CC-Link
CC-Link IE

Control & Communication System Profile Specification (for Machine) Part 5: OPC UA Information Model

CSP+ Ver.1.0
BAP-C2008ENG-002

Revisions

Sub number	Revision description	Issued
*	First edition	June, 2017

CONTENTS

FIGURES

TABLES

1. FOREWORD

This document describes the OPC UA information model in "Control & Communication System Profile Specification (for Machine)".

The Control & Communication System Profile for machine (hereinafter referred to as "CSP+ for machine") is a data set that visualizes machine information to simplify development by application vendors of application software that manages, monitors, and controls the machine, and settings by the machine users. The CSP+ for machine contains the following information related to the machine described.

- Information related to the machine specifications
- Machine information to be released for application software (machine information)
- Information related to data to be acquired from the machine and its acquisition method (machine data)
- Linked information between machine information and machine data

The CSP+ for machine is generally handled as CSP+ file for machine described in the XML format.

This document specifies the mapping of information written in the CSP+ for machine to the OPC UA information model. The specifications specified here are applied for the case where application software using the CSP+ for machine is an OPC UA server.

The version of Control & Communication System Profile for machine specification described in this document (hereinafter referred to as CSP+ for machine specification version) is version 1.0.

2. SCOPE OF APPLICATION

This document specifies mapping specifications for the OPC UA information model of the CSP+ for machine. The specifications specified in this document are recommended to be applied for the case where application software using the CSP+ for machine is an OPC UA server. The image of the application is shown in Figure 2-1.

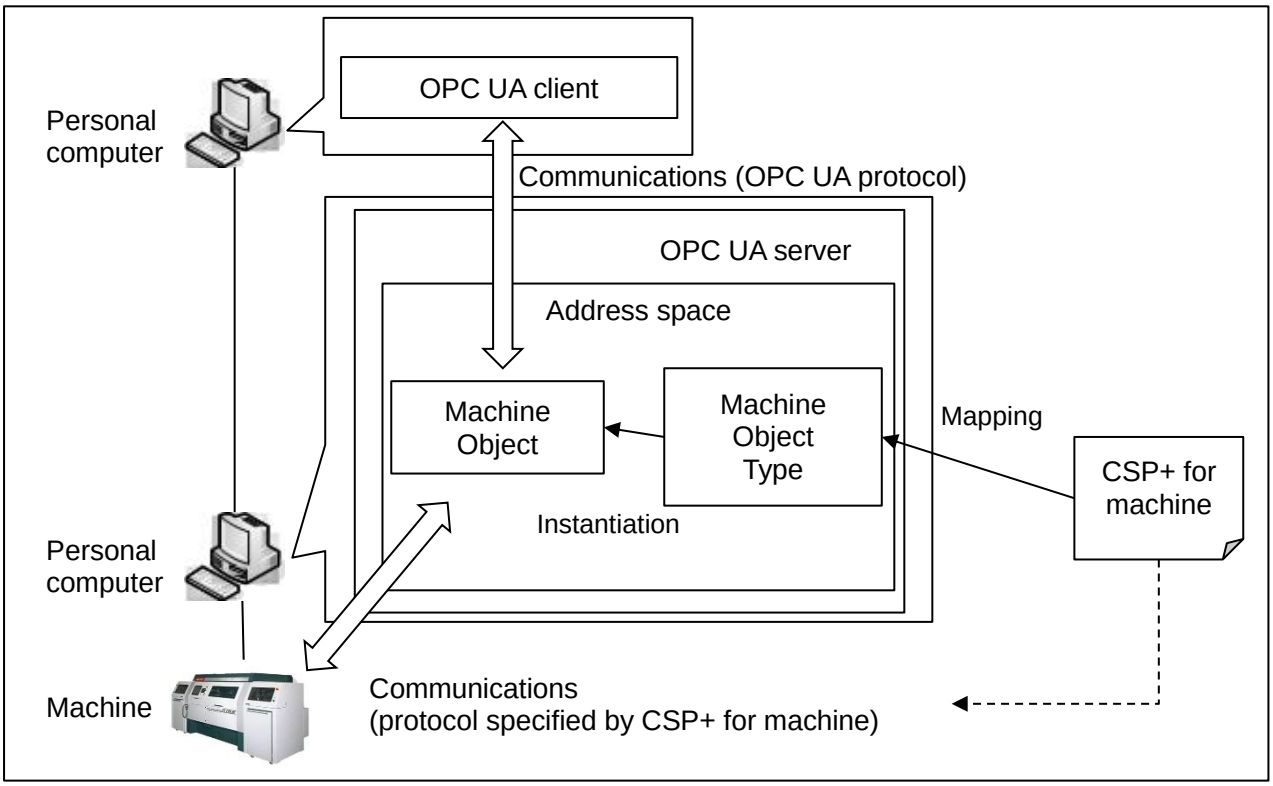

Figure 2-1 – Image of Application from CSP+ for Machine to OPC UA Server

3. NORMATIVE REFERENCES

IEC 62541-3:2015 OPC Unified Architecture - Part 3: Address Space Model
IEC 62541-4:2015 OPC Unified Architecture - Part 4: Services
IEC 62541-5:2015 OPC Unified Architecture - Part 5: Information Model
IEC 62541-6:2015 OPC Unified Architecture - Part 6: Mappings
IEC 62541-8:2015 OPC Unified Architecture - Part 8: Data Access
IEC 62541-100:2015 OPC Unified Architecture - Part 100: Device Interface

4. TERMINOLOGY, DEFINITIONS, ABBREVIATIONS

4.1. Terminology

4.1.1. CSP+ for machine

Data set to describe the following information related to the machine

- Information related to the machine specifications
- Machine information to be released for application software
- Data to be acquired from the machine and its acquisition method
- Linked information between machine information and machine data

4.1.2. CSPP section

Component of the CSP+ for machine

4.1.3. CSPP part

Component of the section

4.1.4. CSPP element

Component of the part

4.1.5. CSPP item

Detailed information related to the element. Example: Data type, engineering unit

4.2. Abbreviations and Symbols

CNC	Computer Numerical Control
CSP+	Control & Communication System Profile
PLC	Programmable Logic Controller

5. CSP+ FOR MACHINE AND OPC UA

5.1. Introduction to CSP+ for Machine

5.1.1. Overview

The CSP+ for machine is a data set that visualizes machine information to simplify development by application vendors of application software that manages, monitors, and controls the machine, and settings by the machine users. The CSP+ file for machine is CSP+ for machine described in the XML format. The CSP+ for machine contains the following information related to the machine.

- Information related to the machine specifications
- Machine information to be released for application software (machine information)
- Information related to data to be acquired from the machine and its acquisition method (machine data)
- Linked information between machine information and machine data

5.1.2. Basic structure of CSP+ for machine

5.1.2.1. CSPP section

The CSP+ for machine consists of four type CSPP sections: FILE section, DEVICE section, COMM_IF section, and BLOCK section. Overview of the CSPP sections is shown in Table 5-1.

Table 5-1 – Overview of Sections in CSP+ for Machine

CSPP section name	Overview	Number of sections
FILE	Describes management information for CSP+ file for machine.	1 section
DEVICE	Describes information such as machine name, identification information, and machine specifications.	1 section
COMM_IF	Describes definition information for the machine information.	1 or more sections
BLOCK	Describes definition information for the machine data.	1 or more sections

Figure 5-1 shows the structure image of the CSP+ for machine.

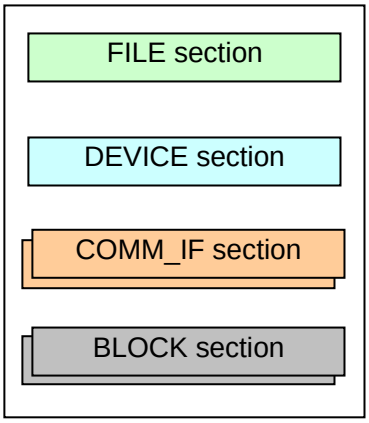

Figure 5-1 – Structure Image of CSP+ File for Machine

5.1.2.2. CSPP part

The CSPP sections consist of one or more CSPP parts. Figure 5-2 shows the structure image in a CSPP section.

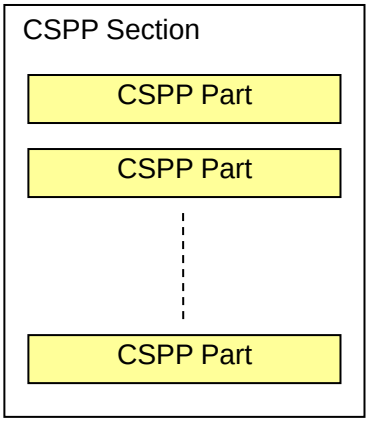

Figure 5-2 – Image of CSPP Section Structure

Type of CSPP parts included in the CSPP sections varies depending on the CSPP sections. Type of CSPP parts included in the FILE section is shown in Table 5-2, type of CSPP parts included in the DEVICE section is shown in Table 5-3, type of CSPP parts included in the COMM_IF section is shown in Table 5-4, and type of CSPP parts included in the BLOCK section is shown in Table 5-5.

Table 5-2 – CSPP Parts Included in FILE Section

CSPP part type	Information to be described	Number of parts
FILE_INFO	- Management information for CSP+ file for machine (e.g. Date of file created, language information, file version)	1 part

Table 5-3 – CSPP Parts Included in DEVICE Section

CSPP part type	Information to be described	Number of parts
DEVICE_INFO	- Machine identification information (e.g. Vendor name, model name) - Machine's product information (e.g. Specifications, image file name)	1 part
DEVICE_IF	- Information related to communications with the machine (e.g. Communications protocol type)	1 or more parts

Table 5-4 – CSPP Parts Included in COMM_IF Section

CSPP part type	Information to be described	Number of parts
COMM_IF_INFO	- Machine identification information	1 part
COMM_IF_VARIABLE	- Machine information for realtime monitor (e.g. Current value)	0 or more parts
COMM_IF_CONFIGURATION	- Machine information for general purpose (e.g. Power consumption for 30 minutes)	0 or more parts
ENUM	- Options for setting range	0 or more parts

Table 5-5 – CSPP Parts Included in BLOCK Section

CSPP part type	Information to be described	Number of parts
BLOCK_INFO	- Machine data identification	1 part
BLOCK_MEMORY	- Variable machine data acquired from the machine (e.g. Current value, measurement time)	0 or more parts
BLOCK_PARAM	- Machine-specific machine data not acquired from the machine (e.g. Accuracy, collection cycle)	0 or more parts
ENUM	- Options for setting range	0 or more parts

There are COMM_IF_VARIABLE part and COMM_IF_CONFIGURATION part as CSPP parts managing the machine information. Machine information for realtime monitor is described in the COMM_IF_VARIABLE part and machine information for general purpose is described in the COMM_IF_CONFIGURATION part. As the information allowed for the COMM_IF_VARIABLE part is limited for realtime monitor, description is easy but the application is limited. On the other hand, there is less restrictions on the description for the COMM_IF_CONFIGURATION part where the machine information for the general purpose applications can be described. For example, a window display layout includes a monitoring window for machine information in the COMM_IF_VARIABLE part and a machine parameter read/write window for machine information in the COMM_IF_CONFIGURATION part.

5.1.3. CSPP element and CSPP item

The CSPP parts consist of one or more CSPP elements. The CSPP elements consist of one or more CSPP items. Figure 5-3 shows the image of the CSPP part structure.

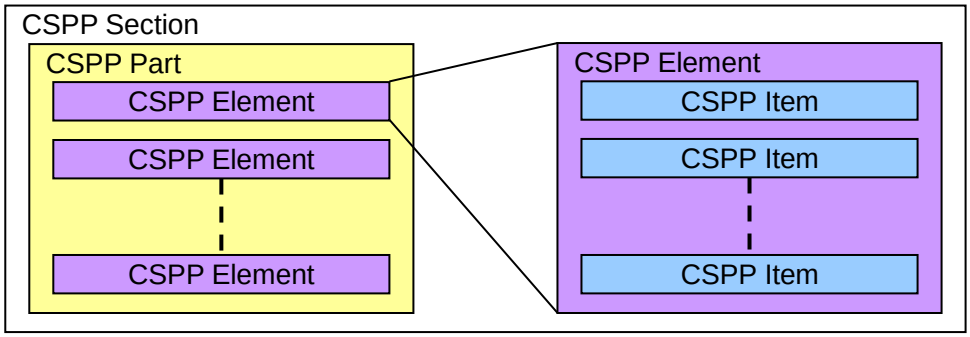

Figure 5-3 – Image of CSPP Part Structure

CSPP elements included in the CSPP parts consist of CSPP elements specified as specifications of CSP+ for machine and CSPP elements that can be flexibly specified by the machine vendors. In contrast, all the CSPP items included in the CSPP elements are specified by specifications of the CSP+ for machine.

5.1.4. Machine information and machine data

Association information is set from one CSPP element in the COMM_IF_CONFIGURATION part managing machine information to one or more machine data in the BLOCK_MEMORY part or BLOCK_PARAM part. This means that aggregation of more than one machine data creates one machine information. The reason why this structure is required is that the information required by application software needs not only a simple measured value (current value) but also the accompanying information such as the time when the value is measured and accuracy of the value.

In contrast, setting association information from the CSPP element in the COMM_IF_VARIABLE part as the machine information to the machine data in the BLOCK_MEMORY part or BLOCK_PARAM part is not allowed. This is because the application is for realtime monitor and all the information required in the machine information can be described. Association between the machine information and machine data is illustrated in Figure 5-4.

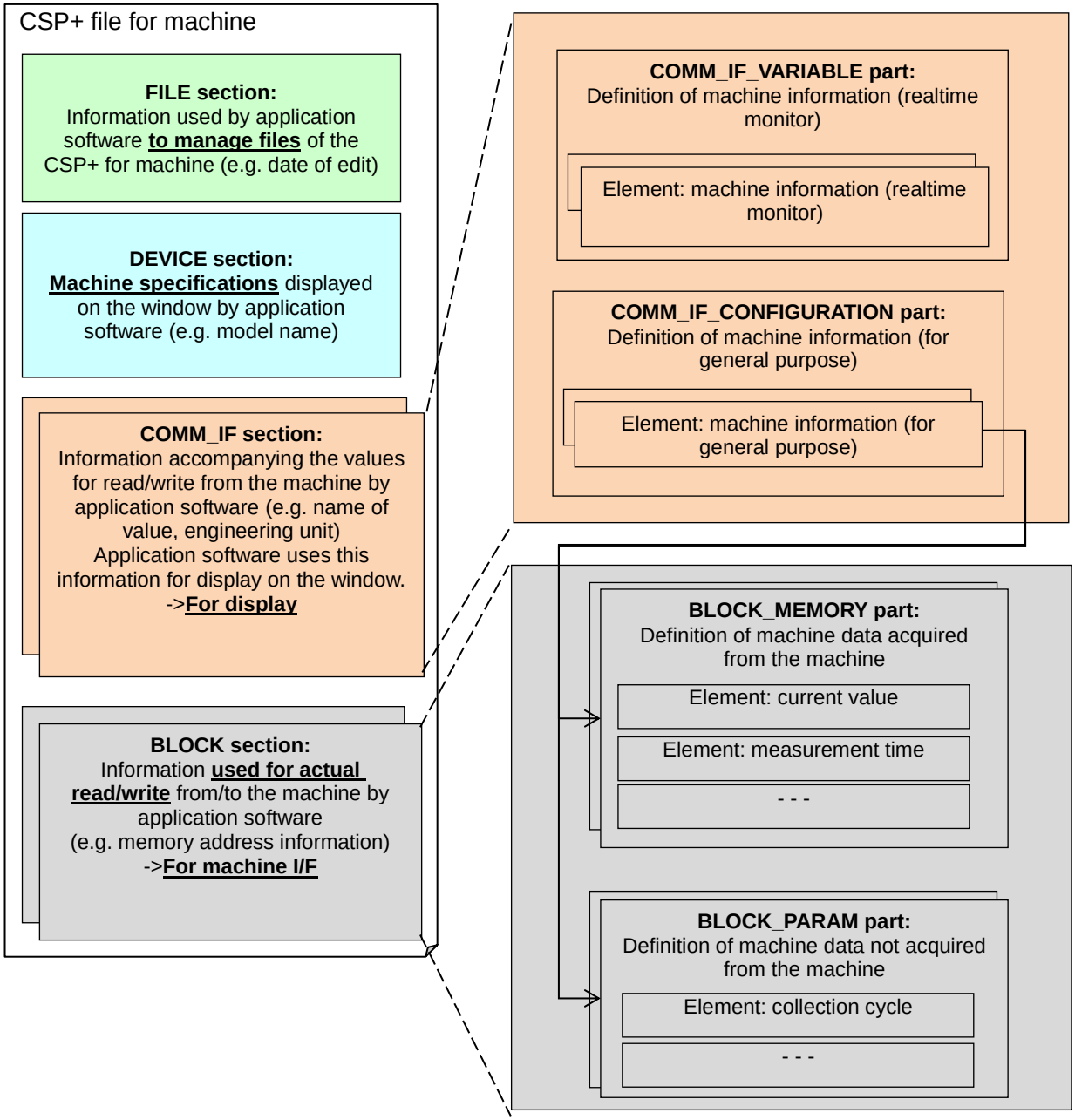

Figure 5-4 – Structure Image of CSP+ for Machine (Association between Machine Information and Machine Data)

5.2. Introduction to OPC Unified Architecture

5.2.1. General

The main use case for OPC classic specifications is the online data exchange between devices and HMI or SCADA systems using Data Access functionality. In this use case the device data is provided by an OPC server and is consumed by an OPC client integrated into the HMI or SCADA system. OPC DA provides functionality to browse through a hierarchical namespaces containing data items and to read, write and to monitor these items for data changes. The classic OPC standards are based on Microsoft COM/DCOM technology for the communication between software components from different vendors. Therefore classic OPC server and clients are restricted to Windows PC based automation systems.

OPC UA incorporates all features of classic OPC standards like OPC DA, A&E and HDA but defines platform independent and secure communication mechanisms and generic, extensible and object-oriented modelling capabilities for the information a system wants to expose. OPC UA is directly integrated into devices and is used for configuration, diagnostic and maintenance use cases in addition to online data exchange. OPC UA is an integrated communication interface used from sensor level devices up to enterprise applications.

IEC 62541-6:2015 defines different mechanisms optimized for different use cases. The first version of OPC UA is defining an optimized binary TCP protocol for high performance intranet communication as well as a mapping to accepted internet standards like Web Services. The abstract communication model does not depend on a specific protocol mapping and allows adding new protocols in the future. Features like security and reliability are directly built into the transport mechanisms. Based on the platform independence of the protocols, OPC UA servers and clients can be directly integrated into devices and controllers.

The OPC UA *Information Model* provides a standard way for *Servers* to expose *Objects* to *Clients*. *Objects* in OPC UA terms are composed of other *Objects*, *Variables* and *Methods*. OPC UA also allows relationships to other *Objects* to be expressed.

The set of *Objects* and related information that an OPC UA *Server* makes available to *Clients* is referred to as its *AddressSpace*. The elements of the OPC UA *Object* Model are represented in the *AddressSpace* as a set of *Nodes* described by *Attributes* and interconnected by *References*. OPC UA defines eight classes of *Nodes* to represent *AddressSpace* components. The classes are *Object*, *Variable*, *Method*, *ObjectType, DataType, ReferenceType* and *View*. Each *NodeClass* has a defined set of *Attributes*.

This specification defines *Nodes* of the OPC UA *NodeClasses Object, Method, Variable, ObjectType and DataType*.

Objects are used to represent components of a system. An *Object* is associated to a corresponding *ObjectType* that provides definitions for that *Object*.

Methods are used to represent commands or services of a system.

Variables are used to represent values. Two categories of *Variables* are defined, *Properties* and *DataVariables*.

Properties are *Server*-defined characteristics of *Objects*, *DataVariables* and other *Nodes*. *Properties* are not allowed to have *Properties* defined for them.

DataVariables represent the data contents of an *Object*.

5.2.2. Graphical notation

OPC UA defines a graphical notation for an OPC UA *AddressSpace*. It defines graphical symbols for all *NodeClasses* and how different types of *References* between *Nodes* can be visualized. Figure 5-5 shows the symbols for the six *NodeClasses* used in this specification. *NodeClasses* representing types always have a shadow.

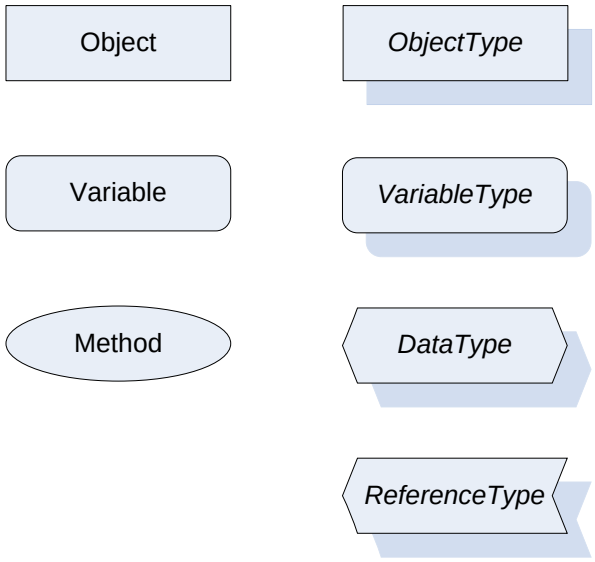

Figure 5-5 – OPC UA Graphical Notation for NodeClasses

Figure 5-6 shows the symbols for the *ReferenceTypes* used in this specification. The *Reference* symbol is normally pointing from the source *Node* to the target *Node*. The only exception is the *HasSubType Reference*. The most important *References* like *HasComponent*, *HasProperty*, *HasTypeDefinition* and *HasSubType* have special symbols avoiding the name of the *Reference*. For other *ReferenceTypes* or derived *ReferenceTypes* the name of the *ReferenceType* is used together with the symbol.

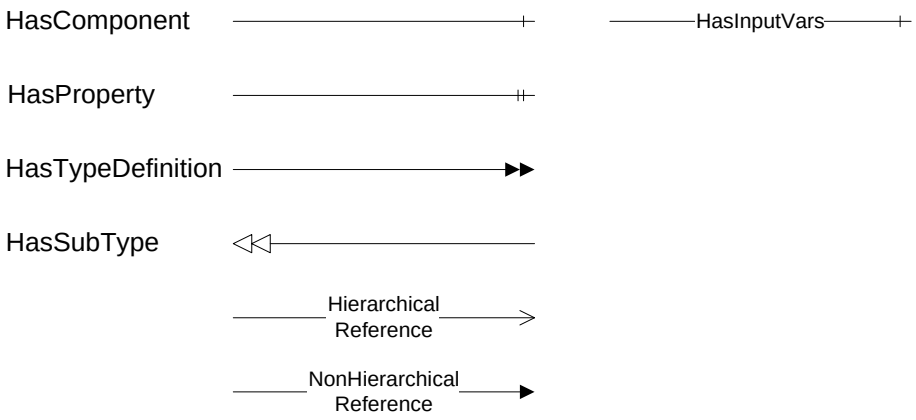

Figure 5-6 – OPC UA Graphical Notation for References

Figure 5-7 shows a typical example for the use of the graphical notation. Object_A and Object_B are instances of the ObjectType_Y indicated by the *HasTypeDefinition References*. The ObjectType_Y is derived from ObjectType_X indicated by the HasSubType *Reference*. The Object_A has the components Variable_1, Variable_2 and Method_1.

To describe the components of an *Object* on the *ObjectType* the same *NodeClasses* and *References* are used on the *Object* and on the *ObjectType* like for ObjectType_Y in the example. The instance *Nodes* used to describe an *ObjectType* are instance declaration *Nodes*.

To provide more detailed information for a *Node*, a subset or all *Attributes* and their values can be added to a graphical symbol.

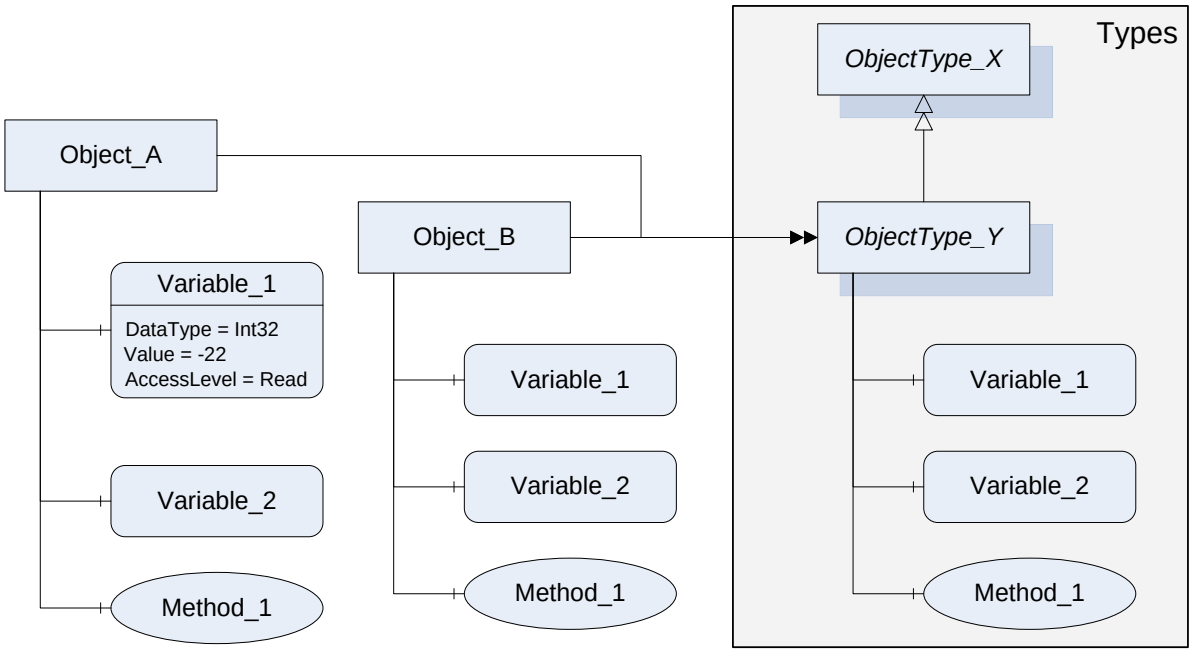

Figure 5-7 – OPC UA Graphical Notation Example

5.3. Use Case

This section describes an example of the OPC UA server according to Figure 5-8, as an application example of the application software using the CSP+ for machine.
This example shows that the OPC UA server itself as the application software using the CSP+ for machine does not provide the function for the machine users and other application software such as SCADA and MES provides the monitoring and management functions for the machine users. When the application software using the CSP+ for machine is an OPC UA server, the address space that the OPC UA server discloses to the OPC UA clients is created from the CSP+ for machine based on the specifications specified in this document.

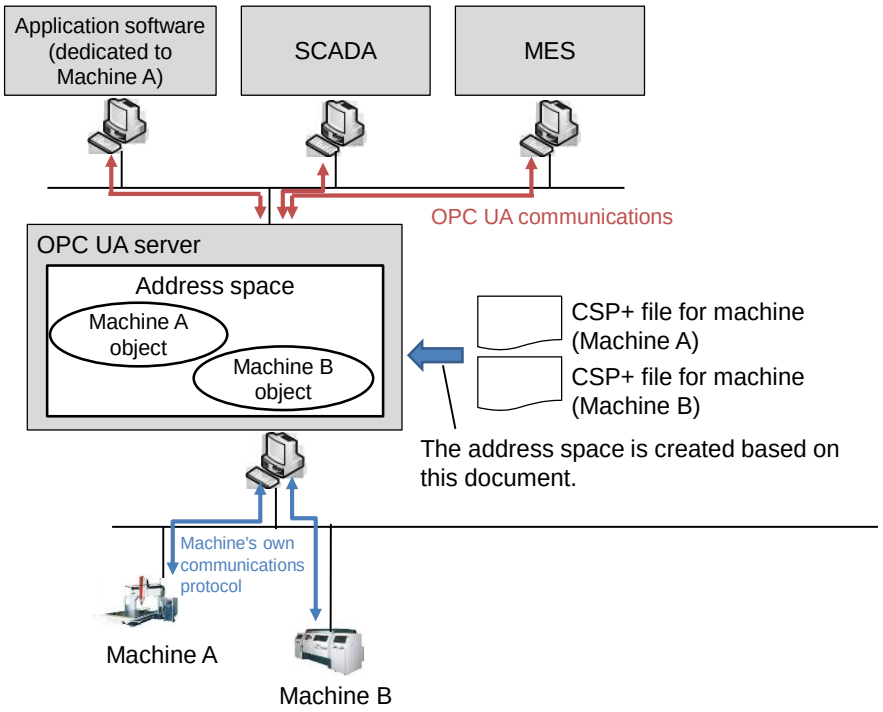

Figure 5-8 – Operation Image of OPC UA Server Supporting CSP+ for Machine

Since this system configuration is achievable, vendors of application software such as SCADA and MES have the following advantage.

- Since not a machine's own communications protocol but the standard OPC UA can be used for communications with machines, the development volume for the communication function can be reduced.

Vendors of OPC UA server application software have the following advantage.

- Supporting the CSP+ for machine allows handling many machines supporting the CSP+ for machine as communication partners.

Machine vendors have the following advantages.

- Since there is no need to request an application vendor to support the machine's own protocol when requesting the application vendor to develop application software dedicated to the machine, the development cost can be reduced.
- Since information to be disclosed to the application software such as process statuses and operation histories of the machine can be limited to the range described in the CSP+ for machine, information related to know-how of the machine can be set to confidential.

6. MACHINE MODEL

6.1. Overview

This chapter specifies the OPC UA information model for machines described by the CSP+ for machine.

6.1.1. Basic policy

This section defines *CsppMachineType ObjectType* corresponding to all the machines described by the CSP+ for machine, and further defines dedicated *ObjectType* corresponding to separate machines described by each CSP+ for machine (hereinafter referred to as "CSP+ machine *ObjectType*"). The *CsppMachineType ObjectType* should be abstract *ObjectType* (IsAbstract attribute is "true") inheriting DeviceType *ObjectType* specified by the IEC 62541-100:2015. The CSP+ machine *ObjectType* should be non-abstract *ObjectType* (IsAbstract attribute is "false") inheriting *CsppMachineType ObjectType*. While inheriting, *Node* definition corresponding to the information described in the CSP+ for machine is added.

The machine information described by the CSP+ for machine is described as an Object of the CSP+ machine *ObjectType*.

6.1.2. Model positioning

Positioning of OPC UA information model specified in this document is shown in Figure 6-1. In addition to the *CsppMachineType ObjectType* corresponding to the machine, *CsppAnalogItemType* as *VariableType* for the *CsppMachineType ObjectType* is defined.

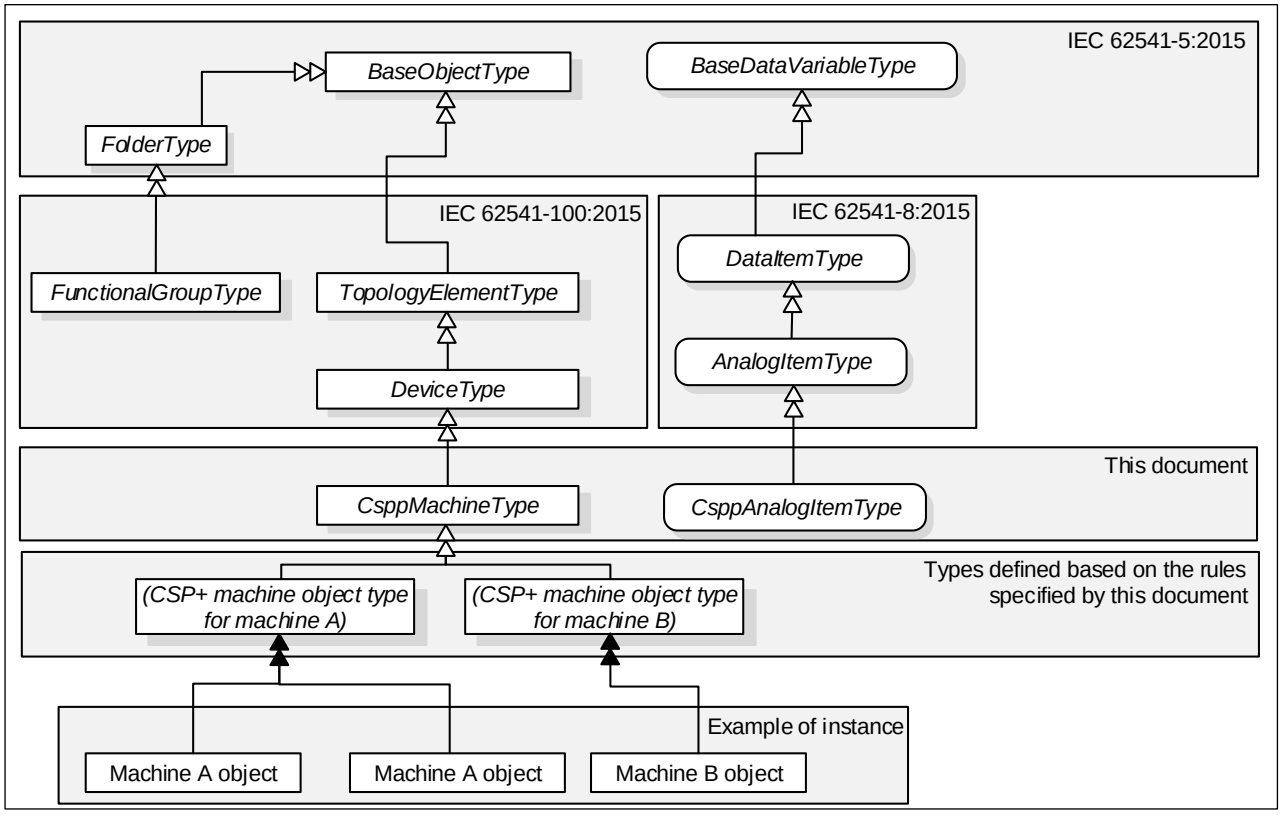

Figure 6-1 – Positioning of OPC UA Information Model Related to Machine

6.2. Type Definition

6.2.1. CsppMachineType ObjectType

This section describes the *CsppMachineType ObjectType* definition and *Node* definition referred to from the *CsppMachineType ObjectType*. Figure 6-2 shows a full picture of the *CsppMachineType ObjectType*.

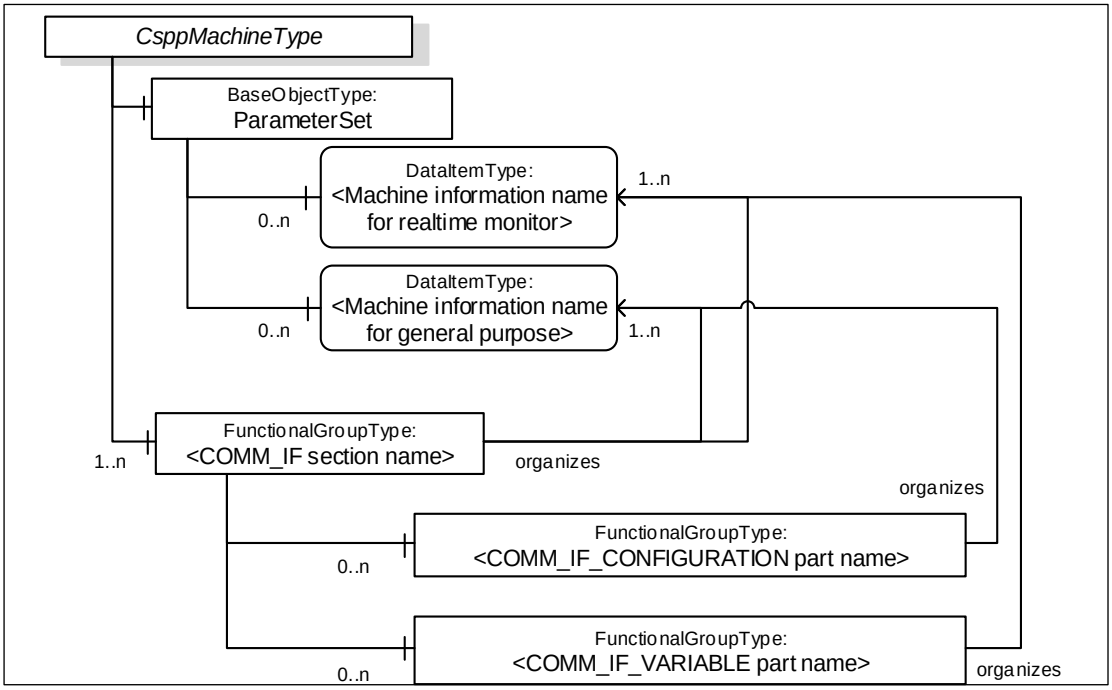

Figure 6-2 – Full Picture of CsppMachineType ObjectType

6.2.1.1. CsppMachineType ObjectType

Definition of the *CsppMachineType ObjectType* is shown in Table 6-1.

Table 6-1 – Definition of CsppMachineType ObjectType

Attribute	**Value**					
BrowseName	CsppMachineType					
IsAbstract	True					
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**	**Details**
Subtype of the DeviceType defined in IEC 62541-100:2015						
HasComponent	Object	1:ParameterSet	-	BaseObjectType	Mandatory	6.2.1.2
HasComponent	Object	<CommIfSection>	-	FunctionalGroup Type	MandatoryPlaceholder	6.2.1.3

ModellingRule of the *ParameterSet Object* is Optional in the source of the inheritance but that of the *CsppMachineType ObjectType* should be Mandatory. The definition of the *ParameterSet Object* itself is overridden. For details, refer to 6.2.1.2.

*Object*s shown by <CommIfSection> correspond to COMM_IF section of the CSP+ for machine and take a role to group the machine information held by the machine.

6.2.1.2. ParameterSet Object

The *ParameterSet Object* takes a role to organize machine information held by the machine *Object*. Table 6-2 shows the definition of the *ParameterSet Object*.

Table 6-2 – Definition for ParameterSet Object

Attribute	**Value**					
BrowseName	1:ParameterSet					
IsAbstract	False					
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**	**Details**
HasTypeDefinition	ObjectType	0:BaseObjectType				
HasComponent	Variable	<VariableName>	(Any)	DataItemType	OptionalPlaceholder	-
HasComponent	Variable	<ConfigurationName>	(Any)	DataItemType	OptionalPlaceholder	-

*DataVariable*s shown by <VariableName> correspond to the CSPP elements in the COMM_IF_VARIABLE part of the CSP+ for machine. The *VariableType* is DataItemType *VariableType* specified by the IEC 62541-8:2015 or its derivative type.

*DataVariable*s shown by <ConfigurationName> correspond to the CSPP elements in the COMM_IF_CONFIGURATION part of the CSP+ for machine. The *VariableType* is DataItemType *VariableType* specified by the IEC 62541-8:2015 or its derivative type.

6.2.1.3. Object corresponding to COMM_IF section

Object corresponding to the COMM_IF section takes a role to make machine *Object* group the machine information per CSPP section. In addition, it takes a role to organize the COMM_IF_VARIABLE part and COMM_IF_CONFIGURATION part in the COMM_IF section. Table 6-3 shows the definition of the *Object*.

Table 6-3 – Definition of Object Corresponding to COMM_IF Section

Attribute	**Value**					
BrowseName	(name of COMM_IF section)					
IsAbstract	False					
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**	**Details**
HasTypeDefinition	ObjectType	1:FunctionalGroupType				
HasComponent	Object	<CommIfVariablePart>	-	FunctionalGroup Type	OptionalPlaceholder	6.2.1.4
HasComponent	Object	<CommIfConfiguration Part>	-	FunctionalGroup Type	OptionalPlaceholder	6.2.1.5
Organizes	Variable	<VariableOrConfigurat ionName>	(Any)	DataItemType	MandatoryPlaceholder	-

*Object*s shown by <CommIfVariablePart> correspond to COMM_IF_VARIABLE part of the CSP+ for machine and take a role to group machine information for realtime monitor held by the machine.

*Object*s shown by <CommIfConfigurationPart> correspond to COMM_IF_CONFIGURATION part of the CSP+ for machine and take a role to group machine information for general purpose held by the machine.

6.2.1.4. Object corresponding to COMM_IF VARIABLE part

Object corresponding to the COMM_IF_VARIABLE part takes a role to group the machine information for realtime monitor held by the machine *Object* per part. Table 6-4 shows the definition of the *Object*.

Table 6-4 – Definition of Object Corresponding to COMM_IF VARIABLE Part

Attribute	**Value**					
BrowseName	(name of COMM_IF_VARIABLE part)					
IsAbstract	False					
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**	**Details**
HasTypeDefinition	ObjectType	1:FunctionalGroupType				
Organizes	Variable	<VariableName>	(Any)	DataItemType	MandatoryPlaceholder	-

6.2.1.5. Object corresponding to COMM_IF_CONFIGURATION part

Object corresponding to the COMM_IF_CONFIGURATION part takes a role to group the machine information for general purpose held by the machine *Object* per part. Table 6-5 shows the definition of the *Object*.

Table 6-5 – Definition of Object Corresponding to COMM_IF_CONFIGURATION Part

Attribute	**Value**					
BrowseName	(name of COMM_IF_CONFIGURATION part)					
IsAbstract	False					
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**	**Details**
HasTypeDefinition	ObjectType	1:FunctionalGroupType				
Organizes	Variable	<ConfigurationName>	(Any)	DataItemType	MandatoryPlaceholder	-

6.2.2. CsppAnalogItemType VariableType

This section defines the *CsppAnalogItemType VariableType* to describe machine information accompanying measurement period of the machine information described by the CSP+ for machine. Table 6-6 shows the definition of the *VariableType*.

Table 6-6 – Definition of CsppAnalogItemType VariableType

Attribute	**Value**					
BrowseName	CsppAnalogItemType					
IsAbstract	False					
ValueRank	-2 (-2 = 'Any')					
DataType	Number					
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**	**Details**
SubType of the AnalogItemType defined in IEC 62541-8:2015						
HasProperty	Variable	Duration	Number	PropertyType	Mandatory	-

The *Duration Property* shows the duration in which applicable variable is calculated or measured in millisecond unit.

6.2.3. CSP+ machine ObjectType

Definition of the CSP+ machine *ObjectType* is shown in Table 6-7.

Table 6-7 – Definition of CSP+ Machine ObjectType

Attribute	**Value**					
BrowseName	(DEVICE section LABEL of the corresponding CSP+)					
IsAbstract	False					
References	**NodeClass**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**	**Details**
Subtype of the *CsppMachineType* defined in 6.2.1						

Definition of a *Node* referred to from the CSP+ machine *ObjectType* is determined by the description of the CSP+ for machine corresponding to applicable CSP+ machine *ObjectType*. Rules for the mapping to determine the description of the CSP+ for machine are shown in Chapter 7.

6.3. Machine Object and Entry Point

The machine *Object* indicating one machine should be an instantiated CSP+ machine *ObjectType* corresponding to the applicable machine and referred to by using *HasComponent Reference* from the *DeviceSet* entry point specified by the IEC 62541-100:2015. Figure 6-3 illustrates the relationship among the entry point, machine *Objects*, and CSP+ machine *ObjectType*. Figure 6-3 illustrates three machines, two machine *Object*s for the machine A, and one machine *Object* for the machine B.

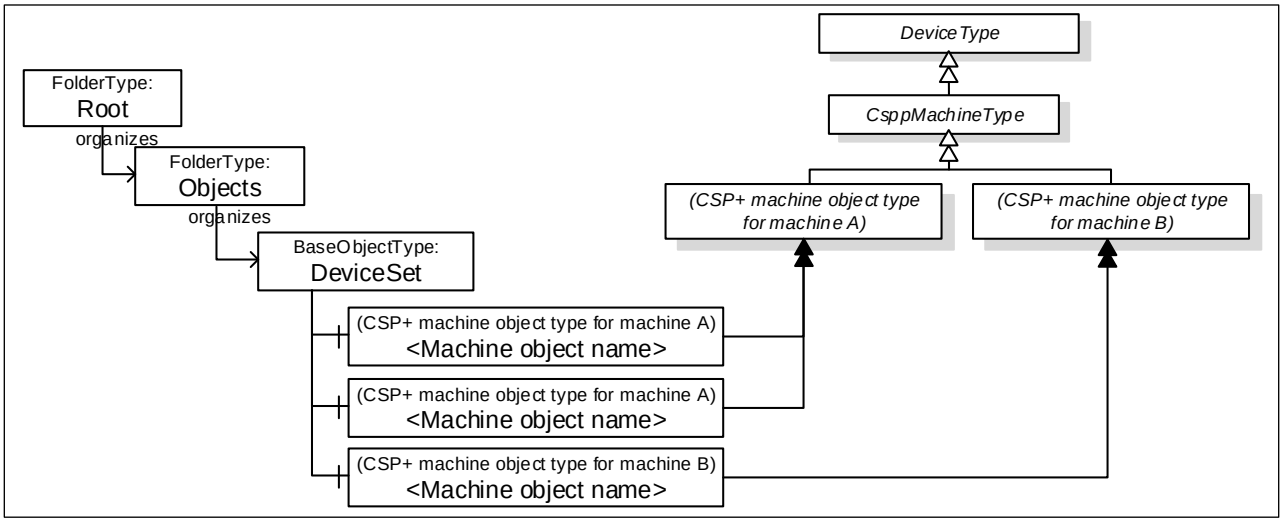

Figure 6-3 – Image of Relationship among Entry Point, Machine Object, and CSP+ Machine ObjectType

7. MAPPING

7.1. Overview

This chapter specifies rules to create CSP+ machine *ObjectType* from the *CsppMachineType ObjectType* based on the information described in the CSP+ for machine.

Inheriting rules not specified in this document for mandatory *Attribute*s, *Properties* and *DataVariable*s of the *CsppMachineType ObjectType* should depend on the implementation of application creating CSP+ machine *ObjectType* (OPC UA *Server*). The instantiation of machine *Object* from the CSP+ machine *ObjectType* should also depend on the implementation of application (OPC UA *Server*).

7.2. FILE Section

No rules are specified to use the information described in the FILE section to create CSP+ machine *ObjectType*. However, DATA item value of Language element as one of COMMON information may be indirectly used to show locale of a string. For details, refer to specifications of each rule.

7.3. DEVICE Section

7.3.1. Header information

Label names described in the DEVICE section header are used as part of *BrowseName Attribute* values of the CSP+ machine *ObjectType*. Specifically, a text "CsppDeviceType" connected after an *Attribute* value is handled as the *BrowseName Attribute* value. For example, when the label name is "ABC", its *BrowseName Attribute* value becomes "ABCCsppDeviceType".

No rules are specified to use any information other than the above to create CSP+ machine *ObjectType*.

7.3.2. DEVICE_INFO part

7.3.2.1. Header information

No rules are specified to use the information described in the DEVICE_INFO part header to create CSP+ machine *ObjectType*.

7.3.2.2. COMMON information CSPP element

Part of COMMON information CSPP elements in the DEVICE_INFO part is mapped to *Attribute*s and *Properties* of the CSP+ machine *ObjectType*.

Table 7-1 shows the mapping specifications.

Table 7-1 – Mapping Specifications for COMMON Information CSPP Elements of DEVICE_INFO Part

No.	LABEL name of CSPP element (mapping source)	Mapping destination	Rule
1.	VendorName	Manufacturer *Property*	Maps DATA item value (Unicode string) of the CSPP element to the *Value Attribute* value of the *Property*.
2.	DeviceModel	Model *Property*	Maps DATA item value (ASCII string) of the CSPP element to the *Value Attribute* value of the *Property*.
3.	ProductID	(No mapping)	-
4.	Version	DeviceRevision *Property*	Maps DATA item value (*1) of the CSPP element to the *Value Attribute* value of the *Property* as a string.
5.	ReferenceURL	DeviceManual *Property*	-
6.	URLInfo	(No mapping)	-
7.	Outline	Description *Attribute*	Maps DATA item value (Unicode string) of the CSPP element to the *Attribute* value.
8.	SpecList	(No mapping)	-
9.	IconFileName	Icon *Property* (ImagePNG *DataType*)	Converts a file specified by DATA item of the CSPP element to a PNG file and maps the file content as a binary string to the *Value Attribute* value of the *Property*.
10.	GraphicsFileName	*DataVariable* referred to from DeviceTypeImage *Object*. The *DataVariable* name should be DATA item value of DeviceModel element.	Maps a file content specified by DATA item of the CSPP element as a binary string to the *Value Attribute* value of the *DataVariable*. *DataType* of the *DataVariable* should be a *DataType* depending on the file format (PNG->ImagePNG *DataType*, BMP->ImageBMP *DataType*, JPG->ImageJPG *DataType*, GIF->ImageGIF *DataType*).

*1: One of bit string type, signed integral data type, unsigned integral data type, or STRING (x)

7.3.2.3. CSPP element other than COMMON information

No rules are specified to use any CSPP elements other than COMMON information to create CSP+ machine *ObjectType*.

7.3.3. DEVICE_IF part

No rules are specified to use the information described in the DEVICE_IF part to create CSP+ machine *ObjectType*.

7.4. COMM_IF Section and BLOCK Section

7.4.1. Basic policy

Basic policy to apply information described in the COMM_IF section and BLOCK section for creating CSP+ machine *ObjectType* is as follows.

- CSPP elements specified in the COMM_IF_VARIABLE part and COMM_IF_CONFIGURATION part are mapped to each *DataVariable* referred to from *ParameterSet Object* of CSP+ machine *ObjectType* by using *HasComponent Reference*.
- The *DataVariable*s above are grouped both per CSPP section and per CSPP part. *FunctionalGroupType Object*s are created only for the number of CSPP sections and CSPP parts for grouping purpose, the *Object*s corresponding to the CSPP sections are referred to from the CSP+ machine *ObjectType* by using *HasComponent Reference*, and the *Object*s corresponding to the CSPP parts are referred to from the *Object*s corresponding to the sections using the *HasComponent Reference*.
- *Attribute* and *Property* information held by the *DataVariable*s (e.g. engineering unit, minimum collection cycle) are configured based on the CSPP elements in the BLOCK_MEMORY part and BLOCK_PARAM part referred to from the element corresponding to the *DataVariable*s, and CSPP element or CSPP part to which the CSPP element belongs.

The image of the basic policy is shown in Figure 7-1.

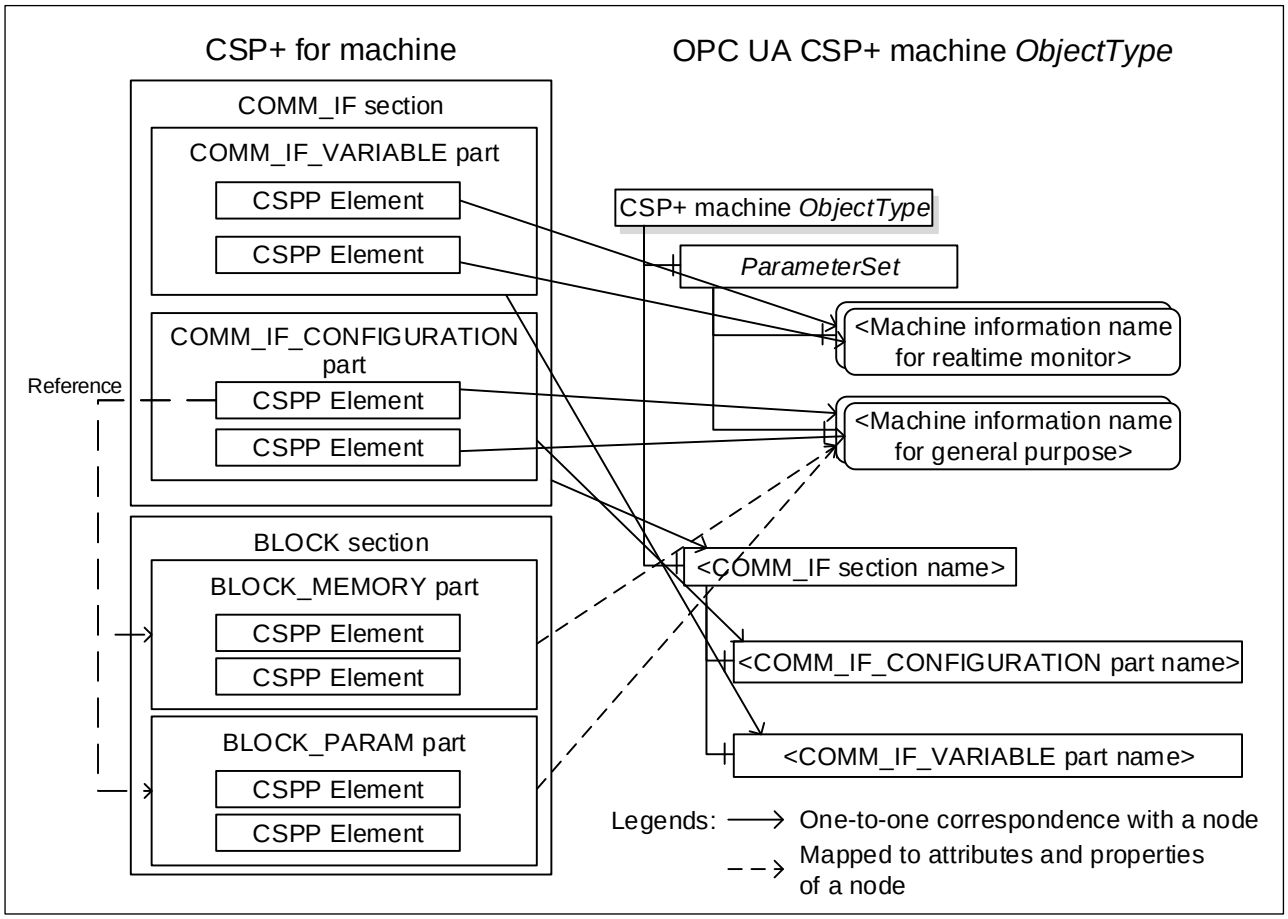

Figure 7-1 – Image of Relationship between COMM_IF Section, BLOCK Section, and CSP+ Machine ObjectType

7.4.2. COMM_IF section header information

From the information described in the COMM_IF section header, LABEL name is mapped to the *BrowseName Attribute* of the *FunctionalGroupType Object* (refer to 6.2.1.3) and LABEL2 name is mapped to the *DisplayName Attribute* of the same *Object* above.

7.4.3. BLOCK section header information

No rules are specified to use the information described in the BLOCK section header to create CSP+ machine *ObjectType*.

7.4.4. COMM_IF_INFO part

No rules are specified to use the information described in the COMM_IF_INFO part to create CSP+ machine *ObjectType*.

7.4.5. BLOCK_INFO part

No rules are specified to use the information described in the BLOCK_INFO part to create CSP+ machine *ObjectType*.

7.4.6. COMM_IF_VARIABLE part, COMM_IF_CONFIGURATION part

7.4.6.1. Header information

From the information described in the COMM_IF_VARIABLE part and COMM_IF_CONFIGURATION part headers, LABEL name is mapped to the *BrowseName Attribute* of the *FunctionalGroupType Object* (refer to 6.2.1.4 and 6.2.1.5) and LABEL2 name is mapped to the *DisplayName Attribute* of the same *Object* above.

No rules are specified to use any information other than the above to create CSP+ machine *ObjectType*.

7.4.6.2. CSPP element

CSPP elements specified in the COMM_IF_VARIABLE part and COMM_IF_CONFIGURATION part are mapped to each *DataVariable* referred to from *ParameterSet Object* of CSP+ machine *ObjectType* by using *HasComponent Reference*. The *VariableType* of each *DataVariable* is determined based on the following rules depending on the item values of the BLOCK_MEMORY part and BLOCK_PARAM part CSPP elements referred to from item values of each CSPP element and the CSPP elements.

- CSPP elements that specify ENUM to the RANGE item are mapped to the *DataVariable*s in one of *VariableType*s: *TwoStateDiscreteType*, *MultiStateDiscreteType*, and *MultiStateValueDiscreteType*.
  - Data type is boolean type: *TwoStateDiscreteType VariableType*
  - CODE item values of entire elements are assigned in a serial number starting from 0 in ENUM part: *MultiStateDiscreteType VariableType*
  - Any case other than the above: *MultiStateValueDiscreteType VariableType*
- All the CSPP elements other than the above referring to BLOCK_PARAM part including P_Period element are mapped to the *DataVariable* of the *CsppAnalogItemType VariableType*.
- All the CSPP elements other than the above in data type of bit string type, signed integral data type, unsigned integral data type, BCD integral data type, or real data type, are mapped to the *DataVariable* of the *AnalogItemType VariableType*.
- All the CSPP elements other than the above are mapped to the *DataVariable* of the *DataItemType VariableType*.

Part of items of each CSPP element is mapped to *Attribute*s and *Properties* of the *DataVariable*s. Table 7-2 shows the mapping specifications. Instead of description omission in the COMM_IF_VARIABLE part and COMM_IF_CONFIGURATION part, the mapping specifications shown in the Table 7-2 are also applied to the items described in the P_Value element of the BLOCK_MEMORY part.

Table 7-2 – Mapping Specifications for COMM_IF_VARIABLE Part and COMM_IF_CONFIGURATION Part Elements

No.	Item name of CSPP element (mapping source)	Mapping destination	Rule
1.	LABEL	*BrowseName Attribute*	Maps the string that consists of the item value followed by a decimal value representing the appearance sequence (initial value: 1) of the item which has the same LABEL name in the CSP+ file for machine, to an *Attribute* value. Example: When the LABEL name is "ABC" and the appearance sequence is 12, the *Attribute* value is "ABC12".
2.	LABEL2	*DisplayName Attribute*	Maps the item value to the *Attribute* value.
3.	CATEGORY	(No mapping)	-
4.	NAME	(No mapping)	-
5.	DATATYPE	*DataType Attribute*	Refer to Table 8-1.
6.	RANGE	When values and value ranges are directly described EURange *Property* When option list (ENUM) is used Refer to 7.4.8.	Mapping is performed only when one type of value setting is specified. No mapping is performed when two or more types of values are specified or when a value is specified. Since the Range as a data type of EURange *Property* cannot identify whether it is closed interval or open interval, upper limit is mapped to the high element of the Value *Attribute* and lower limit is mapped to the low element without the identification.
7.	MIN_INC	(Affected to Value *Attribute* indirectly)	Refer to No.1 of Table 7-3.
8.	ENG_UNIT	EngineeringUnits *Property*	Maps the item value to the *displayName* element of the Value *Attribute* of the *Property*.
9.	ACCESS	*AccessLevel Attribute*	Maps the item value to an *Attribute* value based on the following rules. This item is not present in the COMM_IF_VARIABLE part element; however, it is handled as the item value is R. - Item value is "R": Set bit 0 to 1, bit 1 to 0 - Item value is "W": Set bit 0 to 0, bit 1 to 1 - Item value is "RW": Set bit 0 to 1, bit 1 to 1 - Item value is "NA": Set bit 0 to 0, bit 1 to 0 - Item value is empty: No rules applied
10.	REF_MEMORY(*1)	Refer to 7.4.7.	-
11.	REF_PARAM(*1)	Refer to 7.4.7.	-
12.	ASSIGN(*2)	(No mapping)	Does not map any item values but maps the values acquired from the machine using this item value to the Value *Attribute* value of the *DataVariable*.
13.	COMMENT	*Description Attribute*	Maps the item value to the *Attribute* value.

*1: For the COMM_IF_CONFIGURATION part CSPP element only
*2: For the COMM_IF_VARIABLE part CSPP element only

7.4.7. BLOCK_MEMORY part, BLOCK_PARAM part

7.4.7.1. Header information

No rules are specified to use the information described in the BLOCK_MEMORY part and BLOCK_PARAM part headers to create CSP+ machine *ObjectType*.

7.4.7.2. Preset label CSPP element

Elements having preset labels in the BLOCK_MEMORY part and BLOCK_PARAM part are mapped not to CSP+ machine *ObjectType* but to a machine *Object* with the CSP+ machine *ObjectType* instantiated.

Some of elements having preset labels in the BLOCK_MEMORY part and BLOCK_PARAM part are mapped to *Attribute*s and *Properties* of *DataVariable*s as mapping destination for the elements specified in the COMM_IF_CONFIGURATION as its reference source, and others are mapped to return values for the *Services* (*Read*, *Publish*, *Republish*) to be executed for the *DataVariable*s. Table 7-3 shows the mapping specifications.

Table 7-3 – Mapping Specifications for BLOCK_MEMORY Part and BLOCK_PARAM Part CSPP Elements

No.	LABEL name of CSPP element (mapping source)	Mapping destination	Rule
1.	P_Value	*Value Attribute* of the *DataVariable*	Maps the value acquired from the machine to the *Attribute* value. However, when MIN_INC item has been defined in P_Value element or its reference source element, a value acquired from the machine is converted depending on the MIN_INC item value and mapped to the *Attribute* value.
2.	P_NA	*Severity* field (bit 30 and bit 31) of the *statusCode* element of a value as *DataValue DataType*[*1] from the *Service* return values	When the value acquired from the machine is 0 (not missing), the Severity should be "00" (Good Success). When the value acquired is 1 (missing), the Severity should be "10" (Bad Failure).
3.	P_Accuracy	*ValuePrecision Property* for *DataVariable*	Maps a value acquired from the machine (BLOCK_MEMORY part) or an element DATA item value (BLOCK_PARAM part) to a *Value Attribute* value of a *Property*.
4.	P_ChangeDate	"sourceTimestamp" member of a value as *DataValue DataType*[*1] from the *Service* return values	Maps the item value acquired from the machine to the member value.
5.	P_MeasurementDate	"serverTimestamp" element of a value as *DataValue DataType*[*1] from the *Service* return values	Maps the item value acquired from the machine to the element value.
6.	P_Period	*Duration Property* for *DataVariable*	Converts the value acquired from the machine (the unit is specified by the ENG_UNIT item value for the element) in millisecond unit and maps to the *Attribute* value of the *Property*.
7.	P_Cycle	*MinimumSampleingInterval Attribute* for *DataVariable*	Maps DATA item value of the element to the *Attribute* value.

*1: For a *Read Service*, the results[] value corresponds to the *DataValue DataType*. For a *Publish* or *Republish Service*, the *Value* element of the notificationData[] element in the notificationMessage value corresponds to the *DataValue DataType*.

7.4.7.3. CSPP element other than preset label

No rules are specified to use the information for the elements other than the preset label in the BLOCK_MEMORY part and BLOCK_PARAM part to create CSP+ machine *ObjectType*.

7.4.8. ENUM part

7.4.8.1. Header information

No rules are specified to use the information described in the ENUM part header to create CSP+ machine *ObjectType*.

7.4.8.2. CSPP element

7.4.8.2.1. TwoStateDiscreteType VariableType

When the data type of reference source CSPP element in the ENUM part is boolean type, map the reference source CSPP element to the *DataVariable* of the *TwoStateDiscreteType VariableType*. Map two CSPP elements of the ENUM part to *Properties* of the *DataVariable* based on the following rules.

- Assign the CSPP element with the CODE item value 0 to the *FalseState Property* and map the CSPP element with CODE item value 1 to the *TrueState Property*.
- When the LABEL2 item of the CSPP element is specified, set the item value to the text element of the *Value Attribute* for corresponding *Property*. Set the DATA item value of the Language element in the FILE_INFO part to the *Locale* element.
- When the LABEL2 item of the CSPP element is not specified, set the LABEL item value to the text element of the *Value Attribute*. Set "en-US" to the *Locale* element.

7.4.8.2.2. MultiStateDiscreteType VariableType

When CODE item values of entire CSPP elements of ENUM part are assigned in a serial number starting from 0, map the reference source CSPP element to the *DataVariable* of the *MultiStateDiscreteType VariableType*. Map the CSPP elements of the ENUM part to *Properties* of the *DataVariable* based on the following rules.

- Sort the CSPP elements in the ascending order of the CODE item value to be assigned to each element of the *EnumStrings Property* (Note: the *Property* above is array type) in a sequential order.
- When the LABEL2 item of the CSPP element is specified, set the item value to the text element of the *Value Attribute* for the corresponding element of the *EnumStrings Property*. Set the DATA item value of the Language element in the FILE_INFO part to the *Locale* element.
- When the LABEL2 item of the CSPP element is not specified, set the LABEL item value to the text element. Set "en-US" to the *Locale* element.

7.4.8.2.3. MultiStateValueDiscreteType VariableType

When CODE item values of entire CSPP elements of ENUM part are not assigned in a serial number starting from 0, map the reference source CSPP element to the *DataVariable* of the *MultiStateValueDiscreteType VariableType*. Map the CSPP elements of the ENUM part to the *DataVariable Property* based on the following rules.

- Assign the elements of the *EnumValues Property* (Note: the *Property* above is array type) in the ascending order described in the ENUM part.
- Set the CODE item value of the CSPP element to the *Value* element of the *Value Attribute* for the corresponding element of the *EnumValues Property*.
- When the LABEL2 item of the CSPP element is specified, set the item value to the text element of the *Display* element in the *Value Attribute* for the corresponding element of the EnumValues *Property*. Set the DATA item value of the Language element in the FILE_INFO part to the locale element for the Display element.
- When the LABEL2 item of the CSPP element is not specified, set the LABEL item value to the text element. Set "en-US" to the *Locale* element.

8. DATA TYPE MAPPING

Mapping rules from data type in CSP+ for machine to the OPC UA *DataType* are shown in Table 8-1.

Table 8-1 – Mapping Rules between Data Type in CSP+ for Machine and OPC UA DataType

No.	Data type in CSP+ for machine (Left: Classification, Right: Data type)		OPC UA *DataType*	Remarks
1.	Boolean type	BOOL	Boolean	Map "0" in the CSP+ for machine to "FALSE" for OPC UA and map "1" in the CSP+ for machine to "TRUE" for the OPC UA.
2.	Binary type	BIN8	Int16	The intention "Display values in bit unit" by the binary type is described by the *ValueAsText Attribute*. Example: Set the *ValueAsText Attribute* to "00111010" in BIN8 or "0000000000111010" in BIN16 for the value 58 (0x3A).
3.		BIN16	Int16	
4.		BIN32	Int32	
5.		BINx (x = 1 to 15)	Int16	
6.	Bit string type (Hexadecimal)	BYTE	UInt16	*ValueAsText Attribute* is used to describe hexadecimal. Example: Set the *ValueAsText Attribute* to "0x3A" in BYTE or "0x003A" in WORD for the value 58 (0x3A).
7.		WORD	UInt16	
8.		DWORD	UInt32	
9.		BIT_STRINGx (x = 2 to 15)	UInt16	
10.	Signed integral data type (Decimal)	INT8	Int16	-
11.		INT16	Int16	
12.		INT32	Int32	
13.		INTx (x = 2 to 15)	Int16	
14.	Unsigned integral data type (Decimal)	UINT8	UInt16	-
15.		UINT16	UInt16	
16.		UINT32	UInt32	
17.		UINTx (x = 2 to 15)	UInt16	
18.	BCD integral data type (Decimal)	BCD8	UInt16	Numerical value notation in BCD as one decimal per 4 bits is described by the *ValueAsText Attribute*. Example: Set the *ValueAsText Attribute* to "58" in BYTE or "0058" in WORD for the value "58".
19.		BCD16	UInt16	
20.		BCD32	UInt32	
21.		BCDx (x = 4,12)	UInt16	
22.	Real data type (Decimal)	REAL	Float	-
23.		LREAL	Double	-
24.	String type	STRING(x) ("x" shows an integer from 1 to 2048.)	String	-
25.		STRING_U(x) ("x" shows an integer from 1 to 2048.)	String or LocalizedText	As the *String DataType* for OPC UA is in Unicode, mapping from STRING_U (x) data type in the CSP+ for machine is available. When a locale ID (e.g. "en-US") needs to be specified, the mapping should be conducted to the *LocalizedText DataType*.
26.	Time type	TIME	Int32	Use *EngineeringUnits Property* of the variable to specify a unit "ms" and use *ValueAsText Attribute* to describe a preset notation such as "T#3d11h45m15s123ms".
27.	Date type	DATE	DateTime	-
28.	Accuracy type	ACCURACY	Double	-
29.	IP address type	IP_V4	String	Describe the value in a string.
30.		IP_V4_64	String	Describe the value in a string.
31.	Aggregation type	Data type + "()" [Example] INT8(), STRING(10)()	-	As this document does not specify any rules to the CSP+ for machine element having this data type, no *DataType* mapping rules are specified.
32.	Array type	Data type + "["Number of elements +"]"	-	The OPC UA describes the array by the variable *ValueRank Attribute* (Number of array dimensions) and *ArrayDimensions Attribute* (Number of elements for each dimension).

9. PROFILE AND NAMESPACE

9.1. Namespace Metadata

Table 9-1 shows the namespace metadata for this specifications. This *Object* is used to provide the version information and static *Node*s of the namespace. For the static *Node*s, all *Attribute*s of all servers are the same, including value *Attribute*s. For details, refer to IEC 62541-5:2015.

Information on the namespace metadata is provided as *Object* of type *NamespaceMetadataType*. This Object is a component of *Namespaces* Object which is a part of the Server *Object*. The *NamespaceMetadataType ObjectType* and its *Property* are defined in IEC 62541-5:2015.

The version information is also provided as a part of *ModelTableEntry* in the UANodeSet XML file. The UANodeset XML schema is defined in IEC 62541-6:2015.

Table 9-1 – NamespaceMetadata Object for this Specification

Attribute	Value		
BrowseName	http://opcfoundation.org/UA/CSPPlusForMachine/		
References	BrowseName	DataType	Value
HasProperty	NameSpaceUri	String	http://opcfoundation.org/UA/CSPPlusForMachine/
HasProperty	NameSpaceVersion	String	1.00
HasProperty	NameSppacePublicationData	DateTime	2017-11-28
HasProperty	IsNamespaceSubset	Boolean	False
HasProperty	StaticNodeIdTypes	IdType[]	{Numeric}
HasProperty	StaticNumericNodeIdrange	NumericRange	Null
HasProperty	StaticStringNodeIdPattern	String	Null

9.2. OPC UA Conformance Unit and Profile

This section defines the profiles and conformance units of the OPC UA information model which are related to CSP+ for machine. *Profiles* are defined as a name of grouped conformance units.

The *Profile* which is to be a *Facet* is expected to be combined with other *Profiles* to define full functions of the OPC UA *Server* or *Client*.

Table 9-2 shows the *Facet* which can be used by a *Server* where companion specifications of the *Information Model* of CSP+ for machine are implemented.

Table 9-2 – CSP+ for machine Server Facet Definition

Conformance Unit	Description	Optional/ Mandatory
CSP+ for machine Information Model	Support *Objects* that conform to the types defined by this specification.	M
CSP+ for machine DeviceSet	Support the full component hierarchy with *CsppMachineType* below the DeviceSet *Object* defined in IEC 62541-100:2015	M
Profile		
BaseDevice_Server_Facet (defined in IEC 62541-100:2015)		M

Table 9-3 shows the *Facet* which can be used by a *Client* where companion specifications of the *Information Model* of CSP+ for machine are implemented.

Table 9-3 – CSP+ for machine Client Facet Definition

Conformance Unit	Description	Optional/ Mandatory
CSP+ for machine Information Model	Support *Objects* that conform to the types defined by this specification.	M
CSP+ for machine DeviceSet	Support the full component hierarchy with *CsppMachineType* below the DeviceSet *Object* defined in IEC 62541-100:2015	M
Profile		
BaseDevice_Client_Facet (defined in IEC 62541-100:2015)		M

9.3. Handling of the OPC UA Namespace

Namespaces are used to generate identifiers which are unique among various naming authorities. *NodeId*s and *BrowseName*s of *Attribute*s are identifiers. *Node*s in the UA *Address Space* are definitely identified by the *NodeId*s. In contrast to the *NodeId*s, *BrowseName*s cannot be used to definitely identify *Node*s. Different *Node*s may share a same *BrowseName*. *BrowseName*s are used to create a browse path between two *Node*s or define standard *Properties*.

A server may select the use of a same namespace for *NodeId*s and *BrowseName*s. However, for example, when a local server grants standard *Properties*, their *BrowseName*s must have a namespace of the standardizing body even though the namespace of the *NodeId*s reflect other things. All of *NodeId* for *Node*s which are not defined in this specifications must not use the standard namespace.

Table 9-4 lists mandatory/optional namespaces used by the server.

Table 9-4 – Namespace used in CSPPlusForMachine Server

NameSpace	Description	Optional/ Mandatory
http://opcfoundation.org/UA/	Namespace for *NodeIds* and *BrowseNames* defined in the OPC UA specification. This namespace shall have namespace index 0.	M
Local Server URI	Namespace for *Node*s defined in the local server. This may include types and instances used in a *Cspp Machine* represented by the server. This namespace shall have namespace index 1.	M
http://opcfoundation.org/UA/DI/	Namespace for NodeIds and BrowseNames defined in [IEC 62541-100:2015]. The namespace index is server specific.	M
http://opcfoundation.org/UA/CSPPlusForMachine/	Namespace for NodeIds and BrowseNames defined in this specification. The namespace index is server specific.	M
Vendor specific types and instances	A server may provide vendor specific types like types derived from CsppMachineType or vendor specific instances of devices in a vendor specific namespace.	O

Table 9-5 provides a list of namespaces and their index used for BrowseNames in this specification. The default namespace of this specification is not listed since all BrowseNames without prefix use this default namespace.

Table 9-5 – Namespaces used in this specification

Namespace	Namespace Index	Example
http://opcfoundation.org/UA/	0	0:BaseObjectType
http://opcfoundation.org/UA/DI/	1	1:ParameterSet

APPENDIX A: Namespace and Mapping

A.1 Namespace and Identifiers of the Information Model of CSP+ for Machine

This appendix defines numeric identifiers for all numeric type *NodeId* that are defined in this specifications. The identifiers are specified in the following syntax in CSV files.

< SymbolName >, < Identifier >, < NodeClass >

'SymbolName' described above refers to *BrowseName* of a *Type Node* or BrowsePath of an *Instance Node* when the identifier of the *NodeId* is a numerical value in the *Instance Node* in this specifications.

Browse paths of instance *Node*s are described by connecting the *BrowseName* of the instance or type which holds it with the *BrowseName* of the target *Instance Node*. The underscore ("_") is used as a delimiter for *BrowseName*s of paths.

The NamespaceUri for all *NodeId*s is defined in the following.

http://www.opcfoundation.org/UA/CSPPlusForMachine/

The CSV file for the specifications of this version is available from the following.

http://www.opcfoundation.org/UA/schemas/CSPPlusForMachine/1.0/NodeIds.csv

The latest *NodeId*s are available from the following.

http://www.opcfoundation.org/UA/schemas/CSPPlusForMachine/NodeIds.csv

The electronic version of the full information model defined in this specifications is also available. It conforms to the schema of the information model in the XML format which is defined in IEC 62541-6:2015. The information model schema for this version is available from the following.

http://www.opcfoundation.org/UA/schemas/CSPPlusForMachine/1.0/Opc.Ua.CSPPlusForMachine.Nodeset2.xml

The latest information model schema is available from the following.

http://www.opcfoundation.org/UA/schemas/CSPPlusForMachine/Opc.Ua.CSPPlusForMachine.Nodeset2.xml

A.2 Profile URI in the Information Model of CSP+ for Machine

Table A-1 defines the profile URI in the information model of CSP+ for machine.

Table A-1 – Profile URI

Profile	Profile URI
CSP+ for machine Server Facet	http://opcfoundation.org/UA-Profile/External/CSPPlusForMachine/CSP+formachineServer
CSP+ for machine Client Facet	http://opcfoundation.org/UA-Profile/External/CSPPlusForMachine/CSP+formachineClient

RELATED SPECIFICATIONS

None.

CC-Link
CC-Link IE

Prepared in June, 2017