype	ObjectType	NonTransparentNetworkRedundancyType	Defined in 6.3.10		

ServerUriArray is an array with the URI of all redundant *Servers* of the OPC UA *Server*. See IEC 62541-4 for the definition of redundancy in this standard. In a non-transparent redundancy environment, the *Client* is responsible to subscribe to the redundant *Servers*. Therefore the *Client* might open a session to one or more redundant *Servers* of this array. The *ServerUriArray* shall contain the local *Server*.

RedundancySupport is inherited from the *ServerRedundancyType*. It shall be set to COLD_1, WARM_2, HOT_3 or HOT_AND_MIRRORED_5 for all instances of the *NonTransparentRedundancyType*. It defines the redundancy support provided by the *Server*. Its intended use is defined in IEC 62541-4.

6.3.10 NonTransparentNetworkRedundancyType

This *ObjectType* is a subtype of *NonTransparentRedundancyType* and is used to identify the capabilities of the OPC UA *Server* for non-transparent network redundancy. It is formally defined in Table 18.

Table 18 – NonTransparentNetworkRedundancyType Definition

Attribute	Value				
BrowseName	NonTransparentNetworkRedundancyType				
IsAbstract	False				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the NonTransparentRedundancyType defined in 6.3.9, which means it inherits the InstanceDeclarations of that Node.					
HasProperty	Variable	ServerNetworkGroups	NetworkGroupDataType[]	PropertyType	Mandatory

Clients switching between network paths to the same *Server* behave the same as HotAndMirrored redundancy. *Server* and network redundancy can be combined. In the combined approach it is important for the *Client* to know which ServerUris belong to the same *Server* representing different network paths and which ServerUris represent different Servers. Therefore, a *Server* implementing non-transparent network redundancy shall use the *NonTransparentNetworkRedundancyType* to identify its redundancy support.

RedundancySupport is inherited from the *ServerRedundancyType*. It shall be set to COLD_1, WARM_2, HOT_3 or HOT_AND_MIRRORED_5 for all instances of the *NonTransparentNetworkRedundancyType*. If no *Server* redundancy is supported (the *ServerUriArray* only contains one entry), the *RedundancySupport* shall be set to HOT_AND_MIRRORED_5.

The *ServerNetworkGroups* contains an array of *NetworkGroupDataType*. The URIs of the *Servers* in that array (in the *serverUri* of the structure) shall be exactly the same as the ones provided in the *ServerUriArray*. However, the order might be different. Thus the array represents a list of HotAndMirrored redundant *Servers*. If a *Server* only supports network

AVA Documents Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

redundancy, it has only one entry in the *ServerNetworkGroups*. The *networkPaths* in the structure represents the redundant network paths for each of the *Servers*. The *networkPaths* describes the different paths (one entry for each path) ordered by priority. Each network path contains an *endpointUrlList* having an array of Strings each containing a URL of an *Endpoint.* This allows using different protocol options for the same network path.

The *Endpoints* provided shall match with the *Endpoints* provided by the *GetEndpoints Service* of the corresponding *Server*.

6.3.11 OperationLimitsType

This *ObjectType* is a subtype of *FolderType* and is used to identify the operation limits of the OPC UA *Server*. It is formally defined in Table 19.

Table 19 – OperationLimitsType Definition

Attribute	Value				
BrowseName	OperationLimitsType				
IsAbstract	False				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the FolderType defined in 6.6, which means it inherits the InstanceDeclarations of that Node.					
HasProperty	Variable	MaxNodesPerRead	UInt32	PropertyType	Optional
HasProperty	Variable	MaxNodesPerHistoryReadData	UInt32	PropertyType	Optional
HasProperty	Variable	MaxNodesPerHistoryReadEvents	UInt32	PropertyType	Optional
HasProperty	Variable	MaxNodesPerWrite	UInt32	PropertyType	Optional
HasProperty	Variable	MaxNodesPerHistoryUpdateData	UInt32	PropertyType	Optional
HasProperty	Variable	MaxNodesPerHistoryUpdateEvents	UInt32	PropertyType	Optional
HasProperty	Variable	MaxNodesPerMethodCall	UInt32	PropertyType	Optional
HasProperty	Variable	MaxNodesPerBrowse	UInt32	PropertyType	Optional
HasProperty	Variable	MaxNodesPerRegisterNodes	UInt32	PropertyType	Optional
HasProperty	Variable	MaxNodesPerTranslateBrowsePaths ToNodeIds	UInt32	PropertyType	Optional
HasProperty	Variable	MaxNodesPerNodeManagement	UInt32	PropertyType	Optional
HasProperty	Variable	MaxMonitoredItemsPerCall	UInt32	PropertyType	Optional

Any operational limits *Property* that is provided shall have a non zero value.

The *MaxNodesPerRead Property* indicates the maximum size of the nodesToRead array when a *Client* calls the Read *Service*.

The *MaxNodesPerHistoryReadData Property* indicates the maximum size of the nodesToRead array when a *Client* calls the HistoryRead *Service* using the historyReadDetails RAW, PROCESSED, MODIFIED or ATTIME.

The *MaxNodesPerHistoryReadEvents Property* indicates the maximum size of the nodesToRead array when a *Client* calls the HistoryRead *Service* using the historyReadDetails EVENTS.

The *MaxNodesPerWrite Property* indicates the maximum size of the nodesToWrite array when a *Client* calls the Write *Service*.

The *MaxNodesPerHistoryUpdateData Property* indicates the maximum size of the historyUpdateDetails array supported by the *Server* when a *Client* calls the HistoryUpdate *Service*.

The *MaxNodesPerHistoryUpdateEvents Property* indicates the maximum size of the historyUpdateDetails array when a *Client* calls the HistoryUpdate *Service*.

The *MaxNodesPerMethodCall Property* indicates the maximum size of the methodsToCall array when a *Client* calls the Call *Service*.

The *MaxNodesPerBrowse Property* indicates the maximum size of the nodesToBrowse array when calling the Browse *Service* or the continuationPoints array when a *Client* calls the BrowseNext *Service*.

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

The *MaxNodesPerRegisterNodes Property* indicates the maximum size of the nodesToRegister array when a *Client* calls the RegisterNodes *Service* and the maximum size of the nodesToUnregister when calling the UnregisterNodes *Service*.

The *MaxNodesPerTranslateBrowsePathsToNodeIds Property* indicates the maximum size of the browsePaths array when a *Client* calls the TranslateBrowsePathsToNodeIds *Service*.

The *MaxNodesPerNodeManagement Property* indicates the maximum size of the nodesToAdd array when a *Client* calls the AddNodes *Service*, the maximum size of the referencesToAdd array when a *Client* calls the AddReferences *Service*, the maximum size of the nodesToDelete array when a *Client* calls the DeleteNodes *Service*, and the maximum size of the referencesToDelete array when a *Client* calls the DeleteReferences *Service*.

The *MaxMonitoredItemsPerCall Property* indicates

- the maximum size of the itemsToCreate array when a *Client* calls the CreateMonitoredItems *Service,*
- the maximum size of the itemsToModify array when a *Client* calls the ModifyMonitoredItems *Service*,
- the maximum size of the monitoredItemIds array when a *Client* calls the SetMonitoringMode *Service* or the DeleteMonitoredItems *Service*,
- the maximum size of the sum of the linksToAdd and linksToRemove arrays when a *Client* calls the SetTriggering *Service*.

6.3.12 AddressSpaceFileType

This *ObjectType* defines the file for a namespace provided by the OPC UA *Server*. It is formally defined in Table 20. It represents an XML address space file using the XML schema defined in IEC 62541-6.

Table 20 – AddressSpaceFileType Definition

Attribute	**Value**				
BrowseName	AddressSpaceFileType				
IsAbstract	False				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**Modelling Rule**
Subtype of the FileType defined in C.2					
HasComponent	Method	ExportNamespace	The method has no parameters.		Optional

The *ExportNamespace Method* provides a way to export the namespace from the *Server AddressSpace* to the XML file represented by the *AddressSpaceFileType*. *Value Attributes* are only exported if they represent static configuration information. The client is expected to call the *ExportNamespace Method* first to update the XML file and then access the file with the *Methods* defined in the *FileType*.

Servers might provide some vendor-specific mechanisms importing parts of an address space as subtype of this *ObjectType*, for example by defining appropriate *Methods*.

6.3.13 NamespaceMetadataType

This *ObjectType* defines the metadata for a namespace provided by the *Server*. It is formally defined in Table 21.

Instances of this *Object* allow *Servers* to provide more information like version information in addition to the namespace URI. Important information for aggregating *Servers* is provided by the *StaticNodeIdTypes, StaticNumericNodeIdRange and StaticStringNodeIdPattern Properties*.

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

534 **Table 21 – NamespaceMetadataType Definition**

Attribute	Value				
BrowseName	NamespaceMetadataType				
IsAbstract	False				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**Modelling Rule**
Subtype of the BaseObjectType defined in 6.2					
HasProperty	Variable	NamespaceUri	String	PropertyType	Mandatory
HasProperty	Variable	NamespaceVersion	String	PropertyType	Mandatory
HasProperty	Variable	NamespacePublicationDate	DateTime	PropertyType	Mandatory
HasProperty	Variable	IsNamespaceSubset	Boolean	PropertyType	Mandatory
HasProperty	Variable	StaticNodeIdTypes	IdType[]	PropertyType	Mandatory
HasProperty	Variable	StaticNumericNodeIdRange	NumericRange[]	PropertyType	Mandatory
HasProperty	Variable	StaticStringNodeIdPattern	String	PropertyType	Mandatory
HasComponent	Object	NamespaceFile	-	AddressSpaceFileType	Optional
HasProperty	Variable	DefaultRolePermissions	RolePermission Type[]	PropertyType	Optional
HasProperty	Variable	DefaultUserRolePermissions	RolePermission Type[]	PropretyType	Optional
HasProperty	Variable	DefaultAccessRestrictions	Uint16	PropertyType	Optional

535

536 The *BrowseName* of instances of this type shall be derived from the represented namespace.
537 This can, for example, be done by using the index of the namespace in the *NamespaceArray*
538 as *namespaceIndex* of the *QualifiedName* and the namespace URI as *name* of the
539 *QualifiedName*.

540 The *NamespaceUri Property* contains the namespace represented by an instance of the
541 MetaDataType.

542 The *NamespaceVersion Property* provides version information for the namespace. It is
543 intended for display purposes and shall not be used to programmatically identify the latest
544 version. If there is no formal version defined for the namespace this *Property* shall be set to a
545 null *String*.

546 The *NamespacePublicationDate* Property provides the publication date of the namespace
547 version. This *Property* value can be used by *Clients* to determine the latest version if different
548 versions are provided by different *Servers*. If there is no formal publication date defined for
549 the namespace this *Property* shall be set to a null *DateTime*.

550 The *IsNamespaceSubset Property* defines whether all *Nodes* of the namespace are
551 accessible in the *Server* or only a subset. It is set to FALSE if the full namespace is provided
552 and TRUE if not. If the completeness is unknown then this *Property* shall be set to TRUE.

553 Static *Nodes* are identical for all *Attributes* in all *Servers*, including the *Value Attribute*. For
554 *TypeDefinitionNodes*, also the *InstanceDeclarations* shall be identical. That means that for
555 static *Nodes* the semantic is always the same. Namespaces with static *Nodes* are for example
556 namespaces defined by standard bodies like the OPC Foundation. This is important
557 information for aggregating *Servers*. If the namespace is dynamic and used in several *Servers*
558 the aggregating *Server* needs to distinguish the namespace for each aggregated *Server*. The
559 static *Nodes* of a namespace only need to be handled once, even if they are used by several
560 aggregated *Servers*.

561 The *StaticNodeIdTypes Property* provides a list of *IdTypes* used for static *Nodes*. All *Nodes* in
562 the *AddressSpace* of the namespace using one of the *IdTypes* in the array shall be static
563 *Nodes*.

564 The *StaticNumericNodeIdRange Property* provides a list of *NumericRanges* used for numeric
565 *NodeIds* of static *Nodes*. If the *StaticNodeIdTypes Property* contains an entry for numeric
566 NodeIds then this Property is ignored.

567 The *StaticStringNodeIdPattern Property* provides a regular expression as defined for the *Like*
568 *Operator* defined in IEC 62541-4 to filter for string *NodeIds* of static *Nodes*. If the
569 *StaticNodeIdTypes Property* contains an entry for string *NodeIds* then this *Property* is ignored.

AVA Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

The *Object NamespaceFile* contains all *Nodes* and *References* of the namespace in an XML file where the Information Model XML Schema is defined in IEC 62541-6. The XML file is provided through an *AddressSpaceFileType Object*.

The *DefaultRolePermissions Property* provides the default permissions if a *Server* supports *RolePermissions* for the *Namespace.* A *Node* in the *Namespace* overrides this default by adding a *RolePermissions Attribute* to the *Node*. If a *Server* implements a vendor-specific *RolePermissions* model for a *Namespace,* it does not add the *DefaultRolePermissions Property* to the *NamespaceMetadata Object.*

The *DefaultUserRolePermissions Property* provides the default user permissions if a *Server* supports *UserRolePermissions* for the *Namespace.* A *Node* in the *Namespace* overrides this default by adding a *UserRolePermissions Attribute* to the *Node*. If a *Server* implements a vendor-specific *UserRolePermissions* model for a *Namespace,* it does not add the *DefaultUserRolePermissions Property* to the *NamespaceMetadata Object*.

The *DefaultAccessRestrictions Property* is present if a *Server* supports *AccessRestrictions* for the *Namespace* and provides the defaults. *A Node* in the *Namespace* overrides this default by adding a *AccessRestrictions Attribute* to the *Node.* If a *Server* implements a vendor-specific *AccessRestriction* model for a *Namespace,* it does not add the *DefaultAccessRestrictions Property* to the *NamespaceMetadata Object.*

6.3.14 NamespacesType

This *ObjectType* defines a list of *NamespaceMetadataType Objects* provided by the *Server*. It is formally defined in Table 22.

Table 22 – NamespacesType Definition

Attribute	**Value**				
BrowseName	NamespacesType				
IsAbstract	False				
References	**NodeClass**	**BrowseName**	**Data Type**	**TypeDefinition**	**Modelling Rule**
Subtype of the BaseObjectType defined in 6.2					
HasComponent	Object	<NamespaceIdentifier>	-	NamespaceMetadataType	OptionalPlaceholder

The *ObjectType* contains a list of *NamespaceMetadataType Objects* representing the namespaces in the *Server*. The *BrowseName* of an *Object* shall be derived from the namespace represented by the *Object*. This can, for example, be done by using the index of the namespace in the *NamespaceArray* as *namespaceIndex* of the *QualifiedName* and the namespace URI as *name* of the *QualifiedName*. *Clients* should not assume that all namespaces provided by a *Server* are present in this list as a namespace may not provide the information necessary to fill all mandatory *Properties* of the *NamespaceMetadataType*.

6.4 ObjectTypes used as EventTypes

6.4.1 General

This International Standard defines standard *EventTypes*. They are represented in the *AddressSpace* as *ObjectTypes*. The *EventTypes* are already defined in IEC 62541-3. The following subclauses specify their representation in the *AddressSpace*.

6.4.2 BaseEventType

This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally defined in Table 23.

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

608 **Table 23 – BaseEventType Definition**

Attribute	Value				
BrowseName	BaseEventType				
IsAbstract	True				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**Modelling Rule**
Subtype of the *BaseObjectType* defined in 6.2					
HasSubtype	ObjectType	AuditEventType	Defined in 6.4.3		
HasSubtype	ObjectType	SystemEventType	Defined in 6.4.28		
HasSubtype	ObjectType	BaseModelChangeEventType	Defined in 6.4.31		
HasSubtype	ObjectType	SemanticChangeEventType	Defined in 6.4.33		
HasSubtype	ObjectType	EventQueueOverflowEventType	Defined in 6.4.34		
HasSubtype	ObjectType	ProgressEventType	Defined in 6.4.35		
HasProperty	Variable	EventId	ByteString	PropertyType	Mandatory
HasProperty	Variable	EventType	NodeId	PropertyType	Mandatory
HasProperty	Variable	SourceNode	NodeId	PropertyType	Mandatory
HasProperty	Variable	SourceName	String	PropertyType	Mandatory
HasProperty	Variable	Time	UtcTime	PropertyType	Mandatory
HasProperty	Variable	ReceiveTime	UtcTime	PropertyType	Mandatory
HasProperty	Variable	LocalTime	TimeZoneDataType	PropertyType	Optional
HasProperty	Variable	Message	LocalizedText	PropertyType	Mandatory
HasProperty	Variable	Severity	UInt16	PropertyType	Mandatory

609

610 *EventId* is generated by the *Server* to uniquely identify a particular *Event Notification*. The
611 *Server* is responsible to ensure that each *Event* has its unique *EventId*. It may do this, for
612 example, by putting GUIDs into the ByteString. Clients can use the *EventId* to assist in
613 minimizing or eliminating gaps and overlaps that may occur during a redundancy failover. The
614 *EventId* shall always be returned as value and the *Server* is not allowed to return a
615 *StatusCode* for the *EventId* indicating an error.

616 *EventType* describes the specific type of *Event*. The *EventType* shall always be returned as
617 value and the *Server* is not allowed to return a *StatusCode* for the *EventType* indicating an
618 error.

619 The *SourceNode Property* identifies the *Node* that the *Event* originated from. If the *Event* is
620 not specific to a *Node* the *NodeId* is set to null. Some subtypes of this *BaseEventType* may
621 define additional rules for the *SourceNode Property*.

622 *SourceName* provides a description of the source of the *Event*. This could be the string-part of
623 the *DisplayName* of the *Event* source using the default locale of the server, if the *Event* is
624 specific to a *Node,* or some server-specific notation.

625 *Time* provides the time the *Event* occurred. This value is set as close to the event generator
626 as possible. It often comes from the underlying system or device. Once set, intermediate OPC
627 UA *Servers* shall not alter the value.

628 *ReceiveTime* provides the time the OPC UA *Server* received the *Event* from the underlying
629 device of another *Server*. *ReceiveTime* is analogous to *ServerTimestamp* defined in
630 IEC 62541-4, i.e. in the case where the OPC UA *Server* gets an *Event* from another OPC UA
631 *Server*, each *Server* applies its own *ReceiveTime*. That implies that a *Client* may get the same
632 *Event,* having the same *EventId*, from different *Servers* having different values of the
633 *ReceiveTime*. The *ReceiveTime* shall always be returned as value and the *Server* is not
634 allowed to return a *StatusCode* for the *ReceiveTime* indicating an error.

635 *LocalTime* is a structure containing the Offset and the DaylightSavingInOffset flag. The Offset
636 specifies the time difference (in minutes) between the *Time Property* and the time at the
637 location in which the event was issued. If DaylightSavingInOffset is TRUE, then
638 Standard/Daylight savings time (DST) at the originating location is in effect and Offset
639 includes the DST correction. If FALSE then the Offset does not include DST correction and
640 DST may or may not have been in effect.

641 *Message* provides a human-readable and localizable text description of the *Event*. The *Server*
642 may return any appropriate text to describe the *Event*. A null string is not a valid value, if the

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

Server does not have a description, it shall return the string part of the *BrowseName* of the *Node* associated with the *Event*.

Severity is an indication of the urgency of the *Event*. This is also commonly called “priority”. Values will range from 1 to 1 000, with 1 being the lowest severity and 1 000 being the highest. Typically, a severity of 1 would indicate an *Event* which is informational in nature, while a value of 1 000 would indicate an *Event* of catastrophic nature, which could potentially result in severe financial loss or loss of life.

It is expected that very few *Server* implementations will support 1 000 distinct severity levels. Therefore, *Server* developers are responsible for distributing their severity levels across the 1 to 1 000 range in such a manner that clients can assume a linear distribution. For example, a client wishing to present five severity levels to a user should be able to do the following mapping:

Client Severity	OPC Severity
HIGH	801 – 1 000
MEDIUM HIGH	601 – 800
MEDIUM	401 – 600
MEDIUM LOW	201 – 400
LOW	1 – 200

In many cases a strict linear mapping of underlying source severities to the OPC Severity range is not appropriate. The *Server* developer will instead intelligently map the underlying source severities to the 1 to 1 000 OPC Severity range in some other fashion. In particular, it is recommended that *Server* developers map *Events* of high urgency into the OPC severity range of 667 to 1 000, *Events* of medium urgency into the OPC severity range of 334 to 666 and *Events* of low urgency into OPC severities of 1 to 333.

For example, if a source supports 16 severity levels that are clustered such that severities 0 to 2 are considered to be LOW, 3 to 7 are MEDIUM and 8 to 15 are HIGH, then an appropriate mapping might be as follows:

OPC Range	Source Severity	OPC Severity
HIGH (667 – 1 000)	15	1 000
	14	955
	13	910
	12	865
	11	820
	10	775
	9	730
	8	685
MEDIUM (334 – 666)	7	650
	6	575
	5	500
	4	425
	3	350
LOW (1 – 333)	2	300
	1	150
	0	1

Some *Servers* might not support any *Events* which are catastrophic in nature, so they may choose to map all of their severities into a subset of the 1 to 1 000 range (for example, 1 to 666). Other *Servers* might not support any *Events* which are merely informational, so they may choose to map all of their severities into a different subset of the 1 to 1 000 range (for example, 334 to 1 000).

The purpose of this approach is to allow clients to use severity values from multiple *Servers* from different vendors in a consistent manner. Additional discussions of severity can be found in IEC 62541-9.

AVA Documents Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

6.4.3 AuditEventType

This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally defined in Table 24.

Table 24 – AuditEventType Definition

Attribute		Value			
BrowseName		AuditEventType			
IsAbstract		True			
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**Modelling Rule**
Subtype of the *BaseEventType* defined in 6.4.2, which means it inherits the InstanceDeclarations of that Node.					
HasSubtype	ObjectType	AuditSecurityEventType	Defined in 6.4.4		
HasSubtype	ObjectType	AuditNodeManagementEventType	Defined in 6.4.19		
HasSubtype	ObjectType	AuditUpdateEventType	Defined in 6.4.24		
HasSubtype	ObjectType	AuditUpdateMethodEventType	Defined in 6.4.27		
HasProperty	Variable	ActionTimeStamp	UtcTime	PropertyType	Mandatory
HasProperty	Variable	Status	Boolean	PropertyType	Mandatory
HasProperty	Variable	ServerId	String	PropertyType	Mandatory
HasProperty	Variable	ClientAuditEntryId	String	PropertyType	Mandatory
HasProperty	Variable	ClientUserId	String	PropertyType	Mandatory

This *EventType* inherits all *Properties* of the *BaseEventType*. Their semantic is defined in 6.4.2.

ActionTimeStamp identifies the time the user initiated the action that resulted in the *AuditEvent* being generated. It differs from the *Time Property* because this is the time the server generated the *AuditEvent* documenting the action.

Status identifies whether the requested action could be performed (set *Status* to TRUE) or not (set *Status* to FALSE).

ServerId uniquely identifies the *Server* generating the *Event*. It identifies the *Server* uniquely even in a server-controlled transparent redundancy scenario where several *Servers* may use the same URI.

ClientAuditEntryId contains the human-readable *AuditEntryId* defined in IEC 62541-3.

The *ClientUserId* identifies the user of the client requesting an action. The *ClientUserId* can be obtained from the *UserIdentityToken* passed in the *ActivateSession* call. If the *UserIdentityToken* is a *UserNameIdentityToken* then the *ClientUserId* is the *UserName.* If the *UserIdentityToken* is an *X509IdentityToken* then the *ClientUserId* is the X509 Subject Name of the *Certificate*. If the *UserIdentityToken* is an *IssuedIdentityToken* then the *ClientUserId* shall be a string that represents the owner of the token. The best choice for the string depends on the type of *IssuedIdentityToken.* If an *AnonymousIdentityToken* was used, the value is null.

6.4.4 AuditSecurityEventType

This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally defined in Table 25.

Table 25 – AuditSecurityEventType Definition

Attribute		Value			
BrowseName		AuditSecurityEventType			
IsAbstract		True			
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the AuditEventType defined in 6.4.3, which means it inherits the InstanceDeclarations of that Node.					
HasSubtype	ObjectType	AuditChannelEventType	Defined in 6.4.5		
HasSubtype	ObjectType	AuditSessionEventType	Defined in 6.4.7		
HasSubtype	ObjectType	AuditCertificateEventType	Defined in 6.4.12		
HasProperty	Variable	StatusCodeId	StatusCode	PropertyType	Optional

AVA Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

This *EventType* inherits all *Properties* of the *AuditEventType*. Their semantic is defined in 6.4.3. There are no additional *Properties* defined for this *EventType*.

The optional StatusCodeId *Property* provides the exact security error responsible for producing the *Event.*

6.4.5 AuditChannelEventType

This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally defined in Table 26.

Table 26 – AuditChannelEventType Definition

Attribute	Value				
BrowseName	AuditChannelEventType				
IsAbstract	True				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**Modelling Rule**
Subtype of the AuditSecurityEventType defined in 6.4.4, which means it inherits the InstanceDeclarations of that Node.					
HasSubtype	ObjectType	AuditOpenSecureChannelEventType	Defined in 6.4.6		
HasProperty	Variable	SecureChannelId	String	PropertyType	Mandatory

This *EventType* inherits all *Properties* of the *AuditSecurityEventType*. Their semantic is defined in 6.4.4. The *SourceNode Property* for *Events* of this type shall be assigned to the *Server Object*. The *SourceName* for *Events* of this type shall be “SecureChannel/” and the *Service* that generates the *Event* (e.g. SecureChannel/*OpenSecureChannel* or SecureChannel/*CloseSecureChannel*). If the *ClientUserId* is not available for a *CloseSecureChannel* call, then this parameter shall be set to ”System/CloseSecureChannel”.

The *SecureChannelId* shall uniquely identify the SecureChannel. The application shall use the same identifier in all *AuditEvents* related to the Session Service Set (AuditCreateSessionEventType, AuditActivateSessionEventType and their subtypes) and the SecureChannel Service Set (AuditChannelEventType and its subtypes).

6.4.6 AuditOpenSecureChannelEventType

This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally defined in Table 27.

Table 27 – AuditOpenSecureChannelEventType Definition

Attribute	Value				
BrowseName	AuditOpenSecureChannelEventType				
IsAbstract	True				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Modelling Rule**
Subtype of the *AuditChannelEventType* defined in 6.4.5, which means it inherits the InstanceDeclarations of that Node.					
HasProperty	Variable	ClientCertificate	ByteString	PropertyType	Mandatory
HasProperty	Variable	ClientCertificateThumbprint	String	PropertyType	Mandatory
HasProperty	Variable	RequestType	SecurityTokenRequestType	PropertyType	Mandatory
HasProperty	Variable	SecurityPolicyUri	String	PropertyType	Mandatory
HasProperty	Variable	SecurityMode	MessageSecurityMode	PropertyType	Mandatory
HasProperty	Variable	RequestedLifetime	Duration	PropertyType	Mandatory

This *EventType* inherits all *Properties* of the *AuditChannelEventType*. Their semantic is defined in 6.4.5. The *SourceName* for *Events* of this type shall be “SecureChannel/OpenSecureChannel”. The *ClientUserId* is not available for this call, thus this parameter shall be set to ”System/OpenSecureChannel”.

The additional *Properties* defined for this *EventType* reflect parameters of the *Service* call that triggers the *Event*.

ClientCertificate is the clientCertificate parameter of the OpenSecureChannel *Service* call.

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

734 *ClientCertificateThumbprint* is a thumbprint of the *ClientCertificate*. See IEC 62541-6 for
735 details on thumbprints.

736 *RequestType* is the requestType parameter of the OpenSecureChannel *Service* call.

737 *SecurityPolicyUri* is the securityPolicyUri parameter of the OpenSecureChannel *Service* call.

738 *SecurityMode* is the securityMode parameter of the OpenSecureChannel *Service* call.

739 *RequestedLifetime* is the requestedLifetime parameter of the OpenSecureChannel *Service*
740 call.

741 ### 6.4.7 AuditSessionEventType

742 This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally
743 defined in Table 28.

744 **Table 28 – AuditSessionEventType Definition**

Attribute		Value			
BrowseName		AuditSessionEventType			
IsAbstract		True			
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the *AuditSecurityEventType* defined in 6.4.4, which means it inherits the InstanceDeclarations of that Node.					
HasSubtype	ObjectType	AuditCreateSessionEventType	Defined in 6.4.8		
HasSubtype	ObjectType	AuditActivateSessionEventType	Defined in 6.4.10		
HasSubtype	ObjectType	AuditCancelEventType	Defined in 6.4.11		
HasProperty	Variable	SessionId	NodeId	PropertyType	Mandatory

745

746 This *EventType* inherits all *Properties* of the *AuditSecurityEventType*. Their semantic is
747 defined in 6.4.4.

748 If the *Event* is generated by a *TransferSubscriptions Service* call, the *SourceNode Property*
749 shall be assigned to the *SessionDiagnostics Object* that represents the session. The
750 *SourceName* for *Events* of this type shall be “Session/TransferSubscriptions”.

751 Otherwise, the *SourceNode Property* for *Events* of this type shall be assigned to the *Server*
752 *Object*. The *SourceName* for *Events* of this type shall be “Session/” and the *Service* or cause
753 that generates the *Event* (e.g. *CreateSession*, *ActivateSession* or *CloseSession*).

754 The *SessionId* shall contain the *SessionId* of the session that the *Service* call was issued on
755 In the *CreateSession Service* this shall be set to the newly created *SessionId*. If no session
756 context exists (e.g. for a failed *CreateSession Service* call) the *SessionId* shall be null.

757 ### 6.4.8 AuditCreateSessionEventType

758 This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally
759 defined in Table 29.

760 **Table 29 – AuditCreateSessionEventType Definition**

Attribute		Value			
BrowseName		AuditCreateSessionEventType			
IsAbstract		True			
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the *AuditSessionEventType* defined in 6.4.7, which means it inherits the InstanceDeclarations of that Node.					
HasSubtype	ObjectType	AuditUrlMismatchEventType	Defined in 6.4.9		
HasProperty	Variable	SecureChannelId	String	PropertyType	Mandatory
HasProperty	Variable	ClientCertificate	ByteString	PropertyType	Mandatory
HasProperty	Variable	ClientCertificateThumbprint	String	PropertyType	Mandatory
HasProperty	Variable	RevisedSessionTimeout	Duration	PropertyType	Mandatory

761

762 This *EventType* inherits all *Properties* of the *AuditSessionEventType*. Their semantic is
763 defined in 6.4.7. The *SourceName* for *Events* of this type shall be “Session/CreateSession”.

AVA Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

The *ClientUserId* is not available for this call thus this parameter shall be set to the ”System/CreateSession”.

The additional *Properties* defined for this *EventType* reflect parameters of the *Service* call that triggers the *Event*.

SecureChannelId shall uniquely identify the SecureChannel. The application shall use the same identifier in all *AuditEvents* related to the Session Service Set (AuditCreateSessionEventType, AuditActivateSessionEventType and their subtypes) and the SecureChannel Service Set (AuditChannelEventType and its subtypes).

ClientCertificate is the clientCertificate parameter of the CreateSession *Service* call.

ClientCertificateThumbprint is a thumbprint of the *ClientCertificate.* See IEC 62541-6 for details on thumbprints.

RevisedSessionTimeout is the returned revisedSessionTimeout parameter of the CreateSession *Service* call.

6.4.9 AuditUrlMismatchEventType

This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally defined in Table 30.

Table 30 – AuditUrlMismatchEventType Definition

Attribute	Value				
BrowseName	AuditUrlMismatchEventType				
IsAbstract	True				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the AuditCreateSessionEventType defined in 6.4.8 which means it inherits the InstanceDeclarations of that Node.					
HasProperty	Variable	EndpointUrl	String	PropertyType	Mandatory

This *EventType* inherits all *Properties* of the *AuditSessionEventType*. Their semantic is defined in 6.4.8.

The additional *Properties* defined for this *EventType* reflect parameters of the *Service* call that triggers the *Event*.

EndpointUrl is the endpointUrl parameter of the CreateSession *Service* call.

6.4.10 AuditActivateSessionEventType

This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally defined in Table 31.

Table 31 – AuditActivateSessionEventType Definition

Attribute	Value				
BrowseName	AuditActivateSessionEventType				
IsAbstract	True				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**Modelling Rule**
Subtype of the *AuditSessionEventType* defined in 6.4.7, which means it inherits the InstanceDeclarations of that Node.					
HasProperty	Variable	ClientSoftwareCertificates	SignedSoftwareCertificate[]	PropertyType	Mandatory
HasProperty	Variable	UserIdentityToken	UserIdentityToken	PropertyType	Mandatory
HasProperty	Variable	SecureChannelId	String	PropertyType	Mandatory

This *EventType* inherits all *Properties* of the *AuditSessionEventType*. Their semantic is defined in 6.4.7. The *SourceName* for *Events* of this type shall be “Session/ActivateSession”.

The additional *Properties* defined for this *EventType* reflect parameters of the *Service* call that triggers the *Event*.

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

ClientSoftwareCertificates is the clientSoftwareCertificates parameter of the ActivateSession *Service* call.

UserIdentityToken reflects the userIdentityToken parameter of the ActivateSession *Service* call. For Username/Password tokens the password shall not be included.

SecureChannelId shall uniquely identify the SecureChannel. The application shall use the same identifier in all *AuditEvents* related to the Session Service Set (AuditCreateSessionEventType, AuditActivateSessionEventType and their subtypes) and the SecureChannel Service Set (AuditChannelEventType and its subtypes).

6.4.11 AuditCancelEventType

This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally defined in Table 32.

Table 32 – AuditCancelEventType Definition

Attribute		Value			
BrowseName		AuditCancelEventType			
IsAbstract		True			
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the *AuditSessionEventType* defined in 6.4.7, i.e. inheriting the InstanceDeclarations of that Node.					
HasProperty	Variable	RequestHandle	UInt32	PropertyType	Mandatory

This *EventType* inherits all *Properties* of the *AuditSessionEventType*. Their semantic is defined in 6.4.7. The *SourceName* for *Events* of this type shall be “Session/Cancel”.

The additional *Properties* defined for this *EventType* reflect parameters of the *Service* call that triggers the *Event*.

RequestHandle is the requestHandle parameter of the Cancel *Service* call.

6.4.12 AuditCertificateEventType

This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally defined in Table 33.

Table 33 – AuditCertificateEventType Definition

Attribute		Value			
BrowseName		AuditCertificateEventType			
IsAbstract		True			
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**Modelling Rule**
Subtype of the *AuditSecurityEventType* defined in 6.4.7, which means it inherits the InstanceDeclarations of that Node.					
HasSubtype	ObjectType	AuditCertificateDataMismatchEventType	Defined in 6.4.13		
HasSubtype	ObjectType	AuditCertificateExpiredEventType	Defined in 6.4.14		
HasSubtype	ObjectType	AuditCertificateInvalidEventType	Defined in 6.4.15		
HasSubtype	ObjectType	AuditCertificateUntrustedEventType	Defined in 6.4.16		
HasSubtype	ObjectType	AuditCertificateRevokedEventType	Defined in 6.4.17		
HasSubtype	ObjectType	AuditCertificateMismatchEventType	Defined in 6.4.18		
HasProperty	Variable	Certificate	ByteString	PropertyType	Mandatory

This *EventType* inherits all *Properties* of the *AuditSecurityEventType*. Their semantic is defined in 6.4.4. The *SourceName* for *Events* of this type shall be “Security/Certificate”.

Certificate is the certificate that encountered a validation issue. Additional subtypes of this *EventType* will be defined representing the individual validation errors. This certificate can be matched to the *Service* that passed it (Session or SecureChannel Service Set) since the *AuditEvents* for these *Services* also included the Certificate.

6.4.13 AuditCertificateDataMismatchEventType

This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally defined in Table 34.

AVA Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

828 **Table 34 – AuditCertificateDataMismatchEventType Definition**

Attribute		Value			
BrowseName		AuditCertificateDataMismatchEventType			
IsAbstract		True			
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the AuditCertificateEventType defined in 6.4.12, i.e. inheriting the InstanceDeclarations of that Node.					
HasProperty	Variable	InvalidHostname	String	PropertyType	Mandatory
HasProperty	Variable	InvalidUri	String	PropertyType	Mandatory

829

830 This *EventType* inherits all *Properties* of the *AuditCertificateEventType*. Their semantic is
831 defined in 6.4.12. The *SourceName* for *Events* of this type shall be “Security/Certificate”.

832 *InvalidHostname* is the string that represents the host name passed in as part of the URL that
833 is found to be invalid. If the host name was not invalid it can be null.

834 *InvalidUri* is the URI that was passed in and found to not match what is contained in the
835 certificate. If the URI was not invalid it can be null.

836 Either the *InvalidHostname* or *InvalidUri* shall be provided.

837 ### 6.4.14 AuditCertificateExpiredEventType

838 This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally
839 defined in Table 35.

840 **Table 35 – AuditCertificateExpiredEventType Definition**

Attribute		Value			
BrowseName		AuditCertificateExpiredEventType			
IsAbstract		True			
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the *AuditCertificateEventType* defined in 6.4.12, which means it inherits the InstanceDeclarations of that Node.					

841

842 This *EventType* inherits all *Properties* of the *AuditCertificateEventType*. Their semantic is
843 defined in 6.4.12. The *SourceName* for *Events* of this type shall be “Security/Certificate”. The
844 *Message Variable* shall include a description of why the certificate was expired (i.e. time
845 before start or time after end). There are no additional *Properties* defined for this *EventType*.

846 ### 6.4.15 AuditCertificateInvalidEventType

847 This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally
848 defined in Table 36.

849 **Table 36 – AuditCertificateInvalidEventType Definition**

Attribute		Value			
BrowseName		AuditCertificateInvalidEventType			
IsAbstract		True			
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the *AuditCertificateEventType* defined in 6.4.12, which means it inherits the InstanceDeclarations of that Node.					

850

851 This *EventType* inherits all *Properties* of the *AuditCertificateEventType*. Their semantic is
852 defined in 6.4.12. The *SourceName* for *Events* of this type shall be “Security/Certificate”. The
853 *Message* shall include a description of why the certificate is invalid. There are no additional
854 *Properties* defined for this *EventType*.

855 ### 6.4.16 AuditCertificateUntrustedEventType

856 This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally
857 defined in Table 37.

AVA Documents Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

858 **Table 37 – AuditCertificateUntrustedEventType Definition**

Attribute		Value			
BrowseName		AuditCertificateUntrustedEventType			
IsAbstract		True			
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the *AuditCertificateEventType* defined in 6.4.12, which means it inherits the InstanceDeclarations of that Node.					

859

860 This *EventType* inherits all *Properties* of the *AuditCertificateEventType*. Their semantic is
861 defined in 6.4.12. The *SourceName* for *Events* of this type shall be “Security/Certificate”. The
862 *Message Variable* shall include a description of why the certificate is not trusted. If a trust
863 chain is involved then the certificate that failed in the trust chain should be described. There
864 are no additional *Properties* defined for this *EventType*.

865 ### 6.4.17 AuditCertificateRevokedEventType

866 This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally
867 defined in Table 38.

868 **Table 38 – AuditCertificateRevokedEventType Definition**

Attribute		Value			
BrowseName		AuditCertificateRevokedEventType			
IsAbstract		True			
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the *AuditCertificateEventType* defined in 6.4.12, which means it inherits the InstanceDeclarations of that Node.					

869

870 This *EventType* inherits all *Properties* of the *AuditCertificateEventType*. Their semantic is
871 defined in 6.4.12. The *SourceName* for *Events* of this type shall be “Security/Certificate”. The
872 *Message Variable* shall include a description of why the certificate is revoked (was the
873 revocation list unavailable or was the certificate on the list). There are no additional
874 *Properties* defined for this *EventType*.

875 ### 6.4.18 AuditCertificateMismatchEventType

876 This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally
877 defined in Table 39.

878 **Table 39 – AuditCertificateMismatchEventType Definition**

Attribute		Value			
BrowseName		AuditCertificateMismatchEventType			
IsAbstract		True			
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the *AuditCertificateEventType* defined in 6.4.12, which means it inherits the InstanceDeclarations of that Node.					

879

880 This *EventType* inherits all *Properties* of the *AuditCertificateEventType*. Their semantic is
881 defined in 6.4.12. The *SourceName* for *Events* of this type shall be “Security/Certificate”. The
882 *Message Variable* shall include a description of misuse of the certificate. There are no
883 additional *Properties* defined for this *EventType*.

884 ### 6.4.19 AuditNodeManagementEventType

885 This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally
886 defined in Table 40.

AVA Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

887 **Table 40 – AuditNodeManagementEventType Definition**

Attribute	Value				
BrowseName	AuditNodeManagementEventType				
IsAbstract	True				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the *AuditEventType* defined in 6.4.3, which means it inherits the InstanceDeclarations of that Node.					
HasSubtype	ObjectType	AuditAddNodesEventType			
HasSubtype	ObjectType	AuditDeleteNodesEventType			
HasSubtype	ObjectType	AuditAddReferencesEventType			
HasSubtype	ObjectType	AuditDeleteReferencesEventType			

888

889 This *EventType* inherits all *Properties* of the *AuditEventType*. Their semantic is defined in
890 6.4.3. There are no additional *Properties* defined for this *EventType*. The *SourceNode*
891 *Property* for *Events* of this type shall be assigned to the *Server Object*. The *SourceName* for
892 *Events* of this type shall be "NodeManagement/" and the *Service* that generates the *Event*
893 (e.g. *AddNodes*, *AddReferences*, *DeleteNodes*, *DeleteReferences*).

894 ### 6.4.20 AuditAddNodesEventType

895 This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally
896 defined in Table 41.

897 **Table 41 – AuditAddNodesEventType Definition**

Attribute	Value				
BrowseName	AuditAddNodesEventType				
IsAbstract	True				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the *AuditNodeManagementEventType* defined in 6.4.19, which means it inherits the InstanceDeclarations of that Node.					
HasProperty	Variable	NodesToAdd	AddNodesItem[]	PropertyType	Mandatory

898

899 This *EventType* inherits all *Properties* of the *AuditNodeManagementEventType*. Their
900 semantic is defined in 6.4.19. The *SourceName* for *Events* of this type shall be
901 "NodeManagement/AddNodes".

902 The additional *Properties* defined for this *EventType* reflect parameters of the *Service* call that
903 triggers the *Event*.

904 *NodesToAdd* is the NodesToAdd parameter of the AddNodes *Service* call.

905 ### 6.4.21 AuditDeleteNodesEventType

906 This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally
907 defined in Table 42.

908 **Table 42 – AuditDeleteNodesEventType Definition**

Attribute	Value				
BrowseName	AuditDeleteNodesEventType				
IsAbstract	True				
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the *AuditNodeManagementEventType* defined in 6.4.19, i.e. inheriting the InstanceDeclarations of that Node.					
HasProperty	Variable	NodesToDelete	DeleteNodesItem[]	PropertyType	Mandatory

909

910 This *EventType* inherits all *Properties* of the *AuditNodeManagementEventType*. Their
911 semantic is defined in 6.4.19. The *SourceName* for *Events* of this type shall be
912 "NodeManagement/DeleteNodes".

913 The additional *Properties* defined for this *EventType* reflect parameters of the *Service* call that
914 triggers the *Event*.

915 *NodesToDelete* is the nodesToDelete parameter of the DeleteNodes *Service* call.

AVA Documents Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

6.4.22 AuditAddReferencesEventType

This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally defined in Table 43.

Table 43 – AuditAddReferencesEventType Definition

Attribute		Value			
BrowseName		AuditAddReferencesEventType			
IsAbstract		True			
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the *AuditNodeManagementEventType* defined in 6.4.19, which means it inherits the InstanceDeclarations of that Node.					
HasProperty	Variable	ReferencesToAdd	AddReferencesItem[]	PropertyType	Mandatory

This *EventType* inherits all *Properties* of the *AuditNodeManagementEventType*. Their semantic is defined in 6.4.19. The *SourceName* for *Events* of this type shall be "NodeManagement/AddReferences".

The additional *Properties* defined for this *EventType* reflect parameters of the *Service* call that triggers the *Event*.

ReferencesToAdd is the referencesToAdd parameter of the AddReferences *Service* call.

6.4.23 AuditDeleteReferencesEventType

This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally defined in Table 44.

Table 44 – AuditDeleteReferencesEventType Definition

Attribute		Value			
BrowseName		AuditDeleteReferencesEventType			
IsAbstract		True			
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the *AuditNodeManagementEventType* defined in 6.4.19, which means it inherits the InstanceDeclarations of that Node.					
HasProperty	Variable	ReferencesToDelete	DeleteReferencesItem[]	PropertyType	Mandatory

This *EventType* inherits all *Properties* of the *AuditNodeManagementEventType*. Their semantic is defined in 6.4.19. The *SourceName* for *Events* of this type shall be "NodeManagement/DeleteReferences".

The additional *Properties* defined for this *EventType* reflect parameters of the *Service* call that triggers the *Event*.

ReferencesToDelete is the referencesToDelete parameter of the DeleteReferences *Service* call.

6.4.24 AuditUpdateEventType

This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally defined in Table 45.

Table 45 – AuditUpdateEventType Definition

Attribute		Value			
BrowseName		AuditUpdateEventType			
IsAbstract		True			
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the *AuditEventType* defined in 6.4.3, which means it inherits the InstanceDeclarations of that Node.					
HasSubtype	ObjectType	AuditWriteUpdateEventType	Defined in 6.4.25		
HasSubtype	ObjectType	AuditHistoryUpdateEventType	Defined in 6.4.26		

This *EventType* inherits all *Properties* of the *AuditEventType*. Their semantic is defined in 6.4.3. The *SourceNode Property* for *Events* of this type shall be assigned to the *NodeId* that

AVA Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09

was changed. The *SourceName* for *Events* of this type shall be “Attribute/” and the *Service* that generated the event (e.g. *Write*, *HistoryUpdate*). Note that one *Service* call may generate several *Events* of this type, one per changed value.

6.4.25 AuditWriteUpdateEventType

This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally defined in Table 46.

Table 46 – AuditWriteUpdateEventType Definition

Attribute		Value			
BrowseName		AuditWriteUpdateEventType			
IsAbstract		True			
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the *AuditUpdateEventType* defined in 6.4.24, which means it inherits the InstanceDeclarations of that Node.					
HasProperty	Variable	AttributeId	UInt32	PropertyType	Mandatory
HasProperty	Variable	IndexRange	NumericRange	PropertyType	Mandatory
HasProperty	Variable	NewValue	BaseDataType	PropertyType	Mandatory
HasProperty	Variable	OldValue	BaseDataType	PropertyType	Mandatory

This *EventType* inherits all *Properties* of the *AuditUpdateEventType*. The *SourceName* for *Events* of this type shall be “Attribute/Write”. Their semantic is defined in 6.4.24.

AttributeId identifies the *Attribute* that was written. The *SourceNode Property* identifies the *Node* that was written.

IndexRange identifies the index range of the written *Attribute* if the *Attribute* is an array. If the *Attribute* is not an array or the whole array was written, the *IndexRange* is set to null.

NewValue identifies the value that was written. If the *IndexRange* is provided, only the values in the provided range are shown.

OldValue identifies the value that the *Attribute* contained before the write. If the *IndexRange* is provided, only the value of that range is shown. It is acceptable for a *Server* that does not have this information to report a null value.

Both the *NewValue* and the *OldValue* will contain a value in the *DataType* and encoding used for writing the value.

6.4.26 AuditHistoryUpdateEventType

This *EventType* is defined in IEC 62541-3. Its representation in the *AddressSpace* is formally defined in Table 47.

Table 47 – AuditHistoryUpdateEventType Definition

Attribute		Value			
BrowseName		AuditHistoryUpdateEventType			
IsAbstract		True			
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
Subtype of the *AuditUpdateEventType* defined in 6.4.24, which means it inherits the InstanceDeclarations of that Node.					
HasProperty	Variable	ParameterDataTypeId	NodeId	PropertyType	New

This *EventType* inherits all *Properties* of the *AuditUpdateEventType*. Their semantic is defined in 6.4.24.

The *ParameterDataTypeId* identifies the *DataTypeId* for the extensible parameter used by the HistoryUpdate. This parameter indicates the type of HistoryUpdate being performed.

Subtypes of this *EventType* are defined in IEC 62541-11 representing the different possibilities to manipulate historical data.

Documents
Document Sharing Hub
Full document details: E DIN EN IEC 62541-5:2019-09