

EUROMAP 82.2

**OPC UA interfaces for plastics and rubber
machinery – Peripheral devices –
Part 2: Hot runner devices**

Release Candidate 1.0, 01 November 2019

**EUROMAP 82.2 (Version 1.0) is identical with
OPC 40082-2 (Edition 1.0) and VDMA 40082-2:2019-11**

Contents

| | Page |
|--|-----------|
| Foreword..... | 7 |
| 1 Scope | 7 |
| 2 Normative references | 7 |
| 3 Terms, definitions and conventions | 8 |
| 3.1 Overview | 8 |
| 3.2 Conventions used in this document..... | 8 |
| 3.3 Abbreviations | 8 |
| 4 General information to OPC UA interfaces for plastics and rubber machinery and OPC UA | 8 |
| 5 Use cases | 8 |
| 6 HRD_InterfaceType..... | 9 |
| 6.1 HRD_InterfaceType Definition | 9 |
| 6.2 DisplayLanguage | 11 |
| 7 Identification..... | 11 |
| 8 MachineConfiguration..... | 11 |
| 9 OperationType..... | 11 |
| 9.1 DeviceMappingNumber..... | 11 |
| 9.2 IdentifyDevice..... | 12 |
| 9.3 HighestActiveAlarmSeverity..... | 12 |
| 9.4 ActiveErrors | 12 |
| 9.5 ResetAllErrors..... | 12 |
| 9.6 ResetErrorById..... | 12 |
| 9.7 EnablePower | 13 |
| 9.8 ActiveSetValues | 13 |
| 10 Zones | 13 |
| 11 Diagnostics..... | 13 |
| 12 MaintenanceInformation | 14 |
| 13 ZoneType | 14 |
| 13.1 Name | 14 |
| 13.2 Status | 14 |
| 13.3 ThermocoupleType and CommunicationProtocolType..... | 15 |
| 13.4 ActuatorType..... | 15 |
| 13.5 Temperature | 16 |
| 13.6 Controller..... | 16 |
| 13.7 HeatUp | 16 |
| 13.8 TemperatureRiseMonitoring..... | 16 |
| 14 HRDTemperatureType | 16 |

| | | |
|--------------|---|-----------|
| 15 | ControllerType | 16 |
| 15.1 | SetValueActive | 17 |
| 15.2 | ActualValueActive | 17 |
| 15.3 | SetValueType / ActualType | 17 |
| 15.4 | UpperOutput | 18 |
| 15.5 | LowerOutput | 18 |
| 15.6 | OutputTime | 18 |
| 15.7 | ReferenceZone | 18 |
| 15.8 | SetValueManualOutput | 18 |
| 15.9 | ActualOutput | 18 |
| 15.10 | AverageControllerOutput | 18 |
| 15.11 | ActualLoadCurrent | 19 |
| 15.12 | SmoothEnergyFlowActive | 19 |
| 15.13 | UpperSetValueCascade | 19 |
| 16 | HeatUpType | 19 |
| 16.1 | ManualOutputLimitActive | 19 |
| 16.2 | SetValueManualOutputLimit | 19 |
| 16.3 | SetValueTemperature | 19 |
| 16.4 | HeatUpEvenlyActive | 19 |
| 17 | TemperatureRiseMonitoringType | 19 |
| 17.1 | SetValueActive | 20 |
| 17.2 | ErrorDetected | 20 |
| 17.3 | SupervisionTime | 20 |
| 17.4 | SetValueTemperatureChange | 20 |
| 18 | Alarm management | 20 |
| 18.1 | General | 20 |
| 18.2 | ZoneOffNormalAlarmType | 20 |
| 19 | Profiles and Namespaces | 21 |
| 19.1 | Namespace Metadata | 21 |
| 19.2 | Conformance Units and Profiles | 21 |
| 19.3 | Handling of OPC UA Namespaces | 22 |
| | Annex A (normative) OPC 40082-2 Namespace and mappings | 23 |

Figures

| | |
|---|---|
| Figure 1 – HRD_InterfaceType Overview | 9 |
|---|---|

Tables

| | |
|--|----|
| Table 1 – HRD_InterfaceType Definition | 9 |
| Table 2 – OperationType Definition | 11 |
| Table 3 – HRDActiveErrorDataType Definition (subtype of ActiveErrorDataType) | 12 |
| Table 4 –ResetErrorByld Method Arguments | 12 |
| Table 5 – ResetErrorByld Method AddressSpace Definition | 13 |
| Table 6 – Values for ActiveSetValues | 13 |
| Table 7 – ZonesType Definition | 13 |
| Table 8 – MaintenanceInformationType Definition | 14 |
| Table 9 – ZoneType Definition | 14 |
| Table 10 - ZoneStatusEnumeration Definition | 15 |
| Table 11 – Values for ThermocoupleType | 15 |
| Table 12 – Values for CommunicationProtocolType | 15 |
| Table 13 – Values for ActuatorType | 16 |
| Table 14 – HRDTemperatureType Definition | 16 |
| Table 15 – ControllerType Definition | 17 |
| Table 16 – ControllerTypeEnumeration Definition | 18 |
| Table 17 - HeatUpType | 19 |
| Table 18 - TemperatureRiseMonitoringType Definition | 20 |
| Table 19 - ZoneOffNormalAlarmType Definition | 21 |
| Table 20 – NamespaceMetadata Object for this Specification | 21 |
| Table 21 – OPC 40082-2 <i>Basic Server Facet</i> Definition | 21 |
| Table 22 – OPC 40082-2 <i>Alarms Server Facet</i> Definition | 22 |
| Table 23 – OPC 40082-2 <i>Diagnostics Server Facet</i> Definition | 22 |
| Table 24 – OPC 40082-2 <i>Maintenance Server Facet</i> Definition | 22 |
| Table 25 – Namespaces used in an OPC 40082-2 Server | 22 |
| Table 26 – Namespaces used in this specification | 22 |

OPC Foundation / EUROMAP

AGREEMENT OF USE

COPYRIGHT RESTRICTIONS

- This document is provided "as is" by the OPC Foundation and EUROMAP.
- Right of use for this specification is restricted to this specification and does not grant rights of use for referred documents.
- Right of use for this specification will be granted without cost.
- This document may be distributed through computer systems, printed or copied as long as the content remains unchanged and the document is not modified.
- OPC Foundation and EUROMAP do not guarantee usability for any purpose and shall not be made liable for any case using the content of this document.
- The user of the document agrees to indemnify OPC Foundation and EUROMAP and their officers, directors and agents harmless from all demands, claims, actions, losses, damages (including damages from personal injuries), costs and expenses (including attorneys' fees) which are in any way related to activities associated with its use of content from this specification.
- The document shall not be used in conjunction with company advertising, shall not be sold or licensed to any party.
- The intellectual property and copyright is solely owned by the OPC Foundation and EUROMAP.

PATENTS

The attention of adopters is directed to the possibility that compliance with or adoption of OPC or EUROMAP specifications may require use of an invention covered by patent rights. OPC Foundation or EUROMAP shall not be responsible for identifying patents for which a license may be required by any OPC or EUROMAP specification, or for conducting legal inquiries into the legal validity or scope of those patents that are brought to its attention. OPC or EUROMAP specifications are prospective and advisory only. Prospective users are responsible for protecting themselves against liability for infringement of patents.

WARRANTY AND LIABILITY DISCLAIMERS

WHILE THIS PUBLICATION IS BELIEVED TO BE ACCURATE, IT IS PROVIDED "AS IS" AND MAY CONTAIN ERRORS OR MISPRINTS. THE OPC FOUNDATION NOR EUROMAP MAKES NO WARRANTY OF ANY KIND, EXPRESSED OR IMPLIED, WITH REGARD TO THIS PUBLICATION, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OF TITLE OR OWNERSHIP, IMPLIED WARRANTY OF MERCHANTABILITY OR WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE OR USE. IN NO EVENT SHALL THE OPC FOUNDATION NOR EUROMAP BE LIABLE FOR ERRORS CONTAINED HEREIN OR FOR DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL, RELIANCE OR COVER DAMAGES, INCLUDING LOSS OF PROFITS, REVENUE, DATA OR USE, INCURRED BY ANY USER OR ANY THIRD PARTY IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS MATERIAL, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

The entire risk as to the quality and performance of software developed using this specification is borne by you.

RESTRICTED RIGHTS LEGEND

This Specification is provided with Restricted Rights. Use, duplication or disclosure by the U.S. government is subject to restrictions as set forth in (a) this Agreement pursuant to DFARs 227.7202-3(a); (b) subparagraph (c)(1)(i) of the Rights in Technical Data and Computer Software clause at DFARs 252.227-7013; or (c) the Commercial Computer Software Restricted Rights clause at FAR 52.227-19 subdivision (c)(1) and (2), as applicable. Contractor / manufacturer are the OPC Foundation, 16101 N. 82nd Street, Suite 3B, Scottsdale, AZ, 85260-1830

COMPLIANCE

The combination of EUROMAP and OPC Foundation shall at all times be the sole entities that may authorize developers, suppliers and sellers of hardware and software to use certification marks, trademarks or other special designations to indicate compliance with these materials as specified within this document. Products developed using this specification may claim compliance or conformance with this specification if and only if the software satisfactorily meets the certification requirements set by EUROMAP or the OPC Foundation. Products

that do not meet these requirements may claim only that the product was based on this specification and must not claim compliance or conformance with this specification.

Trademarks

Most computer and software brand names have trademarks or registered trademarks. The individual trademarks have not been listed here.

Foreword

This specification was created by a joint working group of the OPC Foundation and EUROMAP. It is adopted identically as VDMA Specification.

NOTE: The highlighted links in Clause 2 and Annex A are not active yet and may be changed in the final version.

EUROMAP

EUROMAP is the European umbrella association of the plastics and rubber machinery industry which accounts for annual sales of around 13.5 billion euro and a 40 per cent share of worldwide production. Almost 75 per cent of its European output is shipped to worldwide destinations. With global exports of 10.0 billion euro, EUROMAP's around 1,000 machinery manufacturers are market leaders with nearly half of all machines sold being supplied by EUROMAP members.

EUROMAP provides technical recommendations for plastics and rubber machines. In addition to standards for machine descriptions, dimensions and energy measurement, interfaces between machines feature prominently. The provision of manufacturer independent interfaces ensures high levels of machine compatibility.

OPC Foundation

OPC is the interoperability standard for the secure and reliable exchange of data and information in the industrial automation space and in other industries. It is platform independent and ensures the seamless flow of information among devices from multiple vendors. The OPC Foundation is responsible for the development and maintenance of this standard.

OPC UA is a platform independent service-oriented architecture that integrates all the functionality of the individual OPC Classic specifications into one extensible framework. This multi-layered approach accomplishes the original design specification goals of:

- Platform independence: from an embedded microcontroller to cloud-based infrastructure
- Secure: encryption, authentication, authorization and auditing
- Extensible: ability to add new features including transports without affecting existing applications
- Comprehensive information modelling capabilities: for defining any model from simple to complex

1 Scope

OPC 40082-2 describes the interface for hot runner devices (HRD) for data exchange via OPC UA. The target of OPC 40082-2 is to provide a standard interface for hot runner devices from different manufacturers to ensure compatibility. The following functionalities are covered:

- General information about the hot runner device
- Status information
- Process data

Safety related signals like emergency stop are not included.

2 Normative references

The following referenced documents are indispensable for the application of this specification. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments and errata) applies.

OPC 10000-1: OPC Unified Architecture – Part 1: Overview and Concepts (version 1.04)

- <http://www.opcfoundation.org/UA/Part1/>

OPC 10000-2: OPC Unified Architecture – Part 2: Security Model (version 1.04)

- <http://www.opcfoundation.org/UA/Part2/>

OPC 10000-3: OPC Unified Architecture – Part 3: Address Space Model (version 1.04)

- <http://www.opcfoundation.org/UA/Part3/>

OPC 10000-4: OPC Unified Architecture – Part 4: Services (version 1.04)

- <http://www.opcfoundation.org/UA/Part4/>

OPC 10000-5: OPC Unified Architecture – Part 5: Information Model (version 1.04)

- <http://www.opcfoundation.org/UA/Part5/>

OPC 10000-7: OPC Unified Architecture – Part 7: Profiles (version 1.04)

- <http://www.opcfoundation.org/UA/Part7/>

OPC 10000-8: OPC Unified Architecture – Part 8: Data Access (version 1.04)

- <http://www.opcfoundation.org/UA/Part8/>

OPC 10000-9: OPC Unified Architecture – Part 9: Alarms and Conditions (version 1.04)

- <http://www.opcfoundation.org/UA/Part9/>

OPC 10000-11: OPC Unified Architecture – Part 11: Historical Access (version 1.04)

- <http://www.opcfoundation.org/UA/Part11/>

OPC 10000-100: OPC Unified Architecture – Part 100: Device Information Model (version 1.02)

- <http://www.opcfoundation.org/UA/Part100/>

OPC 40083: OPC UA interfaces for plastics and rubber machinery – General Type definitions (version 1.02)

- <http://www.opcfoundation.org/UA/PlasticsRubber/GeneralTypes/>

3 Terms, definitions and conventions

3.1 Overview

It is assumed that basic concepts of OPC UA information modelling are understood in this specification. This specification will use these concepts to describe the OPC 40082-2 Information Model. For the purposes of this document, the terms and definitions given in the documents referenced in Clause 2 apply.

NOTE: OPC UA terms and terms defined in this specification are *italicized* in the specification.

3.2 Conventions used in this document

The conventions described in OPC 40083 apply.

3.3 Abbreviations

HRD hot runner device

4 General information to OPC UA interfaces for plastics and rubber machinery and OPC UA

For general information on OPC UA interfaces for plastics and rubber machinery and OPC UA see OPC 40083.

5 Use cases

OPC 40082-2 covers the following functionalities:

- General information about the hot runner device
- Status information
- Process data

6 HRD_InterfaceType

6.1 HRD_InterfaceType Definition

This OPC UA *ObjectType* is used for the root *Object* representing a hot runner device with its subcomponents. It is formally defined in Table 1.

NOTE: To promote interoperability of *Clients* and *Servers*, all instantiated *Devices* shall be aggregated in an *Object* called "DeviceSet" (see OPC UA for Devices)

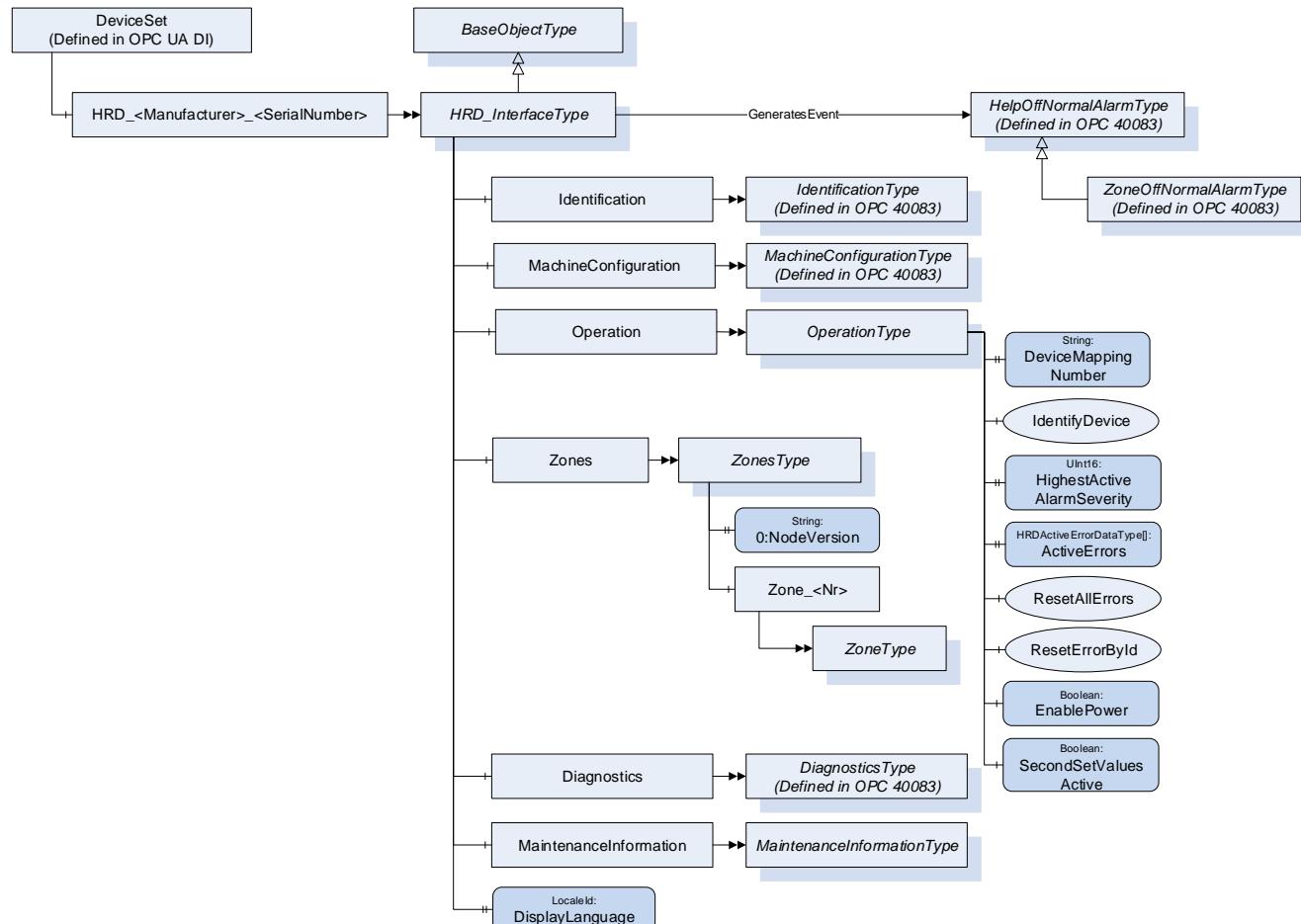


Figure 1 – HRD_InterfaceType Overview

Table 1 – HRD_InterfaceType Definition

| Attribute | Value | | | | |
|----------------------------------|-------------------|------------------------|----------------------|----------------------------|----------------|
| BrowseName | HRD_InterfaceType | | | | |
| IsAbstract | False | | | | |
| References | Node Class | BrowseName | DataType | TypeDefinition | Modelling Rule |
| <i>Subtype of BaseObjectType</i> | | | | | |
| HasComponent | Object | Identification | | IdentificationType | M |
| HasComponent | Object | MachineConfiguration | | MachineConfigurationType | M |
| HasComponent | Object | Operation | | OperationType | M |
| HasComponent | Object | Zones | | ZonesType | M |
| HasComponent | Object | Diagnostics | | DiagnosticsType | O |
| HasComponent | Object | MaintenanceInformation | | MaintenanceInformationType | O |
| HasProperty | Variable | DisplayLanguage | LocaleId | .PropertyType | O, RW |
| GeneratesEvent | ObjectType | HelpOffNormalAlarmType | Defined in OPC 40083 | | |

The *BrowseName* of the object instance shall be "HRD_<Manufacturer>_<SerialNumber>"

Example: "HRD_Gammaflux_0123456".

NOTE: The namespace of this *BrowseName* is the local server URI with namespace index 1 or a vendor specific namespace with server specific namespace index (see Table 25). The *BrowseNames* of the nodes below are in the namespace of the specification where used Type is defined.

Examples:

| BrowseName | Namespace | Namespace index | Remarks |
|---------------------|--|------------------------|---|
| HRD_Inglass_0123456 | Local Server URI or vendor specific namespace | 1 or server specific | OPC 40082-2 only defines the <i>HRD_InterfaceType</i> . The instance is generated in the local server |
| ↓ | | | |
| Identification | http://opcfoundation.org/UA/PlasticsRubber/HotRunner/ | server specific | The object <i>Identification</i> is a child of <i>HRD_InterfaceType</i> which is defined in OPC 40082-2 |
| ↓ | | | |
| Manufacturer | http://opcfoundation.org/UA/PlasticsRubber/GeneralTypes/ | server specific | The variable <i>Manufacturer</i> is a child of <i>IdentificationType</i> which is defined in OPC 40083. |

| BrowseName | Namespace | Namespace index | Remarks |
|---------------------|--|------------------------|--|
| HRD_Inglass_0123456 | Local Server URI or vendor specific namespace | 1 or server specific | OPC 40082-2 only defines the <i>HRD_InterfaceType</i> . The instance is generated in the local server |
| ↓ | | | |
| Zones | http://opcfoundation.org/UA/PlasticsRubber/HotRunner/ | server specific | The object <i>Zones</i> is a child of <i>HRD_InterfaceType</i> which is defined in OPC 40082-2 |
| ↓ | | | |
| Zone_1 | Local Server URI or vendor specific namespace | 1 or server specific | The objects for the zones are modelled as <i>OptionalPlaceholder</i> . The instances are server specific |
| ↓ | | | |
| Temperature | http://opcfoundation.org/UA/PlasticsRubber/HotRunner/ | server specific | The object <i>Temperature</i> is a child of <i>ZoneType</i> which is defined in OPC 40082-2 |
| ↓ | | | |
| ActualValue | http://opcfoundation.org/UA/PlasticsRubber/GeneralTypes/ | server specific | The variable <i>ActualValue</i> is a child of <i>Temperature</i> which has the <i>HRDControlledParameterType</i> as type definition. This is derived from the <i>ControlledParameterType</i> which is defined in OPC 40083 |

| BrowseName | Namespace | Namespace index | Remarks |
|---------------------|---|----------------------|---|
| HRD_Inglass_0123456 | Local Server URI or vendor specific namespace | 1 or server specific | OPC 40082-2 only defines the <i>HRD_InterfaceType</i> . The instance is generated in the local server |
| ↓ | | | |
| Zones | http://opcfoundation.org/UA/PlasticsRubber/HotRunner/ | server specific | The object <i>Zones</i> is a child of <i>HRD_InterfaceType</i> which is defined in OPC 40082-2 |
| ↓ | | | |
| NodeVersion | http://opcfoundation.org/UA/ | 0 | The Property <i>NodeVersion</i> is defined in OPC UA |

6.2 DisplayLanguage

With the *DisplayLanguage* *Property* the client can set the desired language on the user interface at the HRD. If the peripheral device does not support the configured language, it can keep the previous setting or use English as the default.

7 Identification

The *IdentificationType* for the identification of the device is defined in OPC 40083. All mandatory nodes shall be filled with valid values from the server.

The *DeviceClass* *Property* in the *Identification Object* shall have the value "Hot Runner Device".

8 MachineConfiguration

The *MachineConfiguration Object* represents the current configuration of the hot runner device. The *MachineConfigurationType* is defined in OPC 40083.

9 OperationType

This *ObjectType* contains components which are necessary to operate the HRD. It is formally defined in Table 2.

Table 2 – OperationType Definition

| Attribute | Value | | | | |
|----------------------------------|---------------|-----------------------------|--------------------------|------------------------------|----------------|
| BrowseName | OperationType | | | | |
| IsAbstract | False | | | | |
| References | Node Class | BrowseName | DataType | TypeDefinition | Modelling Rule |
| Subtype of <i>BaseObjectType</i> | | | | | |
| HasProperty | Variable | DeviceMappingNumber | UInt32 | .PropertyType | M, RW |
| HasComponent | Method | IdentifyDevice | | | O |
| HasProperty | Variable | HighestActiveAlarm Severity | UInt16 | .PropertyType | M, R |
| HasComponent | Variable | ActiveErrors | HRDActiveErrorDataType[] | BaseDataVariableType | M, R |
| HasComponent | Method | ResetAllErrors | | | O |
| HasComponent | Method | ResetErrorByld | | | O |
| HasProperty | Variable | EnablePower | Boolean | .PropertyType | O, RW |
| HasComponent | Variable | ActiveSetValues | UInt16 | MultiStateValueDiscrete Type | O, RW |

9.1 DeviceMappingNumber

Description: Unique identifier/address/number for devices of the same *DeviceType* within a local network. Several peripheral devices of the same *DeviceType* can be connected to an client (e.g. injection moulding machine). In most applications, the client must map the connected peripheral devices to internal logical devices and zones in a fixed configuration (e.g. hot runner systems according

to the wiring or temperature control devices according to the tubing).
The mapping shall be stable after reconnecting the devices and is therefore not possible via IP addresses, which can be assigned dynamically via DHCP. *DeviceMappingNumber* sets the mapping order of peripheral devices of the same type on the local network and is therefore of type *UInt32*.

Example: 1

9.2 IdentifyDevice

Description: The peripheral device on which this method is called shows itself by e.g. activation of a LED.

Signature:

```
IdentifyDevice();
```

9.3 HighestActiveAlarmSeverity

Description: Indication of the severity of the highest active alarm (0 = no active alarm – 1000 = possible error). It provides a minimal error handling for devices without alarm support. However, the variable shall be filled even if alarms are supported.

Example: 400

9.4 ActiveErrors

Description: List of the active errors of the device. It provides a minimal error handling for devices without alarm support. However, the variable shall be filled even if alarms are supported. The *HRDActiveErrorDataType* is defined in OPC 40083. If there is no active error, the array is empty.

The *HRDActiveErrorDataType* is a subtype of the *ActiveErrorDataType* defined in OPC 40083 and adds the element Zone which represents the Nodeld of the affected zone.

Table 3 – HRDActiveErrorDataType Definition (subtype of ActiveErrorDataType)

| Name | Type |
|------------------------|-----------|
| HRDActiveErrorDataType | structure |
| Zone | Nodeld |

9.5 ResetAllErrors

Description: Method to reset all errors of the device.

Signature:

```
ResetAllErrors();
```

9.6 ResetErrorById

Description: Method to reset one error of the device.

Signature:

```
ResetErrorById(  
    [in] String Id);
```

Table 4 –ResetErrorById Method Arguments

| Argument | Description |
|----------|---|
| Id | Id of the error, listed in <i>ActiveErrors</i> , that shall be reset. |

Table 5 – ResetErrorByld Method AddressSpace Definition

| Attribute | Value | | | | |
|-------------|----------------|----------------|------------|----------------|----------------|
| BrowseName | ResetErrorByld | | | | |
| References | Node Class | BrowseName | DataType | TypeDefinition | Modelling Rule |
| HasProperty | Variable | InputArguments | Argument[] | .PropertyType | Mandatory |

9.7 EnablePower

Description: The optional property is a global power control switch for all zone controllers.

“*EnablePower=false*” turns off the entire device.

“*EnablePower=true*” turns on the device according to the zone-specific settings.

9.8 ActiveSetValues

With the *ActiveSetValues Variable* the used set temperature for the temperature zones is selected (see 14 HRDTemperatureType).

The *TypeDefinition* for the *Variable* is *MultiStateValueDiscreteType*, so the *Properties EnumValues* and *ValueAsText* must be filled with the supported values out of Table 6.

Table 6 – Values for ActiveSetValues

| EnumValue | ValueAsText | Description |
|-----------|-------------|--|
| 0 | First | Use of value stored as SetValue |
| 1 | Second | Use of value stored as SecondSetValue |
| 2 | Standby | Use of value stored as SetStandbyValue |
| 3 | Boost | Use of value stored as BoostSetValue |

The supported values are related on the provided set temperatures in the HRDTemperatureType. If e.g. value 3 “Boost” is available, all temperature zones shall provide the variable *BoostSetValue*. If only some zones have a boost, for the others, the value of *BoostSetValue* is the same as *SetValue*.

10 Zones

Zones is a container for zones, in analogy with the container concept in OPC 40083.

The *ObjectType* is formally defined in Table 7.

Table 7 – ZonesType Definition

| Attribute | Value | | | | |
|----------------------------------|------------|-----------------------------|--------|---------------|------|
| BrowseName | ZonesType | | | | |
| IsAbstract | False | | | | |
| <i>Subtype of BaseObjectType</i> | | | | | |
| HasProperty | Variable | 0:NodeVersion | String | .PropertyType | M, R |
| HasComponent | Object | Zone_<Nr> | | ZoneType | MP |
| GeneratesEvent | ObjectType | GeneralModelChangeEventType | | | |

When instances for device zones are created, the *BrowseNames* shall be “Zone_<Nr>” (starting with 1). Examples: “Zone_1”, “Zone_11”

11 Diagnostics

Diagnostics is an optional component.

Some manufacturers offer diagnosis functions to check, for example, the wiring to the heating system or the sensor and heater allocation. Furthermore, it can be checked whether the cartridge heaters are working correctly. *DiagnosticsType* is defined in OPC 40083.

For zone-related results published by instances of *DiagnosisStepEndEventType*, *InputNode* shall contain the *NodeId* of the corresponding zone instance.

12 MaintenanceInformation

MaintenanceInformation is an optional component that provides information about the maintenance status of various parts of a hot runner device.

Table 8 – MaintenanceInformationType Definition

| Attribute | Value | | | | |
|---------------------------|----------------------------|------------|----------|-----------------|-----------------|
| BrowseName | MaintenanceInformationType | | | | |
| IsAbstract | False | | | | |
| References | Node Class | BrowseName | DataType | TypeDefinition | Modelling Rules |
| Subtype of BaseObjectType | | | | | |
| HasComponent | Object | Heating | | MaintenanceType | O |
| HasComponent | Object | SafetyTest | | MaintenanceType | O |
| HasComponent | Object | CoolingFan | | MaintenanceType | O |

The *MaintenanceType* is defined in OPC 40083.

13 ZoneType

ZoneType represent all functionalities of a heating zone, such as temperature monitoring, control, heatup and is formally defined in Table 9.

Table 9 – ZoneType Definition

| Attribute | Value | | | | |
|---------------------------|------------|---------------------------|-----------------------|-------------------------------|-----------------|
| BrowseName | ZoneType | | | | |
| IsAbstract | False | | | | |
| References | Node Class | BrowseName | DataType | TypeDefinition | Modelling Rules |
| Subtype of BaseObjectType | | | | | |
| HasProperty | Variable | Name | String | .PropertyType | O, RW |
| HasProperty | Variable | Status | ZoneStatusEnumeration | .PropertyType | M, R |
| HasProperty | Variable | ThermocoupleType | UInt16 | MultiStateValueDiscreteType | O, RW |
| HasComponent | Variable | CommunicationProtocolType | UInt16 | MultiStateValueDiscreteType | O, RW |
| HasProperty | Variable | ActuatorType | UInt16 | MultiStateValueDiscreteType | O, RW |
| HasComponent | Object | Temperature | | HRDTemperatureType | M |
| HasComponent | Object | Controller | | ControllerType | M |
| HasComponent | Object | HeatUp | | HeatUpType | O |
| HasComponent | Object | TemperatureRiseMonitoring | | TemperatureRiseMonitoringType | O |

13.1 Name

A user given name of the zone.

13.2 Status

ZoneStatus provides a zone-related status for hot runner devices without event support. Hot runner devices may provide detailed information about zone-related alarms and conditions via event notifications of *ZoneOffNormalAlarmType*.

Table 10 – ZoneStatusEnumeration Definition

| Value | Description |
|-------------------------------|--|
| OTHER_0 | Zone status is unknown. |
| GOOD_1 | Zone is ready. However, low- and medium-severity alarms of type ZoneOffNormalAlarmType can be active. |
| SENSOR_FAULT_2 | Zone has an unknown sensor fault |
| TEMPERATURE_SENSOR_BROKEN_3 | Temperature sensor is broken. Operation in manual mode is possible |
| TEMPERATURE_SENSOR_REVERSED_4 | Temperature sensor is reversed. |
| POWER_UNIT_FAILED_5 | Power unit failed |
| HEATING_OUTPUT_TO_LOW_6 | Heating output is to low |
| ERROR_7 | The zone has an error and is not ready for production. Detailed information may have been published via ZoneOffNormalAlarmType. |
| WARNING_8 | The zone has a warning, but no need to stop the production. Detailed information may be published via an alarm (ZoneOffNormalAlarmType). |
| LEAKAGE_DETECTED_9 | Leakage has been detected. |

13.3 ThermocoupleType and CommunicationProtocolType

This two *Variables* are used to specify the type of connected external temperature sensor and the used communication protocol between the sensor and the control system of the HRD.

The *TypeDefinition* for both *Variables* is *MultiStateValueDiscreteType*, so the *Properties EnumValues* and *ValueAsText* must be filled with the supported values out of Table 11and Table 12.

Table 11 – Values for ThermocoupleType

| EnumValue | ValueAsText | Description |
|-----------|-------------|----------------------------|
| 0 | OTHER | Other sensor type |
| 1 | E | Type E sensor: NiCr-CuNi |
| 2 | J | Type J, L sensor: Fe-CuNi |
| 3 | K | Type K sensor: NiCr-Ni |
| 4 | N | Type N sensor: NiCrSi-NiSi |
| 5 | T | Type T sensor: Cu-CuNi |
| 6 | PT100 | Pt 100-Sensor |

Table 12 – Values for CommunicationProtocolType

| EnumValue | ValueAsText | Description |
|-----------|-------------|--|
| 0 | OTHER | Other connection type |
| 1 | LOCAL | Communication integrated in the local control system (local input) |
| 2 | PROFIBUS | Values via Profibus |
| 3 | OPC_UA | Values via OPC UA |
| 4 | I2C | Values via I2C |
| 5 | CAN | Values via CAN |

Which sensor types and protocols and combinations are supported is device dependent. Especially when the *CommunicationProtocolType* has the value 1 (LOCAL), the *ThermocoupleType* could be set to a fixed value by the HRD.

13.4 ActuatorType

Type of the actuator. The *TypeDefinition* is *MultiStateValueDiscreteType*, so the *Properties EnumValues* and *ValueAsText* must be filled with the supported values out of Table 13.

With hotrunner systems that are pre-installed, this value is set to *ReadOnly*.

Table 13 – Values for ActuatorType

| EnumValue | ValueAsText | Description |
|-----------|---------------------------------------|---|
| 0 | SSR_WITHOUT_ERROR_DETECTION | Solid state relays without error monitoring |
| 1 | SSR_WITH_ERROR_DETECTION | Solid state relays with error monitoring |
| 2 | PHASE_CONTROLLED_MODULATOR | Solid state relays with phase control |
| 3 | DISCRETE_CONTROL_WITH_ERROR_DETECTION | Discrete Device Control – with error monitoring |

13.5 Temperature

Setting and monitoring of the temperature. The *HRDControlledParameterType* is defined in clause 14.

Unit: °C or F

13.6 Controller

Setting and monitoring of the controller. See 15 *ControllerType*.

13.7 HeatUp

Setting for heat up procedure. *HeatUp* is an alternative to the *SetRamp* functionality of *MonitoredParameterType*. See 16 *HeatUpType*.

13.8 TemperatureRiseMonitoring

TemperatureRiseMonitoring is an additional monitoring for the process temperature.

See *TemperatureRiseMonitoringType*.

14 HRDTemperatureType

The *ControlledParameterType* as defined in OPC 40083 is used for process parameters that are controlled by the client by writing a set value and optional ramps and parameters for closed loop control. For HRD the *HRDControlledParameterType* is derived from the *ControlledParameterType*. There, the Variables *SecondSetValue*, *BoostSetValue* and *StandbySetValue* are added.

The *SecondSetValue* can be used for various applications (e.g. variotherm process, etc.) in conjunction with Varan or “OPC-UA PubSub” in order to achieve fast switch-over. With the *StandbySetValue* and *BoostSetValue* additional values for setting the hot runner to standby or boost mode can be provided.

The provided set temperatures are related on the supported values in *ActiveSetValues Variable* in the *OperationType* (see 9.8).

Table 14 – HRDTemperatureType Definition

| Attribute | Value | | | | |
|--|--------------------|-----------------|----------|----------------|----------------|
| BrowseName | HRDTemperatureType | | | | |
| IsAbstract | False | | | | |
| References | Node Class | BrowseName | DataType | TypeDefinition | Modelling Rule |
| Subtype of <i>ControlledParameterType</i> (defined in OPC 40083) | | | | | |
| HasComponent | Variable | SecondSetValue | Double | AnalogItemType | O, RW |
| HasComponent | Variable | BoostSetValue | Double | AnalogItemType | O, RW |
| HasComponent | Variable | StandbySetValue | Double | AnalogItemType | O, RW |

15 ControllerType

Configuration and operation of the zone controller.

Table 15 – ControllerType Definition

| Attribute | Value | | | | |
|---------------------------|----------------|-------------------------|---------------------------|---------------------------------|-----------------|
| BrowseName | ControllerType | | | | |
| IsAbstract | False | | | | |
| References | Node Class | BrowseName | DataType | TypeDefinition | Modelling Rules |
| Subtype of BaseObjectType | | | | | |
| HasProperty | Variable | SetValueActive | Boolean | .PropertyType | M, RW |
| HasProperty | Variable | ActualValueActive | Boolean | .PropertyType | M, R |
| HasComponent | Variable | SetValueType | UInt16 | MultiStateValue DiscreteType | M, RW |
| HasProperty | Variable | ActualType | ControllerTypeEnumeration | .PropertyType | M, R |
| HasProperty | Variable | UpperOutput | Double | AnalogItemType | O, RW |
| HasProperty | Variable | LowerOutput | Double | AnalogItemType | O, RW |
| HasProperty | Variable | OutputTime | Duration | .PropertyType | O, RW |
| HasProperty | Variable | ReferenceZone | UInt32 | .PropertyType | O, RW |
| HasProperty | Variable | SetValueManualOutput | Double | AnalogItemType | O, RW |
| HasProperty | Variable | ActualOutput | Double | AnalogItemType | O, R |
| HasProperty | Variable | AverageControllerOutput | Double | AnalogItemType | O, R |
| HasProperty | Variable | ActualLoadCurrent | Double | AnalogItemType | O, R |
| HasProperty | Variable | SmoothEnergyFlowActive | Boolean | PropertyParams | O, RW |
| HasProperty | Variable | UpperSetValueCascade | Double | AnalogItemType | O, RW |

15.1 SetValueActive

A control zone is switched *on* and *off* with this parameter. *ActualValueActive* shows the current status of the controller. When it is switched *on*, the power control is realised with the predefined *ControlType*. When it is switched *off*, there is no power control. Only the actual temperature value is measured cyclically.

Usually there are various controller types available.

15.2 ActualValueActive

Indicates the current status of the controller. *ActualValueActive* is *true*, if the controller is switched *on* via *SetValueActive*.

15.3 SetValueType / ActualType

Set value and actual value for the controller type used by the zone.

The Enumeration for the possible Types is defined in Table 16.

Table 16 – ControllerTypeEnumeration Definition

| Value | Description |
|-----------------------|---|
| CLOSED_LOOP_CONTROL_0 | Closed loop control is active for the Temperature object of the zone. This is the default control behaviour in which the hot runner system is usually operated. |
| MANUAL_1 | Open loop control is active. SetValueManualOutput defines the constant required output. The zone is not controlled to the actual temperature value. |
| SYNCHRONOUS_ZONE_2 | Additional zones can be added to the standard type. This way, various heating elements can be operated with one sensor. The required parameter ReferenceZone (zone of the standard type) can be assigned multiple times. |
| CASCADE_3 | Two control circuits (zones) are interlinked. There is one nominal temperature, two actual temperature values and one actuator. The first control circuit is the cascade type and the second a standard type. The cascade type has no actuator. The cascade type requires a ReferenceZone with SetValueType=Standard to transfer its controller output internally to the second zone. The standard type uses the parameter UpperSetValueCascadeControllerSlave as the upper limit to prevent overheating. |
| COOL_ZONE_4 | This zone is assigned to a standard type. The two zones control to the same nominal and actual temperature, but have two separate outputs. The standard type operates the heating circuit and the cool zone the cooling circuit. The cool zone and the standard type are linked via the ReferenceZone parameter. This regulator is specially designed for cooling performance and has the respective control parameters (PiD: Xpk, Tn,Tv). Required parameters: ReferenceZone of the standard type and ControllerOutputTime if required to adapt the actuation cycle to the respective actuator. |

For SetValueType the TypeDefinition is MultiStateValueDiscreteType, so the Properties EnumValues and ValueAsText must be filled with the supported values.

15.4 UpperOutput

Limitation of the maximum output in closed-loop control in %.

15.5 LowerOutput

Limitation of the minimum output in closed-loop control in %.

15.6 OutputTime

Time basis for operating the actuator. The basic cycle of the actuator. It can also be called actuation cycle time and it is used with contactor-controlled power units in order to reduce the switching rate.

15.7 ReferenceZone

If zones are to operate parallel to a control zone, the reference relation can be realised with this parameter. The same applies to the cooling channel and cascade controller.

Valid range: 1 – “number of zones”

Not used: 0

15.8 SetValueManualOutput

If a zone is to be operated in the control mode, the respective output is predefined. Automatic switch-over to control mode is not permitted. If a sensor breaks, the production is stopped to make the operator aware of the situation. The device can recommend an optimum output percentage.

15.9 ActualOutput

Indicates the currently active output in percent.

15.10 AverageControllerOutput

Indicates the average output which can be used when a sensor is broken. The determination is active when the partial control circuit has reached the nominal value.

15.11 ActualLoadCurrent

Actual value of the load current in ampere.

15.12 SmoothEnergyFlowActive

The property is writeable by the client to switch on/off the smooth energy flow. With quick-reacting heating elements it is useful to adapt the output rate to the controller output time.

15.13 UpperSetValueCascade

If the two controllers are cascaded, the master defines to which nominal temperature value the slave is to control to. With this parameter, the upper limit for the slave is defined.

16 HeatUpType

HeatUpType is optional and an alternative to the *SetRamp* functionality of *MonitoredParameterType* defined in OPC 40083. With *HeatUpType*, it can be pre-defined how the control circuit is to be operated when it is switched on for the next time; with or without heat-up process.

There are various terms for heat-up, amongst others softstart. The target is to limit the controller output in order to heat up the heating element slowly, so that the moisture can evaporate without causing damage to the heating element.

Table 17 - HeatUpType

| Attribute | Value | | | | |
|---------------------------|------------|---------------------------|----------|----------------|-----------------|
| BrowseName | HeatUpType | | | | |
| IsAbstract | False | | | | |
| References | Node Class | BrowseName | DataType | TypeDefinition | Modelling Rules |
| Subtype of BaseObjectType | | | | | |
| HasProperty | Variable | ManualOutputLimitActive | Boolean | .PropertyType | O, RW |
| HasProperty | Variable | SetValueManualOutputLimit | Double | AnalogItemType | O, RW |
| HasProperty | Variable | SetValueTemperature | Double | AnalogItemType | O, RW |
| HasProperty | Variable | HeatUpEvenlyActive | Boolean | .PropertyType | O, RW |

16.1 ManualOutputLimitActive

Activates heat-up process with pre-defined *SetValueManualOutputLimit* until *SetValueTemperature* is reached.

16.2 SetValueManualOutputLimit

This pre-defined maximum output is valid until the *SetValueTemperature* is reached.

16.3 SetValueTemperature

If *ManualOutputLimitActive* is set, *SetValueManualOutputLimit* is active until this nominal temperature value is reached.

Unit: °C or F

16.4 HeatUpEvenlyActive

Activates evenly heat-up process until nominal *SetValue* of *Temperature* of *ZoneType* is reached.

17 TemperatureRiseMonitoringType

At maximum controller output, the temperature value must change in a given time by a specified value, otherwise there is an error in the measuring circuit.

Table 18 - TemperatureRiseMonitoringType Definition

| Attribute | Value | | | | |
|---------------------------|-------------------------------|---------------------------|----------|----------------|-----------------|
| BrowseName | TemperatureRiseMonitoringType | | | | |
| IsAbstract | False | | | | |
| References | Node Class | BrowseName | DataType | TypeDefinition | Modelling Rules |
| Subtype of BaseObjectType | | | | | |
| HasProperty | Variable | SetValueActive | Boolean | .PropertyType | M, RW |
| HasProperty | Variable | ErrorDetected | Boolao | .PropertyType | M, R |
| HasProperty | Variable | SupervisionTime | Duration | .PropertyType | O, RW |
| HasProperty | Variable | SetValueTemperatureChange | Double | AnalogItemType | O, RW |

17.1 SetValueActive

On / Off for the TemperatureRiseMonitoring.

17.2 ErrorDetected

Result of the TemperatureRiseMonitoring.

17.3 SupervisionTime

Specification of the time within the temperature must have changed.

17.4 SetValueTemperatureChange

Specification of the set value change

Unit: °C or F

18 Alarm management

18.1 General

As defined in OPC 40083, the root node of the specific interface, e.g. an instance of *HRD_InterfaceType*, set the *SubscribeToEvents* flag in the *EventNotifier* attribute.

The client subscribes to events at this root node and receives the events already defined in this specification, such as temperature limit alarms or diagnostic events.

A hot runner may optionally generate additional manufacturer-specific alarms, warnings or information displayed on the user interface of the device and can publish these events via two special *AlarmTypes*.

Zone-related messages should be represented by instances of *ZoneOffNormalAlarmType*, other device information is of type *HelpOffNormalAlarmType*.

Both are subtypes of *OffNormalAlarmType*, can be synchronized via *ConditionRefresh* and contain a *Severity* for error handling according to OPC 40083.

18.2 ZoneOffNormalAlarmType

The *ZoneOffNormalAlarmType* represent zone-related text messages (alarms, error messages, warnings, information) of the peripheral device and is a subtype of *HelpOffNormalAlarmType* as defined in OPC 40083.

NOTE: For messages related to the whole device, the *HelpOffNormalAlarmType* shall be used.

Table 19 - ZoneOffNormalAlarmType Definition

| Attribute | Value | | | | |
|-----------------------------------|------------------------|------------|----------|----------------|----------------|
| BrowseName | ZoneOffNormalAlarmType | | | | |
| IsAbstract | False | | | | |
| References | Node Class | BrowseName | DataType | TypeDefinition | Modelling Rule |
| Subtype of HelpOffNormalAlarmType | | | | | |
| HasProperty | Variable | Zoneld | Nodeld | .PropertyType | M, R |

Zoneld specifies the *Nodeld* of the related Zone. In case of medium or high severity, the client can prevent the use of this zone.

19 Profiles and Namespaces

19.1 Namespace Metadata

Table 20 defines the namespace metadata for this specification. 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 Part5 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 Part5.

The version information is also provided as part of the *ModelTableEntry* in the *UANodeSet XML file*. The *UANodeSet XML schema* is defined in Part 6.

Table 20 – NamespaceMetadata Object for this Specification

| Attribute | Value | | |
|-------------|---|----------------|---|
| BrowseName | http://opcfoundation.org/UA/PlasticsRubber/HotRunner/ | | |
| References | BrowseName | DataType | Value |
| HasProperty | NamespaceUri | String | http://opcfoundation.org/UA/PlasticsRubber/HotRunner/ |
| HasProperty | NamespaceVersion | String | RC 1.00.1 |
| HasProperty | NamespacePublicationDate | DateTime | 2019-09-10 12:00:00 |
| HasProperty | IsNamespaceSubset | Boolean | False |
| HasProperty | StaticNodeldTypes | IdType[] | {Numeric} |
| HasProperty | StaticNumericNodeldRange | NumericRange[] | Null |
| HasProperty | StaticStringNodeldPattern | String | Null |

19.2 Conformance Units and Profiles

This chapter defines the corresponding profiles and conformance units for the OPC UA Information Model for OPC 40082-2. *Profiles* are named groupings of conformance units. *Facets* are profiles that will be combined with other *Profiles* to define the complete functionality of an OPC UA *Server* or *Client*. The following tables specify the facets available for *Servers* that implement the OPC 40082-2 Information Model companion specification.

NOTE: The names of the supported profiles are available in the *Server Object* under *ServerCapabilities.ServerProfileArray*

Table 21 – OPC 40082-2 Basic Server Facet Definition

| Conformance Unit | Description | Optional/ Mandatory |
|--|--|------------------------|
| OPC 40082-2 Basic | Support of <i>HRD_InterfaceType</i> and all mandatory child elements giving information on the hot runner device itself, the current configuration and status. | M |
| Profile | | |
| ComplexType Server Facet (defined in OPC UA Part 7) | | M |
| Method Server Facet (defined in OPC UA Part 7) | | M |
| BaseDevice_Server_Facet (defined in OPC UA Part 100) | | M |

Table 22 – OPC 40082-2 Alarms Server Facet Definition

| Conformance Unit | Description | Optional/Mandatory |
|---|--|--------------------|
| OPC 40082-2 Alarms | Support of <i>HelpOffNormalAlarmType</i> and <i>ZoneOffNormalAlarmType</i> providing error information. If this facet is supported and a client subscribes to the events, the server shall provide all errors via alarms in addition to the error variables included in the <i>OperationType</i> | M |
| A & C Alarm Server Facet (defined in OPC UA Part 7) | | M |

Table 23 – OPC 40082-2 Diagnostics Server Facet Definition

| Conformance Unit | Description | Optional/Mandatory |
|-------------------------|--|--------------------|
| OPC 40082-2 Diagnostics | Support of OPC 40082-2 Extended and diagnosis functions. Therefore, the component Diagnostics of HRD_InterfaceType is mandatory. | M |

Table 24 – OPC 40082-2 Maintenance Server Facet Definition

| Conformance Unit | Description | Optional/Mandatory |
|-------------------------|---|--------------------|
| OPC 40082-2 Maintenance | Support of OPC 40082-2 Extended and maintenance information. Therefore, the component MaintenanceInformation of HRD_InterfaceType is mandatory. | M |

19.3 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 specification shall not use the standard namespaces.

Table 25 provides a list of mandatory and optional namespaces used in an OPC 40082-2 OPC UA Server.

Table 25 – Namespaces used in an OPC 40082-2 Server

| Namespace | Description | Use |
|--|---|-----------|
| http://opcfoundation.org/UA/ | Namespace for <i>NodeIds</i> and <i>BrowseNames</i> 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 may include types and instances used in a device represented by the server. This namespace shall have namespace index 1. | Mandatory |
| http://opcfoundation.org/UA/DI/ | Namespace for <i>NodeIds</i> and <i>BrowseNames</i> defined in OPC UA Part 100. The namespace index is server specific. | Mandatory |
| http://opcfoundation.org/UA/PlasticsRubber/GeneralTypes/ | Namespace for <i>NodeIds</i> and <i>BrowseNames</i> defined in OPC 40083. The namespace index is server specific. | Mandatory |
| http://opcfoundation.org/UA/PlasticsRubber/HotRunner/ | Namespace for <i>NodeIds</i> and <i>BrowseNames</i> defined in this specification. The namespace index is server specific. | Mandatory |
| Vendor specific types and instances | A server may provide vendor specific types like types derived from <i>MachineType</i> or <i>MachineStatusType</i> or vendor specific instances of devices in a vendor specific namespace. | Optional |

Table 26 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 26 – Namespaces used in this specification

| NamespaceURI | Namespace Index | Example |
|------------------------------|-----------------|---------------|
| http://opcfoundation.org/UA/ | 0 | 0:NodeVersion |

Annex A (normative)

OPC 40082-2 Namespace and mappings

A.1 Namespace and identifiers for OPC 40082-2 Information Model

This appendix defines the numeric identifiers for all of the numeric *NodeIds* defined in this specification. 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 *MachineInformationType ObjectType Node* which has the *ControllerName Property*. The **Name** for the *ControllerName InstanceDeclaration* within the *MachineInformationType* declaration is: *MachineInformationType_ControllerName*.

The *NamespaceUri* for all *NodeIds* defined here is <http://opcfoundation.org/UA/PlasticsRubber/HotRunner/>

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

- <http://www.opcfoundation.org/UA/schemas/PlasticsRubber/HotRunner/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/PlasticsRubber/HotRunner/NodeIds.csv>

A computer processible version of the complete Information Model defined in this specification is also provided. It follows the XML Information Model schema syntax defined in Part 6.

The Information Model Schema released with this version of the specification can be found here:

- [http://www.opcfoundation.org/UA/schemas/PlasticsRubber/HotRunner/1.0/
Opc.Ua.PlasticsRubber.HotRunner.NodeSet2.xml](http://www.opcfoundation.org/UA/schemas/PlasticsRubber/HotRunner/1.0/Opc.Ua.PlasticsRubber.HotRunner.NodeSet2.xml)

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

- <http://www.opcfoundation.org/UA/schemas/PlasticsRubber/HotRunner/Opc.Ua.PlasticsRubber.HotRunner.NodeSet2.xml>