|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | |  | |  |  | OPC UA |
|  | | |
| Title | | |

**CONTENTS**

[1 Scope i](#_Toc78197945)

[2 Normative references ii](#_Toc78197946)

[3 Terms, abbreviated terms and conventions iii](#_Toc78197947)

[3.1 Overview iii](#_Toc78197948)

[3.2 OPC UA for OPC UA 4 GMS terms iii](#_Toc78197949)

[3.3 Abbreviated terms iv](#_Toc78197950)

[3.4 Conventions used in this document iv](#_Toc78197951)

[3.4.1 Conventions for Node descriptions iv](#_Toc78197952)

[3.4.2 NodeIds and BrowseNames vii](#_Toc78197953)

[3.4.3 Common Attributes viii](#_Toc78197954)

[4 General information to GMS and OPC UA 10](#_Toc78197955)

[4.1 Introduction to GMS 10](#_Toc78197956)

[4.1.1 What is a GMS? 10](#_Toc78197957)

[4.1.2 Result Management in GMS 10](#_Toc78197958)

[4.2 Introduction to OPC Unified Architecture 11](#_Toc78197959)

[4.2.1 What is OPC UA? 11](#_Toc78197960)

[4.2.2 Basics of OPC UA 11](#_Toc78197961)

[4.2.3 Information modelling in OPC UA 12](#_Toc78197962)

[5 Use cases 16](#_Toc78197963)

[5.1 Retrieve dynamic machine status 16](#_Toc78197964)

[5.2 • Managing workpiece data 17](#_Toc78197965)

[5.3 • Retrieve measurement results 17](#_Toc78197966)

[5.4 • Retrieve static machine data 17](#_Toc78197967)

[5.5 • Job management 17](#_Toc78197968)

[6 GMS Information Model overview 17](#_Toc78197969)

[7 OPC UA ObjectTypes 18](#_Toc78197970)

[7.1 undefined 18](#_Toc78197971)

[8 OPC UA EventTypes 19](#_Toc78197972)

[8.1 <some>EventType 19](#_Toc78197973)

[9 OPC UA VariableTypes 19](#_Toc78197974)

[9.1 <some>VariableType 19](#_Toc78197975)

[10 OPC UA DataTypes 20](#_Toc78197976)

[10.1 <someStructure> 20](#_Toc78197977)

[10.2 <someUnion> 20](#_Toc78197978)

[10.3 <someEnumeration> 21](#_Toc78197979)

[10.4 <someOptionSet> 21](#_Toc78197980)

[11 OPC UA ReferenceTypes 22](#_Toc78197981)

[11.1 <someReferenceType> 22](#_Toc78197982)

[12 Instances 22](#_Toc78197983)

[12.1 <someInstance> 22](#_Toc78197984)

[13 Profiles and Conformance Units 23](#_Toc78197985)

[13.1 Conformance Units 23](#_Toc78197986)

[13.2 Profiles 23](#_Toc78197987)

[13.2.1 Profile list 23](#_Toc78197988)

[13.2.2 Server Facets 24](#_Toc78197989)

[13.2.3 Client Facets 25](#_Toc78197990)

[14 Namespaces 25](#_Toc78197991)

[14.1 Namespace Metadata 25](#_Toc78197992)

[14.2 Handling of OPC UA Namespaces 26](#_Toc78197993)

[Annex A (normative) <Title> Namespace and mappings 28](#_Toc78197994)

[A.1 Namespace and identifiers for <Title> Information Model 28](#_Toc78197995)

**Figures**

[Figure 1 – The Scope of OPC UA within an Enterprise 13](#_Toc73441124)

[Figure 2 – A Basic Object in an OPC UA Address Space 14](#_Toc73441125)

[Figure 3 – The Relationship between Type Definitions and Instances 15](#_Toc73441126)

[Figure 4 – Examples of References between Objects 16](#_Toc73441127)

[Figure 5 – The OPC UA Information Model Notation 16](#_Toc73441128)

**Tables**

[Table 1 – Examples of DataTypes v](#_Toc73441129)

[Table 2 – Type Definition Table v](#_Toc73441130)

[Table 3 – Examples of Other Characteristics vi](#_Toc73441131)

[Table 4 – <some>Type Additional References vi](#_Toc73441132)

[Table 5 – <some>Type Additional Subcomponents vi](#_Toc73441133)

[Table 6 – <some>Type Attribute values for child Nodes vii](#_Toc73441134)

[Table 7 – Common Node Attributes viii](#_Toc73441135)

[Table 8 – Common Object Attributes viii](#_Toc73441136)

[Table 9 – Common Variable Attributes ix](#_Toc73441137)

[Table 10 – Common VariableType Attributes ix](#_Toc73441138)

[Table 11 – Common Method Attributes ix](#_Toc73441139)

[Table 12 – <Table With Embedded Graphics> x](#_Toc73441140)

[Table 13 – <some>Type Definition 20](#_Toc73441141)

[Table 14 – <some>Type Additional References 20](#_Toc73441142)

[Table 15 – <some>Type Additional Subcomponents 21](#_Toc73441143)

[Table 16 – <some>Type Attribute values for child Nodes 23](#_Toc73441144)

[Table 17 – <some>Method Method Arguments 23](#_Toc73441145)

[Table 18 – <some>Method Method AddressSpace definition 24](#_Toc73441146)

[Table 19 – ExampleStateMachineType Definition 24](#_Toc73441147)

[Table 20 – ExampleStateMachineType Additional References 25](#_Toc73441148)

[Table 21 – ExampleStateMachineType Attribute values for child Nodes 25](#_Toc73441149)

[Table 22 – <some>EventType Definition 25](#_Toc73441150)

[Table 23 – <some>Type Definition 26](#_Toc73441151)

[Table 24 – <someStructure> Structure 26](#_Toc73441152)

[Table 25 – <someStructure> Definition 26](#_Toc73441153)

[Table 26 – <someUnion> Union 27](#_Toc73441154)

[Table 27 – <someUnion> Definition 27](#_Toc73441155)

[Table 28 – <someEnumeration> Items 27](#_Toc73441156)

[Table 29 – <someEnumeration> Definition 27](#_Toc73441157)

[Table 30 – <someOptionSet> Values 28](#_Toc73441158)

[Table 31 – <someOptionSet> Definition 28](#_Toc73441159)

[Table 32 – <someReferenceType> Definition 28](#_Toc73441160)

[Table 33 – <someInstance> Definition 29](#_Toc73441161)

[Table 34 – Conformance Units for <Title> 29](#_Toc73441162)

[Table 35 – Profile URIs for <Title> 30](#_Toc73441163)

[Table 36 - <short name> <Prf1name> Server Profile 30](#_Toc73441164)

[Table 37 - <short name> <Prf2name> Server Facet 31](#_Toc73441165)

[Table 38 - <short name> < Prf3name> Client Facet 31](#_Toc73441166)

[Table 39 – NamespaceMetadata Object for this Document 32](#_Toc73441167)

[Table 40 – Namespaces used in a <title> Server 33](#_Toc73441168)

[Table 41 – Namespaces used in this document 33](#_Toc73441169)

# OPC UA ObjectTypes

## ProbeType

The ProbeType provides information about …

The ProbeType is formally defined in …

Table 13 – ProbeType Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | ProbeType | | | | |
| IsAbstract | False | | | | |
| **References** | **Node Class** | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the ns=2;i=51 defined in undefined i.e. inheriting the InstanceDeclarations of that Node. | | | | | |

## ProbeTipType

The ProbeTipType provides information about …

The ProbeTipType is formally defined in …

Table 13 – ProbeTipType Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | ProbeTipType | | | | |
| IsAbstract | False | | | | |
| **References** | **Node Class** | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the ns=2;i=50 defined in undefined i.e. inheriting the InstanceDeclarations of that Node. | | | | | |
| HasProperty | UAVariable | ToolAlignement | ToolAlignmentState | PropertyType | Optional |
| HasComponent | UAVariable | ToolCapabilities | UInteger | MultiStateDiscreteType | Optional |
| HasComponent | UAVariable | ToolClass | UInteger | MultiStateDiscreteType | Mandatory |
| HasProperty | UAVariable | ToolIsQualifiedStatus | ToolIsQualifiedStatus | PropertyType | Optional |
| HasComponent | UAObject | 2:ToolLife |  | BaseObjectType | Optional |

ToolAlignement is defined as Describes if tool is fixed or flexible

ToolCapabilities is defined as List of tool capabilities (e.g. single point scanning, contact, …)

ToolClass is defined as Class of active tool

ToolIsQualifiedStatus is defined as Status of the tool qualification

2:ToolLife is defined as undefined

## LoadingMonitoringType

The LoadingMonitoringType provides information about …

The LoadingMonitoringType is formally defined in …

Table 13 – LoadingMonitoringType Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | LoadingMonitoringType | | | | |
| IsAbstract | False | | | | |
| **References** | **Node Class** | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the ns=2;i=23 defined in undefined i.e. inheriting the InstanceDeclarations of that Node. | | | | | |
| HasComponent | UAVariable | LoadingPosition | Boolean | BaseDataVariableType | Optional |
| HasComponent | UAVariable | LoadStatus | LoadStatusEnum | BaseDataVariableType | Mandatory |

LoadingPosition is defined as Machine is in a safe position for loading

LoadStatus is defined as Information if a part is loaded or not

## ToolMonitoringType

The ToolMonitoringType provides information about …

The ToolMonitoringType is formally defined in …

Table 13 – ToolMonitoringType Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | ToolMonitoringType | | | | |
| IsAbstract | False | | | | |
| **References** | **Node Class** | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the ns=2;i=41 defined in undefined i.e. inheriting the InstanceDeclarations of that Node. | | | | | |
| HasProperty | UAVariable | ActiveTool | NodeId | PropertyType | Optional |

ActiveTool is defined as undefined

## GMSEquipmentType

The GMSEquipmentType provides information about …

The GMSEquipmentType is formally defined in …

Table 13 – GMSEquipmentType Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | GMSEquipmentType | | | | |
| IsAbstract | False | | | | |
| **References** | **Node Class** | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the ns=2;i=12 defined in undefined i.e. inheriting the InstanceDeclarations of that Node. | | | | | |
| HasProperty | UAVariable | NodeVersion | String | PropertyType | Optional |
| HasComponent | UAObject | Sensors |  | 1:SensorListType | Optional |
| HasComponent | UAObject | 2:Tools |  | ToolListType | Optional |

NodeVersion is defined as undefined

Sensors is defined as List of all available sensors.
  
Important: the tool is in this case not a sensor!

2:Tools is defined as undefined

## GMSIdentificationType

The GMSIdentificationType provides information about …

The GMSIdentificationType is formally defined in …

Table 13 – GMSIdentificationType Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | GMSIdentificationType | | | | |
| IsAbstract | False | | | | |
| **References** | **Node Class** | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the ns=2;i=11 defined in undefined i.e. inheriting the InstanceDeclarations of that Node. | | | | | |
| HasProperty | UAVariable | SubDeviceClass | String | PropertyType | Optional |
| HasComponent | UAVariable | Workspace | 3DVector | BaseDataVariableType | Optional |

SubDeviceClass is defined as SubDeviceClass is a more different classification of the DeviceClass.

Workspace is defined as workspace of the axes / usually but not always the same as the Measuring Range

## GMSOperationMonitoringType

The GMSOperationMonitoringType provides information about …

The GMSOperationMonitoringType is formally defined in …

Table 13 – GMSOperationMonitoringType Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | GMSOperationMonitoringType | | | | |
| IsAbstract | False | | | | |
| **References** | **Node Class** | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the ns=2;i=26 defined in undefined i.e. inheriting the InstanceDeclarations of that Node. | | | | | |
| HasComponent | UAVariable | WorkingHoursSinceCalibration | UInt32 | AnalogUnitType | Mandatory |

WorkingHoursSinceCalibration is defined as Duration since last calibration

## GMSType

The GMSType provides information about …

The GMSType is formally defined in …

Table 13 – GMSType Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | GMSType | | | | |
| IsAbstract | False | | | | |
| **References** | **Node Class** | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the ns=2;i=13 defined in undefined i.e. inheriting the InstanceDeclarations of that Node. | | | | | |
| HasComponent | UAObject | 2:Equipment |  | 1:GMSEquipmentType | Mandatory |
| HasAddIn | UAObject | 3:Identification |  | 1:GMSIdentificationType | Mandatory |
| HasComponent | UAObject | 2:Notification |  | NotificationType | Mandatory |
| HasComponent | UAObject | 2:Production |  | 2:ProductionType | Mandatory |
| HasComponent | UAObject | Results |  | 1:ResultManagmentType | Mandatory |

2:Equipment is defined as undefined

3:Identification is defined as undefined

2:Notification is defined as undefined

2:Production is defined as undefined

Results is defined as undefined

## GMSMonitoringType

The GMSMonitoringType provides information about …

The GMSMonitoringType is formally defined in …

Table 13 – GMSMonitoringType Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | GMSMonitoringType | | | | |
| IsAbstract | False | | | | |
| **References** | **Node Class** | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the ns=2;i=14 defined in undefined i.e. inheriting the InstanceDeclarations of that Node. | | | | | |
| HasComponent | UAObject | LoadingMonitoring |  | 1:LoadingMonitoringType | Optional |
| HasComponent | UAObject | 2:MachineTool |  | 1:GMSOperationMonitoringType | Mandatory |
| HasComponent | UAObject | ToolMonitoring |  | 1:ToolMonitoringType | Optional |

LoadingMonitoring is defined as undefined

2:MachineTool is defined as undefined

ToolMonitoring is defined as undefined

## GMSJobType

The GMSJobType provides information about …

The GMSJobType is formally defined in …

Table 13 – GMSJobType Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | GMSJobType | | | | |
| IsAbstract | False | | | | |
| **References** | **Node Class** | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the ns=2;i=29 defined in undefined i.e. inheriting the InstanceDeclarations of that Node. | | | | | |
| HasProperty | UAVariable | BatchIdentifier | Duration | PropertyType | Optional |
| HasProperty | UAVariable | Duration | Duration | PropertyType | Optional |
| HasProperty | UAVariable | ReamingTime | Duration | PropertyType | Optional |

BatchIdentifier is defined as undefined

Duration is defined as Approximative time this job will take. This value can be estimated by the machine based on the last measurement runs or on the measuring equipment capability test.

ReamingTime is defined as undefined

## GMSPartType

The GMSPartType provides information about …

The GMSPartType is formally defined in …

Table 13 – GMSPartType Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | GMSPartType | | | | |
| IsAbstract | False | | | | |
| **References** | **Node Class** | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the ns=2;i=56 defined in undefined i.e. inheriting the InstanceDeclarations of that Node. | | | | | |
| HasComponent | UAVariable | NestIdentifier | Number | 1:CatalogueType | Optional |
| HasComponent | UAVariable | Operator | Number | 1:CatalogueType | Optional |
| HasProperty | UAVariable | PartAmendmentStatus | String | PropertyType | Optional |
| HasComponent | UAVariable | PartCarrierIdentifier | Number | 1:CatalogueType | Optional |
| HasProperty | UAVariable | PartDescription | String | PropertyType | Optional |
| HasComponent | UAVariable | ProccessParamter | Number | 1:CatalogueType | Optional |
| HasComponent | UAVariable | ProcessingMachineIdentifier | Number | 1:CatalogueType | Optional |
| HasProperty | UAVariable | ProductionNumber | String | PropertyType | Optional |

NestIdentifier is defined as Indicates the nesting or spindle used to create or process this part

Operator is defined as Specifies the operator who performs the measurement

PartAmendmentStatus is defined as The Identifier is used to distinguish and identify an version of the part Type in production in a GMS.

PartCarrierIdentifier is defined as Defines an unique ID for the part carrier

PartDescription is defined as undefined

ProccessParamter is defined as Additional parameter that descripes the process

ProcessingMachineIdentifier is defined as Defines an unique Identifier of the machine that has procced the part.

ProductionNumber is defined as Is a free identifier that idenfitier a subset of a batch.

## LMS\_ProductionStatisticsType

The LMS\_ProductionStatisticsType provides information about …

The LMS\_ProductionStatisticsType is formally defined in …

Table 13 – LMS\_ProductionStatisticsType Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | LMS\_ProductionStatisticsType | | | | |
| IsAbstract | False | | | | |
| **References** | **Node Class** | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the ns=2;i=20 defined in undefined i.e. inheriting the InstanceDeclarations of that Node. | | | | | |

## CalibrationPrognosisType

The CalibrationPrognosisType provides information about …

The CalibrationPrognosisType is formally defined in …

Table 13 – CalibrationPrognosisType Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | CalibrationPrognosisType | | | | |
| IsAbstract | False | | | | |
| **References** | **Node Class** | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the ns=2;i=3 defined in undefined i.e. inheriting the InstanceDeclarations of that Node. | | | | | |
| HasProperty | UAVariable | Calibrated | Boolean | PropertyType | Mandatory |
| HasProperty | UAVariable | CalibrationInterval | Duration | PropertyType | Mandatory |
| HasProperty | UAVariable | CalibrationPreptime | Duration | PropertyType | Mandatory |
| HasProperty | UAVariable | DateOfCalibration | UtcTime | PropertyType | Mandatory |

Calibrated is defined as Indicates whether the machine is calibrated

CalibrationInterval is defined as undefined

CalibrationPreptime is defined as The time between CalibrationWarningDateTime and the Time the machine is not calibrated.

DateOfCalibration is defined as Date and time when the last calibration was carried out.

## ResultManagmentType

The ResultManagmentType provides information about …

The ResultManagmentType is formally defined in …

Table 13 – ResultManagmentType Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | ResultManagmentType | | | | |
| IsAbstract | False | | | | |
| **References** | **Node Class** | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the i=58 defined in undefined i.e. inheriting the InstanceDeclarations of that Node. | | | | | |
| HasComponent | UAObject | <Results> |  | BaseObjectType | OptionalPlaceholder |

<Results> is defined as undefined

## ResultType

The ResultType provides information about …

The ResultType is formally defined in …

Table 13 – ResultType Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | ResultType | | | | |
| IsAbstract | False | | | | |
| **References** | **Node Class** | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the i=58 defined in undefined i.e. inheriting the InstanceDeclarations of that Node. | | | | | |
| HasProperty | UAVariable | EndTime | String | PropertyType | Mandatory |
| HasComponent | UAObject | File |  | FileType | Optional |
| HasProperty | UAVariable | Format | String | PropertyType | Mandatory |
| HasProperty | UAVariable | Identifier | String | PropertyType | Mandatory |
| HasProperty | UAVariable | JobIdentifier | String | PropertyType | Mandatory |
| HasProperty | UAVariable | PartId | String | PropertyType | Mandatory |
| HasProperty | UAVariable | Program | String | PropertyType | Mandatory |
| HasProperty | UAVariable | StartTime | String | PropertyType | Mandatory |
| HasProperty | UAVariable | Status | String | PropertyType | Mandatory |
| HasProperty | UAVariable | Uri | String | PropertyType | Mandatory |
| HasProperty | UAVariable | UsedTools | String | PropertyType | Mandatory |

EndTime is defined as Exact time when the MeasurementRoutine was ended

File is defined as Optional value for when the measuremet results are available as a file download over OPC UA

Format is defined as The format in which the measurement results are available (e.g. QDAS, CSV, …)

Identifier is defined as The Identifier Property provides an identifier to distinguish the software component.

JobIdentifier is defined as undefined

PartId is defined as undefined

Program is defined as undefined

StartTime is defined as Exact time when the MeasurementRoutine was started

Status is defined as NIO defined as: a certain feature is outside of the tolerance

Uri is defined as undefined

UsedTools is defined as An array of all used tools to create this result

## RotaryTableType

The RotaryTableType provides information about …

The RotaryTableType is formally defined in …

Table 13 – RotaryTableType Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | RotaryTableType | | | | |
| IsAbstract | False | | | | |
| **References** | **Node Class** | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the i=58 defined in undefined i.e. inheriting the InstanceDeclarations of that Node. | | | | | |
| HasComponent | UAObject | Identifcation |  | MachineryItemIdentificationType | Mandatory |
| HasProperty | UAVariable | IsIntegarted | String | PropertyType | Mandatory |
| HasProperty | UAVariable | NumberOfAxes | String | PropertyType | Mandatory |

Identifcation is defined as undefined

IsIntegarted is defined as undefined

NumberOfAxes is defined as The number of DegreeOfFreedom the RotaryTable can be changes.

## SensorListType

The SensorListType provides information about …

The SensorListType is formally defined in …

Table 13 – SensorListType Definition

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | | |
| BrowseName | SensorListType | | | | |
| IsAbstract | False | | | | |
| **References** | **Node Class** | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the i=58 defined in undefined i.e. inheriting the InstanceDeclarations of that Node. | | | | | |
| HasComponent | UAVariable | <Sensor> | Number | 1:BaseSensorType | OptionalPlaceholder |

<Sensor> is defined as undefined

# OPC UA EventTypes

## <some>EventType

This *EventType* is ….. Its representation in the *AddressSpace* is formally defined in Table 15.

Table 15 – <some>EventType Definition

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Attribute** | | **Value** | | | | |
| BrowseName | | <some>EventType | | | | |
| IsAbstract | | True | | | | |
| **References** | **NodeClass** | | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the *BaseEventType* defined in …, which means it inherits the InstanceDeclarations of that Node. | | | | | | |
| 0:HasSubtype | ObjectType | | <someother>EventType | Defined in | | |
| 0:HasProperty | Variable | | <some>Eventfield | 0:String | 0:PropertyType | 0:Mandatory |

This *EventType* inherits all *Properties* of the *BaseEventType*. ….

# OPC UA VariableTypes

## <some>VariableType

The <some>*VariableType* is a subtype of the *BaseVariableType*. It is used ….

It is formally defined in Table 23.

Table 23 – <some>Type Definition

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Attribute** | | **Value** | | | | |
| BrowseName | | <some>Type | | | | |
| IsAbstract | | False | | | | |
| ValueRank | | −1 (−1 = Scalar) | | | | |
| DataType | | String | | | | |
| **References** | **NodeClass** | | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the BaseDataVariableType defined in … | | | | | | |
| 0:HasComponent | Variable | | <var1> | 0:UtcTime | 0:BaseDataVariableType | 0:Mandatory |
| 0:HasComponent | Variable | | <var2> | 0:UtcTime | 0:BaseDataVariableType | 0:Mandatory |

# OPC UA DataTypes

## <someStructure>

This structure contains …. The structure is defined in Table 24.

Table 24 – <someStructure> Structure

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| <someStructure> | structure | Subtype of <someParentStructure> defined in … |
| SP1 | 0:Byte[] | Setpoint 1 |
| SP2 | 0:Byte[] | Setpoint 2 |

Its representation in the *AddressSpace* is defined in Table 25.

The *AddressSpace* definition can be omitted if isAbstract=False and there are no *Properties*.

Table 25 – <someStructure> Definition

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Attribute** | | **Value** | | | | |
| BrowseName | | <someStructure> | | | | |
| IsAbstract | | False | | | | |
| **References** | **NodeClass** | | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the <someParentStructure> defined in … | | | | | | |

## <someUnion>

This union contains …. The union is defined in Table 26.

Table 26 – <someUnion> Union

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| <someUnion> | union |  |
| Var\_1 | 0:String | First set |
| Var\_2 | <someStructure> | Second set |
| Var\_3 | <someEnumeration> | Third set |

Its representation in the *AddressSpace* is defined in Table 27.

Table 27 – <someUnion> Definition

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Attributes** | | **Value** | | | | |
| BrowseName | | <someUnion> | | | | |
| IsAbstract | | False | | | | |
| **References** | **NodeClass** | | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of Union defined in OPC 10000-5. | | | | | | |

## <someEnumeration>

This enumeration …. The enumeration is defined in Table 28.

Table 28 – <someEnumeration> Items

|  |  |  |
| --- | --- | --- |
| Name | Value | Description |
| <Enum1\_Name> | 0 | <Enum1Description> |
| <Enum2\_Name> | 1 | <Enum2Description> |
| <Enum3\_Name> | 2 | <Enum4Description> |

Each *Enumeration* item is represented by a "Name" - the human readable representation and a "Value" - the numeric representation. If the *Enumeration* is zero-based and sequential, the *EnumStrings Property* is used for the names. In all other cases the *EnumValues Property* has to be used.

Its representation in the AddressSpace is defined in Table 29.

The *AddressSpace* definition can be omitted if isAbstract=False and there are no *Properties* other than *EnumStrings*.

Table 29 – <someEnumeration> Definition

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Attribute** | | **Value** | | | | |
| BrowseName | | <someEnumeration> | | | | |
| IsAbstract | | False | | | | |
| **References** | **NodeClass** | | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the Enumeration type defined in OPC 10000-5 | | | | | | |
| 0:HasProperty | Variable | | 0:EnumStrings | 0:LocalizedText [] | 0:PropertyType |  |

## <someOptionSet>

This *DataType* defines flags for … *<*someOptionSet*>* is formally defined in Table 30.

Table 30 – <someOptionSet> Values

|  |  |  |
| --- | --- | --- |
| **Value** | **Bit No.** | **Description** |
| <Value1> | 0 | This flag…. |
| <Value2> | 1 | This flag…. |
| <Value3> | 2 | This flag…. |

The *<*someOptionSet*>* representation in the *AddressSpace* is defined in Table 31.

Table 31 – <someOptionSet> Definition

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Attribute** | | **Value** | | | | |
| BrowseName | | <someOptionSet> | | | | |
| IsAbstract | | False | | | | |
| **References** | **NodeClass** | | **BrowseName** | **DataType** | **TypeDefinition** | **Other** |
| Subtype of the OptionSet DataType defined in OPC 10000-5 | | | | | | |
| 0:HasProperty | Variable | | 0:OptionSetValues | 0:LocalizedText [] | 0:PropertyType |  |

# OPC UA ReferenceTypes

## <someReferenceType>

The <someReferenceType> is a concrete *ReferenceType* and can be used directly. It is a subtype of <someParentReferenceType>.

The semantic of this *ReferenceType* is to link …...

The *SourceNode* of *References* of this type shall be an…...

The *TargetNode* of this *ReferenceType* shall be an …..

The *<*someReferenceType*>* is formally defined in Table 32.

Table 32 – <someReferenceType> Definition

|  |  |  |  |
| --- | --- | --- | --- |
| **Attributes** | **Value** | | |
| BrowseName | <someReferenceType> | | |
| InverseName | <someinverseName> | | |
| Symmetric | <True/False> | | |
| IsAbstract | <True/False> | | |
| **References** | **NodeClass** | **BrowseName** | **Comment** |
| Subtype <someParentReferenceType> | | | |

# Instances

## <someInstance>

The *<*someInstance*>* is formally defined in Table 33.

Table 33 – <someInstance> Definition

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute** | **Value** | | | |
| BrowseName | *<*someInstance*>* | | | |
| **References** | **NodeClass** | **BrowseName** | **DataType** | **TypeDefinition** |
| OrganizedBy by the <TheLocationInAddressSpace> defined in <Where It is Defined> | | | | |
| 0:HasTypeDefinition | <class of SomeInstance> | *<Type of someInstance>* | Defined in <Where Type of SomeInstance isdefined> | |

Provide some description of the instance, what it is used for, constraints on it etc

# Profiles and Conformance Units

*Profiles* and *ConformanceUnits* break functionality into testable groups. All companion specification shall include at least one *Profile*/*Facet*. If there are any groupings of functionality that not all *Servers*/*Client* would implement then multiple *Profile*/*Facet* are encouraged. A *ConformanceUnit* should describe a testable unit. A single *ConformanceUnit* is tested as a unit so all items covered by it must be support or the *ConformanceUnit* will fail. *ConformanceUnits* can be included in multiple *Profiles*, thus they are declared in their own table.

The name of the *Profile* should end with *Facet* or *Profile*. A *Facet* is a grouping of functionality that must also be paired with other *Facets* to create a running *Server* or *Client*. A *Profile* is all inclusive, in that is the *Profile* is implemented no additional functionality would be required to have a running application.

**<short name>**

A <short name> is required for each companion specification to assure uniqueness of string identifiers. It precedes the names of Profiles and ConformanceUnits and is included in URIs and URLs defined in a companion specification.

A <short name> is all caps if an acronym, otherwise camel case.

Exception if the short name is a trademark. Use trademark casing.

## Conformance Units

Table 34 defines the corresponding *ConformanceUnits* for the OPC UA Information Model for <title>.

Table 34 – Conformance Units for <Title>

| **Category** | **Title** | **Description** |
| --- | --- | --- |
| Server | ProbeType | The *ProbeType* node is available in the AddressSpace. Supports nodes that conform  to the (subtypes of) ProbeType.This node has to include all mandatory components of the ProbeType and may include the optional components. The instance(s) of the (subtypes of)  ProbeType is/are available in the AddressSpace [Location] in Model. |
| Server | ProbeTipType | The *ProbeTipType* node is available in the AddressSpace. Supports nodes that conform  to the (subtypes of) ProbeTipType.This node has to include all mandatory components of the ProbeTipType and may include the optional components. The instance(s) of the (subtypes of)  ProbeTipType is/are available in the AddressSpace [Location] in Model. |
| Server | LoadingMonitoringType | The *LoadingMonitoringType* node is available in the AddressSpace. Supports nodes that conform  to the (subtypes of) LoadingMonitoringType.This node has to include all mandatory components of the LoadingMonitoringType and may include the optional components. The instance(s) of the (subtypes of)  LoadingMonitoringType is/are available in the AddressSpace [Location] in Model. |
| Server | ToolMonitoringType | The *ToolMonitoringType* node is available in the AddressSpace. Supports nodes that conform  to the (subtypes of) ToolMonitoringType.This node has to include all mandatory components of the ToolMonitoringType and may include the optional components. The instance(s) of the (subtypes of)  ToolMonitoringType is/are available in the AddressSpace [Location] in Model. |
| Server | GMSEquipmentType | The *GMSEquipmentType* node is available in the AddressSpace. Supports nodes that conform  to the (subtypes of) GMSEquipmentType.This node has to include all mandatory components of the GMSEquipmentType and may include the optional components. The instance(s) of the (subtypes of)  GMSEquipmentType is/are available in the AddressSpace [Location] in Model. |
| Server | GMSIdentificationType | The *GMSIdentificationType* node is available in the AddressSpace. Supports nodes that conform  to the (subtypes of) GMSIdentificationType.This node has to include all mandatory components of the GMSIdentificationType and may include the optional components. The instance(s) of the (subtypes of)  GMSIdentificationType is/are available in the AddressSpace [Location] in Model. |
| Server | GMSOperationMonitoringType | The *GMSOperationMonitoringType* node is available in the AddressSpace. Supports nodes that conform  to the (subtypes of) GMSOperationMonitoringType.This node has to include all mandatory components of the GMSOperationMonitoringType and may include the optional components. The instance(s) of the (subtypes of)  GMSOperationMonitoringType is/are available in the AddressSpace [Location] in Model. |
| Server | GMSType | The *GMSType* node is available in the AddressSpace. Supports nodes that conform  to the (subtypes of) GMSType.This node has to include all mandatory components of the GMSType and may include the optional components. The instance(s) of the (subtypes of)  GMSType is/are available in the AddressSpace [Location] in Model. |
| Server | GMSMonitoringType | The *GMSMonitoringType* node is available in the AddressSpace. Supports nodes that conform  to the (subtypes of) GMSMonitoringType.This node has to include all mandatory components of the GMSMonitoringType and may include the optional components. The instance(s) of the (subtypes of)  GMSMonitoringType is/are available in the AddressSpace [Location] in Model. |
| Server | GMSJobType | The *GMSJobType* node is available in the AddressSpace. Supports nodes that conform  to the (subtypes of) GMSJobType.This node has to include all mandatory components of the GMSJobType and may include the optional components. The instance(s) of the (subtypes of)  GMSJobType is/are available in the AddressSpace [Location] in Model. |
| Server | GMSPartType | The *GMSPartType* node is available in the AddressSpace. Supports nodes that conform  to the (subtypes of) GMSPartType.This node has to include all mandatory components of the GMSPartType and may include the optional components. The instance(s) of the (subtypes of)  GMSPartType is/are available in the AddressSpace [Location] in Model. |
| Server | LMS\_ProductionStatisticsType | The *LMS\_ProductionStatisticsType* node is available in the AddressSpace. Supports nodes that conform  to the (subtypes of) LMS\_ProductionStatisticsType.This node has to include all mandatory components of the LMS\_ProductionStatisticsType and may include the optional components. The instance(s) of the (subtypes of)  LMS\_ProductionStatisticsType is/are available in the AddressSpace [Location] in Model. |
| Server | CalibrationPrognosisType | The *CalibrationPrognosisType* node is available in the AddressSpace. Supports nodes that conform  to the (subtypes of) CalibrationPrognosisType.This node has to include all mandatory components of the CalibrationPrognosisType and may include the optional components. The instance(s) of the (subtypes of)  CalibrationPrognosisType is/are available in the AddressSpace [Location] in Model. |
| Server | ResultManagmentType | The *ResultManagmentType* node is available in the AddressSpace. Supports nodes that conform  to the (subtypes of) ResultManagmentType.This node has to include all mandatory components of the ResultManagmentType and may include the optional components. The instance(s) of the (subtypes of)  ResultManagmentType is/are available in the AddressSpace [Location] in Model. |
| Server | ResultType | The *ResultType* node is available in the AddressSpace. Supports nodes that conform  to the (subtypes of) ResultType.This node has to include all mandatory components of the ResultType and may include the optional components. The instance(s) of the (subtypes of)  ResultType is/are available in the AddressSpace [Location] in Model. |
| Server | RotaryTableType | The *RotaryTableType* node is available in the AddressSpace. Supports nodes that conform  to the (subtypes of) RotaryTableType.This node has to include all mandatory components of the RotaryTableType and may include the optional components. The instance(s) of the (subtypes of)  RotaryTableType is/are available in the AddressSpace [Location] in Model. |
| Server | SensorListType | The *SensorListType* node is available in the AddressSpace. Supports nodes that conform  to the (subtypes of) SensorListType.This node has to include all mandatory components of the SensorListType and may include the optional components. The instance(s) of the (subtypes of)  SensorListType is/are available in the AddressSpace [Location] in Model. |
| Server | <short name> <Function2> | Supports the …... |
| Server | <short name> <Function3> | Supports the …... |
| Client | <short name> Client <Function1> | The client can make use of the …... |

Typically, *Client* *ConformanceUnits* describe the use of a function, but they do not need to match 1 to 1 with *Server* *ConformanceUnits*. They might also reference to other categories defined in Part 7 (Pub, Sub, GDS…). For larger companion specifications, there might be separate tables for *Client* *ConformanceUnits*, *Server* *ConformanceUnits*, etc.

## Profiles

### Profile list

Table 35 lists all Profiles defined in this document and defines their URIs.

Table 35 – Profile URIs for <Title>

| **Profile** | **URI** |
| --- | --- |
| <short name> <Prf1name> Server Profile | [http://opcfoundation.org/UA-Profile/<short](http://opcfoundation.org/UA-Profile/%3cshort) name>/Server/<Prf1name> |
| <short name> <Prf2name> Server Facet | [http://opcfoundation.org/UA-Profile/<short](http://opcfoundation.org/UA-Profile/%3cshort) name>/Server/<Prf2name> |
| <short name> <Prf3name> Client Facet | [http://opcfoundation.org/UA-Profile/<short](http://opcfoundation.org/UA-Profile/%3cshort) name>/Client/<Prf3name> |

### Server Facets

#### Overview

The following sections specify the *Facets* available for *Servers* that implement the <title> companion specification. Each section defines and describes a *Facet* or *Profile*.

A specification can define multiple *Facets* if not all features are to be implemented by all *Servers* and *Clients*. The name of the *Facet* shall give a hint of the subset. An overall description shall be provided that explains the subset and it potential use.

#### <short name> <Prf1name> Server Profile

Table 36 defines a *Profile* that describes the …….

Table 36 - <short name> <Prf1name> Server Profile

| **Group** | **Conformance Unit / Profile Title** | **Mandatory / Optional** |
| --- | --- | --- |
| Profile | 0:Core 2017 Server Facet http://opcfoundation.org/UA-Profile/Server/Core2017Facet |  |
| Profile | 0:UA-TCP UA-SC UA Binary http://opcfoundation.org/UA-Profile/Transport/uatcp-uasc-uabinary |  |
| Profile | 0:Data Access Server Facet http://opcfoundation.org/UA-Profile/Server/DataAccess |  |
| Profile | 2:BaseDevice\_Server\_Facet |  |
| Profile | <short name> <Prf2name> Server Facet |  |
| Subscription Services | 0:Subscription Durable | M |
| <short name> | <short name> <Function1> | M |

This table lists a *Profile*, in which it includes other base *Profiles* that would be needed to make a working *Server*. It also includes other *Facets* defined in this companion specification and *ConformanceUnits* defined in this companion standard.

A namespace shall be included if Profiles or ConformanceUnits of another specification are included. In the example above '0' represents the OPC UA core specification and '2' UA for Devices (see Table 41).

The column with title "Mandatory / Optional" defines whether support of included *ConformanceUnits* is optional or mandatory. Optional means that an application has the option to not support the *ConformanceUnit*. However, if supported, the application shall pass all tests associated with the *ConformanceUnit*.

The "Group" for all ConformanceUnits defined in this document shall be the <short name>. If ConformanceUnits of OPC 10000-7 are referenced, the corresponding Groups shall be used. See the example with group "Subscription Services".

#### <short name><Prf2name> Server Facet

Table 37 defines a *Facet* that describes the …….

Table 37 - <short name> <Prf2name> Server Facet

| **Group** | **Conformance Unit / Profile Title** | **Mandatory / Optional** |
| --- | --- | --- |
| <short name> | <short name> <Function1> | M |
| <short name> | <short name> <Function3> | O |

This table lists a *Facet*, in that it must be include with other *Facets* to create a running application. It defines the *ConformanceUnits* and other facets that are required

### Client Facets

#### Overview

The following tables specify the *Facets* available for *Clients* that implement the <title> companion specification.

A specification can define multiple facets if not all features are to be implemented by all *Servers* and *Clients*. The name of the facet shall give a hint of the subset. An overall description shall be provided that explains the subset and it potential use.

#### <short name> < Prf3name> Client Facet

Table 38 defines a *Facet* that describes the base characteristics for all OPC UA *Clients* that make use of this companion specification. Additional *Profiles* will define support for various information models that are part of this document.

Table 38 - <short name> < Prf3name> Client Facet

| **Group** | **Conformance Unit / Profile Title** | **Mandatory / Optional** |
| --- | --- | --- |
| Profile | 0:AddressSpace Lookup Client Facet http://opcfoundation.org/UA-Profile/Client/AddressSpaceLookup |  |
| Profile | 0:DataAccess Client Facet http://opcfoundation.org/UA-Profile/Client/DataAccess |  |
| Profile | 0:DataChange Subscriber Client Facet http://opcfoundation.org/UA-Profile/Client/DataChangeSubscriber |  |
| Session Services | 0:Session Client Detect Shutdown | M |
| <short name> | <short name> Client <Function1> | M |

This table lists a *Facet*, in that it must be include with other *Facets* to create a running application. It defines the *ConformanceUnits* and other facets that are required as an example it include other base Facets and a Base system *ConformanceUnit*

# Namespaces

## Namespace Metadata

Namespace Metadata are required for any companion standard that specifies an information model (e.g. *Objects* and *ObjectTypes*). The metadata provide standardized information about the elements of this namespace. This information is particularly important for aggregating *Servers*.

Typically, all Nodes of a companion specification are static and therefore the metadata shall describe them as static. This is done by setting all Numeric NodeIds to static (StaticNodeIdTypes). If you use different NodeIds (e.g. Strings), this needs to be adapted. If not all Nodes are static, it needs to be adapted as well. Static NodeIds mean, that the same Node is used in all servers, e.g. for TypeDefinitions or entry points like the “Root” Object of the base specification. Not static Nodes would be Nodes providing server-specific information (e.g. typically all the instances based on the TypeDefinitions of a companion specification) or other dynamic behaviour (e.g. a standardized Method that adds or removes something from a server).

Table 39 defines the namespace metadata for this document. The *Object* is used to provide version information for the namespace and an indication about static *Nodes*. Static *Nodes* are identical for all *Attributes* in all *Servers*, including the *Value Attribute*. See OPC 10000-5 for more details.

The information is provided as *Object* of type *NamespaceMetadataType*. This *Object* is a component of the *Namespaces* *Object* that is part of the *Server Object*. The *NamespaceMetadataType ObjectType* and its *Properties* are defined in OPC 10000-5.

The version information is also provided as part of the ModelTableEntry in the UANodeSet XML file. The UANodeSet XML schema is defined in OPC 10000-6.

Table 39 – NamespaceMetadata Object for this Document

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Value** | | |
| BrowseName | [http://opcfoundation.org/UA/<short](http://opcfoundation.org/UA/%3cshort) name>/ | | |
| **Property** | | **DataType** | **Value** |
| NamespaceUri | | String | [http://opcfoundation.org/UA/<short](http://opcfoundation.org/UA/%3cshort) name> |
| NamespaceVersion | | String | X.YY |
| NamespacePublicationDate | | DateTime | YYYY-MM-DD |
| IsNamespaceSubset | | Boolean | False |
| StaticNodeIdTypes | | IdType [] | undefined |
| StaticNumericNodeIdRange | | NumericRange [] | Null |
| StaticStringNodeIdPattern | | String | Null |

Note: The *IsNamespaceSubset* *Property* is set to False as the UaNodeSet XML file contains the complete Namespace. *Servers* only exposing a subset of the Namespace need to change the value to True.

## Handling of OPC UA Namespaces

Namespaces are used by OPC UA to create unique identifiers across different naming authorities. The *Attributes* *NodeId* and *BrowseName* are identifiers. A *Node* in the UA *AddressSpace* is unambiguously identified using a *NodeId*. Unlike *NodeIds*, the *BrowseName* cannot be used to unambiguously identify a *Node*. Different *Nodes* may have the same *BrowseName*. They are used to build a browse path between two *Nodes* or to define a standard *Property*.

*Servers* may often choose to use the same namespace for the *NodeId* and the *BrowseName*. However, if they want to provide a standard *Property*, its *BrowseName* shall have the namespace of the standards body although the namespace of the *NodeId* reflects something else, for example the *EngineeringUnits* *Property*. All *NodeIds* of *Nodes* not defined in this document shall not use the standard namespaces.

Table 40 provides a list of mandatory and optional namespaces used in an <title> OPC UA *Server*.

Table 40 – Namespaces used in a <title> Server

| **NamespaceURI** | **Description** | **Use** |
| --- | --- | --- |
| http://opcfoundation.org/UA/ | Namespace for *NodeIds* and *BrowseNames* defined in the OPC UA specification. This namespace shall have namespace index 0. | Mandatory |
| Local Server URI | Namespace for nodes defined in the local server. This namespace shall have namespace index 1. | Mandatory |
| http://opcfoundation.org/UA/DI/ | Namespace for *NodeIds* and *BrowseNames* defined in OPC 10000-100. The namespace index is *Server* specific. | Mandatory |
| http://opcfoundation.org/UA/<title>/ | Namespace for *NodeIds* and *BrowseNames* defined in this document. The namespace index is *Server* specific. | Mandatory |
| Vendor specific types | A *Server* may provide vendor-specific types like types derived from *ObjectTypes* defined in this document in a vendor-specific namespace. | Optional |
| Vendor specific instances | A *Server* provides vendor-specific instances of the standard types or vendor-specific instances of vendor-specific types in a vendor-specific namespace.  It is recommended to separate vendor specific types and vendor specific instances into two or more namespaces. | Mandatory |

Table 41 provides a list of namespaces and their indices used for *BrowseNames* in this document. The default namespace of this document is not listed since all *BrowseNames* without prefix use this default namespace.

Table 41 – Namespaces used in this document

| **NamespaceURI** | **Namespace Index** | **Example** |
| --- | --- | --- |
| http://opcfoundation.org/UA/ | 0 | 0:EngineeringUnits |
| http://opcfoundation.org/UA/DI/ | 2 | 2:DeviceRevision |
| http://opcfoundation.org/UA/Dictionary/IRDI/ | 3 | 3:0112/2///61987#xzx608 |

1. (normative)   
     
   <Title> Namespace and mappings
   1. Namespace and identifiers for <Title> Information Model

This appendix defines the numeric identifiers for all of the numeric *NodeIds* defined in this document. The identifiers are specified in a CSV file with the following syntax:

<SymbolName>, <Identifier>, <NodeClass>

Where the *SymbolName* is either the *BrowseName* of a *Type Node* or the *BrowsePath* for an *Instance Node* that appears in the specification and the *Identifier* is the numeric value for the *NodeId*.

The *BrowsePath* for an *Instance Node* is constructed by appending the *BrowseName* of the instance *Node* to the *BrowseName* for the containing instance or type. An underscore character is used to separate each *BrowseName* in the path. Let’s take for example, the *<type>* *ObjectType* *Node* which has the *<property> Property*. The **Name** for the *<property>* *InstanceDeclaration* within the *<type>* declaration is: *<type>\_<property>*.

A NamespaceURI follows the convention: [http://opcfoundation.org/UA/<short name>/](http://opcfoundation.org/UA/POWERLINK/).

<short name> is described in 14.

.

Note that NamespaceURIs are NOT live URLs. Text in the specification should not suggest that they are.

The *NamespaceUri* for all *NodeIds* defined here is [http://opcfoundation.org/UA/<short name>/](http://opcfoundation.org/UA/%3cshort%20name%3e/)

**File Locations**

The location of any version dependent files follow this convention:

[http://opcfoundation.org/UA/schemas/<short name>/<version>/<file name>](http://opcfoundation.org/UA/schemas/%3cshort%20name%3e/%3cversion%3e/%3cfile%20name%3e)

The <short name> is the same as specified in the NamespaceURI;

The <version> is a number with the form #.# or #.##;

The location of the version independent files are the same but with the <version> omitted.

e.g. [http://opcfoundation.org/UA/schemas/<short name>/<file name>](http://opcfoundation.org/UA/schemas/%3cshort%20name%3e/%3cfile%20name%3e)

**File Names**

**NodeIds**: Opc.Ua.<short name>.NodeIds.csv or <short name>.NodeIds.csv

**NodeSet**: Opc.Ua.<short name>.NodeSet.xml or <short name>.NodeSet.xml;

Any other files should have a prefix that provides context when the file is downloaded in a browser.

All published files must be added to GitHub <https://github.com/OPCFoundation/UA-Nodeset>

This can be done by creating a mantis issue in the “NodeSets, XSDs and Generated Code” project:

<https://opcfoundation-onlineapplications.org/mantis/main_page.php>

The files should be attached to the mantis issue.

If the NodeSet was generated with the Opc.Ua.ModelCompiler the design file should be attached as well.

The CSV released with this version of the specification can be found here:

[http://www.opcfoundation.org/UA/schemas/<short name>/1.0/NodeIds.csv](http://www.opcfoundation.org/UA/schemas/%3cshort%20name%3e/1.0/NodeIds.csv)

NOTE    The latest CSV that is compatible with this version of the specification can be found here:

[http://www.opcfoundation.org/UA/schemas/<short name>/NodeIds.csv](http://www.opcfoundation.org/UA/schemas/%3cshort%20name%3e/NodeIds.csv)

A NodeIds.csv file is not mandated but recommended.

It contains a flat list of NodeIds with unique names and can be used instead of a full NodeSet if only such NodeId constants for a programming environment are needed.

A computer processible version of the complete Information Model defined in this document is also provided. It follows the XML Information Model schema syntax defined in OPC 10000-6.

The Information Model Schema for this version of the document (including any revisions, amendments or errata) can be found here:

[http://www.opcfoundation.org/UA/schemas/<short name>/1.0/Opc.Ua.<short name>.NodeSet2.xml](http://www.opcfoundation.org/UA/schemas/%3cshort%20name%3e/1.0/Opc.Ua.%3cshort%20name%3e.NodeSet2.xml)

NOTE    The latest Information Model schema that is compatible with this version of the document can be found here:

[http://www.opcfoundation.org/UA/schemas/<short name>/Opc.Ua.<short name>.NodeSet2.xml](http://www.opcfoundation.org/UA/schemas/%3cshort%20name%3e/Opc.Ua.%3cshort%20name%3e.NodeSet2.xml)

\_\_\_\_\_\_\_\_\_\_\_\_\_