European Plastics and Rubber Machinery

EUROMAP 82.5	OPC UA interfaces for plastics and rubber machinery - Peripheral devices - Part 5: Moulds

Release Candidate 1.0.0, 2024-09-01

EUROMAP 82.5 (Release Candidate 1.0.0) is identical with

OPC 40082-5 (Release Candidate 1.0.0) and VDMA 40082-5:2024-09

www.euromap.org

EUROMAP Technical Commission
c/o VDMA • FV KuG

Lyoner Str. 18
DE 60528 Frankfurt am Main

Phone + 49 (69) 6603-1833
Fax + 49 (69) 6603-2833
E-Mail:euromap@vdma.org

Contents

Figures

Tables

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 FOUDATION 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.

GENERAL PROVISIONS

Should any provision of this Agreement be held to be void, invalid, unenforceable or illegal by a court, the validity and enforceability of the other provisions shall not be affected thereby.

This Agreement shall be governed by and construed under the laws of Germany.

This Agreement embodies the entire understanding between the parties with respect to, and supersedes any prior understanding or agreement (oral or written) relating to, this specification.

Forewords

OPC UA is a machine to machine communication technology to transmit characteristics of products (e.g. manufacturer name, device type or components) and process data (e.g. temperatures, pressures or feed rates). To enable vendor unspecific interoperability the description of product characteristics and process data has to be standardized utilizing technical specifications, the OPC UA companion specifications.

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

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 40082-5 describes the data model for moulds used in plastics and rubber machines (like injection moulding machines) for data exchange during operation. The target of OPC 40082-5 is to provide a unique interface for moulds and machines from different manufacturers to ensure compatibility.

The following functionalities are covered:

– General information about the mould (manufacturer, model, serial number…), current configuration and status.

– Process information like temperatures, pressures etc. for monitoring and process optimization.

– Login of the machine user on the mould

– Maintenance activities

Following functions are not included:

– Safety related signals like emergency stop

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. 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*

http://www.opcfoundation.org/UA/Part1/

OPC 10000-3, *OPC Unified Architecture - Part 3: Address Space Model*

http://www.opcfoundation.org/UA/Part3/

OPC 10000-5, *OPC Unified Architecture - Part 5: Information Model*

http://www.opcfoundation.org/UA/Part5/

OPC 10000-6, *OPC Unified Architecture - Part 6: Mappings*

http://www.opcfoundation.org/UA/Part6/

OPC 10000-7, *OPC Unified Architecture - Part 7: Profiles*

http://www.opcfoundation.org/UA/Part7/

OPC 10000-100, *OPC Unified Architecture - Part 100: Devices*

http://www.opcfoundation.org/UA/Part100/

OPC 40001-1, *OPC UA for Machinery - Part 1: Basic Building Blocks*

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

OPC 40001-2, *OPC UA for Machinery - Part 2: Process Values*

http://www.opcfoundation.org/UA/Machinery/ProcessValues

OPC 40083: *OPC UA interfaces for plastics and rubber machinery – General Type definitions*

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 40082-5 Information Model. For the purposes of this document, the terms and definitions given in the documents referenced in Clause 2 apply.

Note that 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.

The same applies to the Container Concept described in OPC 40083, especially the use of the *Property NodeVersion:*

“Several objects can occur several times in the parent object (e.g. several moulds in one machine). For these, container objects are modelled. The benefit is that all instances are collected in one object so that changes can be easily recognized by using a Property NodeVersion which can be subscribed by clients. According to OPC UA, Part 3, the instances of the container objects shall also trigger a *GeneralModelChangeEvent*.”

3.3 Abbreviations

MES Manufacturing Execution System

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

The following functionalities are covered:

– General information about the mould (manufacturer, model, serial number…), current configuration and status

– Process information: time based and cyclic (temperatures, pressures, …)

– Login of the production machine user on the mould.

– Monitoring (Status, Errors)

– Maintenance information

6 OPC 40082-5 Information Model Overview

The information model structure of OPC 40082-5 consists of the root *ObjectType Mould_InterfaceType*, which contains instances of all other *ObjectTypes* defined in this specification.

The *Mould_InteraceType* is defined in chapter 7. The top level objects are separated by use cases:

- Identification
- Machine state
- Configuration
- User Login handling
- Operation
- Process Monitoring
- Maintenance
- Cycle information.

7 Mould_InterfaceType

7.1 Mould_InterfaceType definition

This OPC UA *ObjectType* is used for the root *Object* representing a mould. It is formally defined in Table 1.

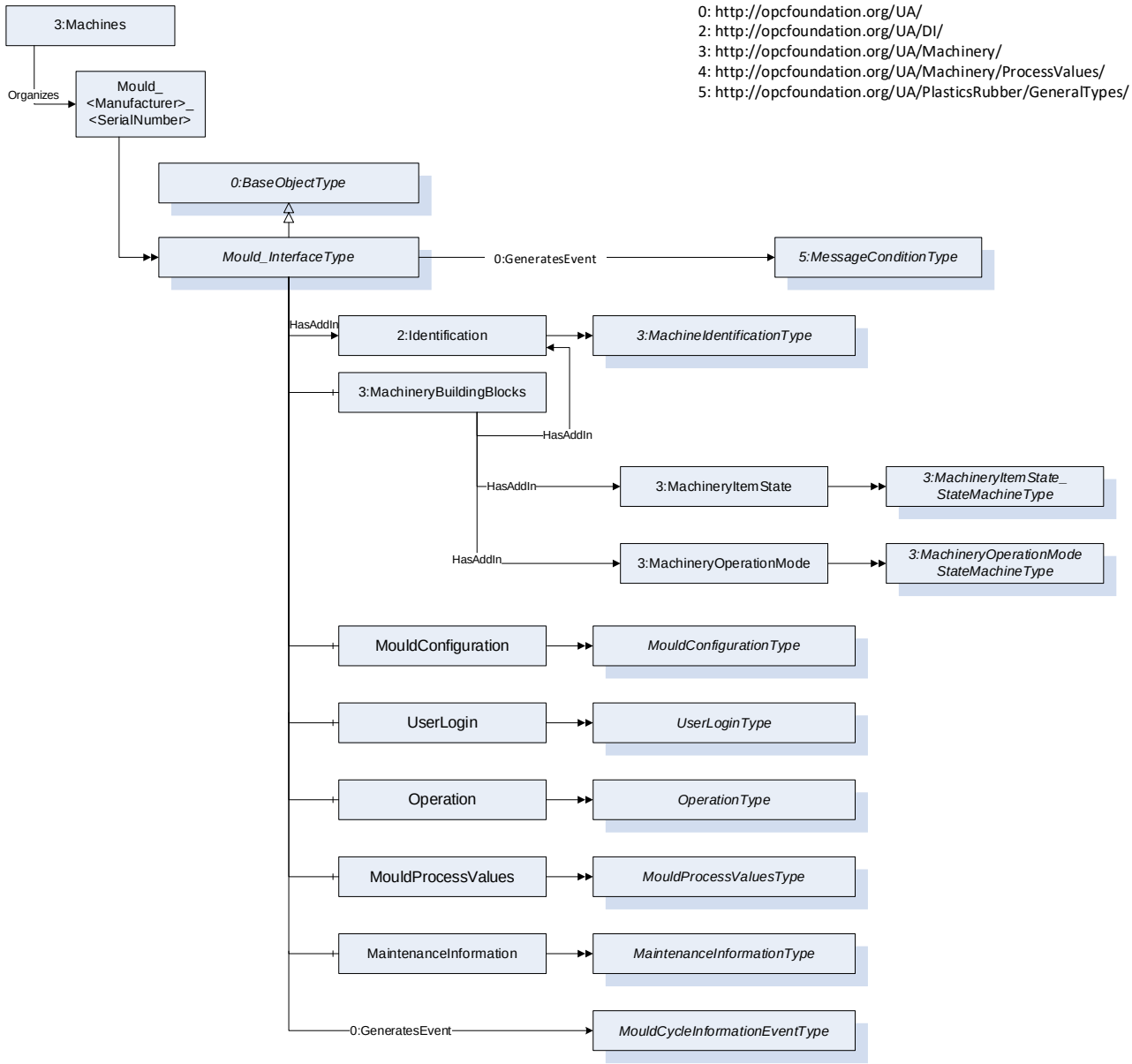

Figure 1 – Mould_InterfaceType Overview

The instance(s) of *Mould_InterfaceType* shall be located under the *Machines Object* of the Server (see OPC UA for Machinery).

NOTE: If the OPC UA server is implemented in the control of the mould so only one instance of *Mould_InterfaceType* will be created. But it is also possible that one OPC UA server is connected to several mould controls as aggregating server. In this case several instances of the *Mould_InterfaceType* will be created.

Table 1 – Mould_InterfaceType Definition

Attribute	Value				
BrowseName	Mould_InterfaceType				
IsAbstract	False				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Other**
Subtype of 0:*BaseObjectType* defined in OPC 10000-5					
0:HasAddIn	Object	2:Identification		3:MachineIdentificationType	M
0:HasComponent	Object	3:MachineryBuildingBlocks		0:FolderType	M
0:HasComponent	Object	MouldConfiguration		MouldConfigurationType	M
0:HasComponent	Object	UserLogin		UserLoginType	O
0:HasComponent	Object	Operation		OperationType	M
0:HasComponent	Object	MouldProcessValues		MouldProcessValuesType	M
0:HasComponent	Object	MaintenanceInformation		MaintenanceInformationType	O
0:GeneratesEvent	Object Type	MouldCycleInformation EventType	Defined in 24		
0:GeneratesEvent	Object Type	5:MessageConditionType	Defined in OPC 40083		
Conformance Units					
OPC 40082-5 Basic					
OPC 40082-5 Maintenance					
OPC 40082-5 Users					

7.2 Identification and MachineryBuildingBlocks

The *MachineIdentificationType* is defined in OPC UA for Machinery (OPC 40001-1) and provides basic information on a machine/device.

For the *InstanceDeclaration* the *ModellingRules* of the *Properties Model* and *DeviceClass* are overridden to mandatory.

The *Object MachineryBuildingBlocks* contains building blocks from OPC UA for Machinery as defined in OPC 40001-1. For this version of OPC 40082-5, the *Object* uses the two *AddIns MachineryItemState* and *MachineryOperationMode*.

Table 2 – Mould_InterfaceType Additional Subcomponents

BrowsePath	References	NodeClass	BrowseName	DataType	TypeDefinition	Other
2:Identification	0:HasProperty	Variable	2:Model	0:LocalizedText	0:PropertyType	M, RO
2:Identification	0:HasProperty	Variable	2:DeviceClass	0:String	0:PropertyType	M, RO
3:MachineryBuilding Blocks	0:HasAddIn	Object	2:Identification		3:MachineIdentificationType	M
3:MachineryBuilding Blocks	0:HasAddIn	Object	3:MachineryItem State		3:MachineryItemState_State MachineType	M
3:MachineryBuilding Blocks	0:HasAddIn	Object	3:MachineryOperationMode		3:MachineryOperationMode StateMachineType	M

The *DeviceClass Property* shall have the value “Mould”.

7.3 MouldConfiguration

This *Object* provides information about the configuration and the cavities of the mould with methods to change these configurations. The *MouldConfigurationType* is defined in chapter 8.

7.4 UserLogin

This *Object* provides the option of logging the user of the production machine on and off the Mould system. The *UserLoginType* is defined in chapter 10.

7.5 Operation

This *Object* provides cycle counters, cycle times and methods to set the machine state and cycle numbers. The *OperationType* is defined in chapter 11.

7.6 MouldProcessValues

This *Object* provides information about the process values of the mould. The *MouldProcessValuesType* is defined in chapter 12.

7.7 MaintenanceInformation

This *Object* provides information about all maintenance activities related to the mould. The *MaintenanceInformationType* is defined in chapter 22.

8 MouldConfigurationType

8.1 MouldConfigurationType definition

The *MouldConfigurationType* provides information about the equipment of the mould and its current configuration. It is formally defined in Table 3.

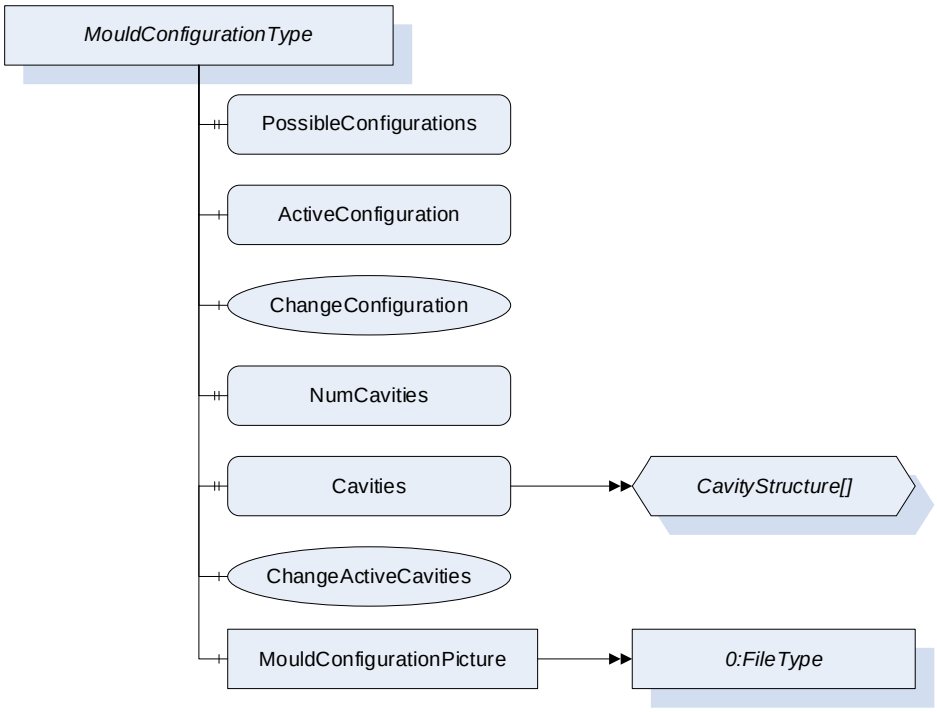

Figure 2 – MouldConfiguration Overview

Table 3 – MouldConfigurationType Definition

Attribute	**Value**				
BrowseName	MouldConfigurationType				
IsAbstract	False				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Other**
Subtype of 0:*BaseObjectType* defined in OPC 10000-5					
0:HasProperty	Variable	PossibleConfigurations	0:String[]	0:PropertyType	M,RO
0:HasComponent	Variable	ActiveConfiguration	0:String	0:BaseDataVariableType	M,RO
0:HasComponent	Method	ChangeConfiguration			M
0:HasProperty	Variable	NumCavities	0:UInt16	0:PropertyType	M,RO
0:HasProperty	Variable	Cavities	CavityStructure[]	0:PropertyType	M,RO
0:HasComponent	Method	ChangeActiveCavities			M
0:HasComponent	Object	MouldConfigurationPicture		0:FileType	O
Conformance Units					
OPC 40082-5 Basic					

NOTE: OPC 40082-5 only includes the information that may change during usage of the mould at the machine. The comprehensive mould description should be stored in a different location (e.g. Asset Administration Shell).

8.2 PossibleConfigurations

Some moulds are family moulds with different possible configurations e.g. by using inserts. The *Property PossibleConfigurations* lists the possible configurations.

8.3 ActiveConfiguration

This Property contains the active configuration of the mould. The value is one of the values given in the *Property PossibleConfigurations.*

8.4 ChangeConfiguration

Description: Method to change the *ActiveConfiguration Variable* of the Mould. Input Argument of the method is the desired configuration, which is given as the corresponding array index in the *PossibleConfigurations Array.*

Signature:

```
ChangeConfiguration(
   [in]    0:UInt16          DesiredConfiguration);
```

Table 4 – ChangeConfiguration Method Arguments

Argument	Description
DesiredConfiguration	Desired configuration of the mould. Array index of the desired configuration in the *PossibleConfigurations* array.

Table 5 – ChangeConfiguration Method AddressSpace Definition

Attribute	Value				
BrowseName	ChangeConfiguration				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Modelling Rule**
HasProperty	Variable	InputArguments	Argument[]	PropertyType	Mandatory

8.5 NumCavities

This *Property* represents the number of cavities of the mould.

8.6 Cavities

This *Structure* gives information about the cavities of the mould. The *CavityStructure DataType* is defined in chapter 9. The length of the one-dimensional array shall be the value of the *Property NumCavities,* so that every array entry represents one cavity of the mould.

8.7 ChangeActiveCavities

Description: Method to change the *CavityActive* value of one single cavity in the *Cavities Array* of the Mould. Input Argument of the method is the desired *CavityActive* value and the index of the cavity in the *Cavities Array*.

Signature:

```
ChangeActiveCavities(
   [in]    0:UInt16          CavityIndex
   [in]    0:Boolean         DesiredActiveValue);
```

Table 6 – ChangeActiveCavities Method Arguments

Argument	Description
CavityIndex	Index of the cavity in the *Cavities* array, of which the value shall be changed.
DesiredActiveValue	Desired value of *CavityActive* at the *CavitiyIndex* array position

Table 7 – ChangeActiveCavities Method AddressSpace Definition

Attribute	Value				
BrowseName	ChangeActiveCavities				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Modelling Rule**
HasProperty	Variable	InputArguments	Argument[]	PropertyType	Mandatory

8.8 MouldConfigurationPicture

The *MouldConfigurationPicture Object* provides a graphic file with information regarding the configuration of the mould. This can be used to display the picture on the HMI of the production machine.

9 CavityStructure DataType

9.1 CavityStructure Definition

This *DataType* provides information about one cavity of the mould. Its composition is formally defined in Table 8.

Table 8 – CavityStructure Structure DataType

Name	Type	Description
CavityStructure	Structure	Subtype of 0:*Structure* defined in OPC 10000-5
CavityId	0:String	unique (in scope of the mould) ID of the cavity
CavityActive	0:Boolean	Information, if the cavity is active

Its representation in the AdressSpace is defined in Table 9.

Table 9 – CavityStructure Definition

Attribute	Value				
BrowseName	CavityStructure				
IsAbstract	False				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Other**
Subtype of 0:*Structure* defined in OPC 10000-5					

10 UserLoginType

10.1 UserLoginType definition

This *ObjectType* allows to automatically transfer the login information of the user logged in on the production machine to the mould so that there no separate login is necessary.

The mould as server shows the user who is currently logged in at the mould and provides *Method*s to pass the user currently logged in to the machine to the mould. The *Methods* for the remote logging in/out shall be called automatically by the machine when a user logs in/out there. The mould can decide whether to accept the login of the machine user via OPC UA or to deny the *Method* with an error code, e.g. if a user is already logged in at the mould itself or the machine is not authorized.

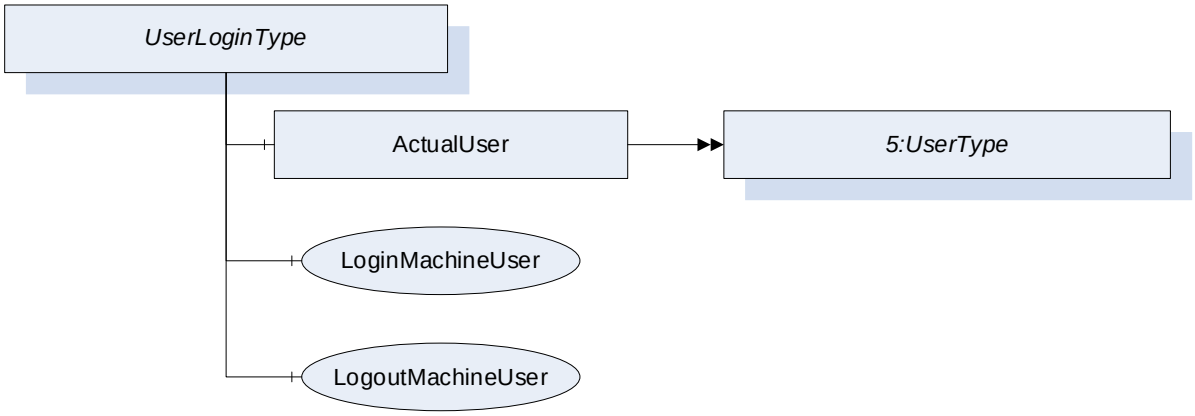

Figure 3 – UserLoginType Overview

Table 10 – UserLoginType Definiton

Attribute	**Value**				
BrowseName	UserLoginType				
IsAbstract	False				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Other**
Subtype of the 0:*BaseObjectType* defined in OPC 10000-5					
HasComponent	Object	ActualUser		5:UserType	M
HasComponent	Method	LoginMachineUser			M
HasComponent	Method	LogoutMachineUser			M
Conformance Units					
OPC 40082-5 Users					

10.2 ActualUser

The *ActualUser Object* holds always the actual active user on the mould system (HMI). If a remote login of a machine user via the *Method LoginMachineUser* is successful, the user should appear in this variable. If the machine user is logged out via the *Method LogoutMachineUser,* the *IsPresent* property of the *ActualUser* becomes *false*. The *UserType* is defined in OPC 40083.

10.3 LoginMachineUser

When the *Method LoginMachineUser* from the mould is called by the machine, the requested user can be logged in to the mould system. If a user is already logged in to the mould or the passed arguments cannot be interpreted or supported, the server can deny the *Method* and *ActualUser* remains unchanged.

Signature

```
LoginMachineUser (
   [in]  0:String              Id,
   [in]  0:String              Name,
   [in]  0:String              CardUid,
   [in]  0:String              UserLevel,
   [in]  0:String              UserRole,
   [in]  0:LocaleId            Language);
```

Table 11 – LoginImmUser Method Arguments

Argument	Description
Id	The *Id* of the user according to 5:*UserType*
Name	The *Name* of the user according to 5:*UserType*
CardUid	The *CardUid* of the user according to 5:*UserType*
UserLevel	The *UserLevel* of the user according to 5:*UserType*
UserRole	The *UserRole* of the user according to 5:*UserType*
Language	The *Language* of the user according to 5:*UserType*

All arguments may contain empty Strings if not supported.

Table 12 – LoginMachineUser Method AddressSpace Definiton

Attribute	Value				
BrowseName	LoginMachineUser				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
0:HasProperty	Variable	0:InputArguments	0:Argument[]	0:PropertyType	0:Mandatory

10.4 LogoutMachineUser

The machine shall report to the mould via the *Method LogoutMachineUser* when a user is logged out. The *IsPresent* property of the ActualUser then becomes false again.

Signature

```
LogoutMachineUser();
```

The *Method* has no *Input*- or *OutputArguments*.

Table 13 – LogoutMachineUser Method AddressSpace Definiton

Attribute	Value				
BrowseName	LogoutMachineUser				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**

11 OperationType

11.1 OperationType definition

The *OperationType* provides cycle counters, cycle times and methods to set the machine state and cycle numbers.

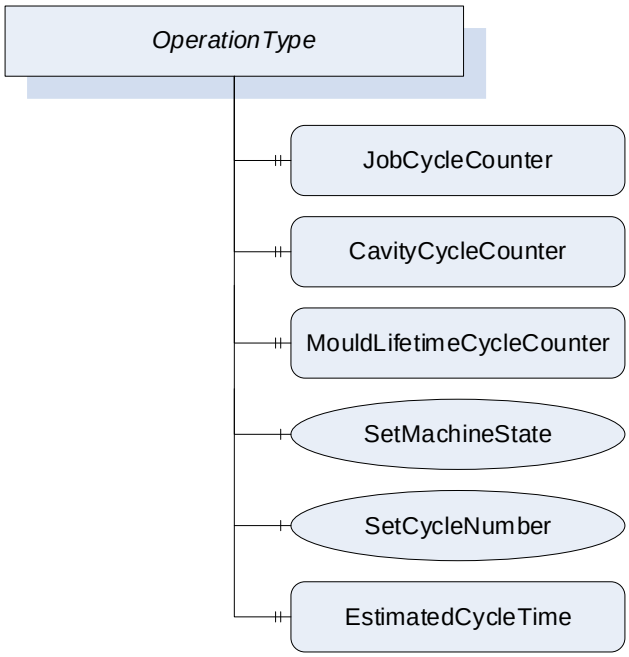

Figure 4 – OperationType Overview

Table 14 – OperationType Definition

Attribute	**Value**				
BrowseName	OperationType				
IsAbstract	False				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Other**
Subtype of the 0:*BaseObjectType* defined in OPC 10000-5					
0:HasComponent	Method	SetMachineState			M
0:HasComponent	Method	SetCycleNumber			M
0:HasComponent	Variable	MouldLifetimeCycleCounter	0:UInt64	0:BaseDataVariableType	M,RO
0:HasComponent	Variable	JobCycleCounter	0:UInt64	0:BaseDataVariableType	O,RO
0:HasComponent	Variable	CavityCycleCounter	0:UInt64[]	0:BaseDataVariableType	O,RO
0:HasProperty	Variable	EstimatedCycleTime	0:Duration	0:PropertyType	O,RW
Conformance Units					
OPC 40082-5 Basic					

11.2 SetMachineState

Method to set the *MachineryItemState*. The input argument of this method is the desired *State*. Allowed values are the *StateNumber* values in the *StateType* defined the *MachineryItemState_StateMachineType Definition* in OPC 40001-1: 0 (*NotAvailable), 1 (OutOfService), 2 (NotExecuting), 3 (Executing)*.

Note: Not all *States* have to be supported by the Mould. In this case, the server shall respond with the status code “Bad_NotSupported”.

Signature

```
SetMachineState (
   [in] 0:UInt32            DesiredState);
```

Table 15 – SetMachineState Method Arguments

Argument	Description
DesiredState	Desired MachineryItemState

Table 16 – SetMachineState Method AddressSpace Definition

Attribute	Value				
BrowseName	SetMachineState				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Modelling Rule**
HasProperty	Variable	InputArguments	Argument[]	PropertyType	Mandatory

11.3 SetCycleNumber

Description: Method to set the cycle number of the Mould to synchronize it with the cycle number of the production machine.

Signature:

```
SetCycleNumber(
   [in]    0:UInt64         JobCycleNumber);
```

Table 17 – SetCycleNumber Method Arguments

Argument	Description
JobCycleNumber	Number, to which the *JobCycleNumber* of the Mould shall be set. The next *MouldCycleInformationEvent* will use this value.

Table 18 – SetCycleNumber Method AddressSpace Definition

Attribute	Value				
BrowseName	SetCycleNumber				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Modelling Rule**
HasProperty	Variable	InputArguments	Argument[]	PropertyType	Mandatory

11.4 MouldLifetimeCycleCounter

Total number of production cycles since the first operation of the mould. The value shall increase with every production cycle and shall not be resettable.

11.5 JobCycleCounter

Number of production cycles in the current job. Increases with every production cycle. Can be set with the *SetCycleNumber* method.

11.6 CavityCycleCounter

Number of production cycles for each cavity of the mould. This *Variable* is a one-dimensional array with the same size as the *Cavities Array.*

11.7 EstimatedCycleTime

The estimated cycle time for one production cycle of the current job.

12 MouldProcessValuesType

12.1 MouldProcessValuesType definition

The *MouldProcessValuesType* provides information about the process values of the mould and is formally defined in Table 19.

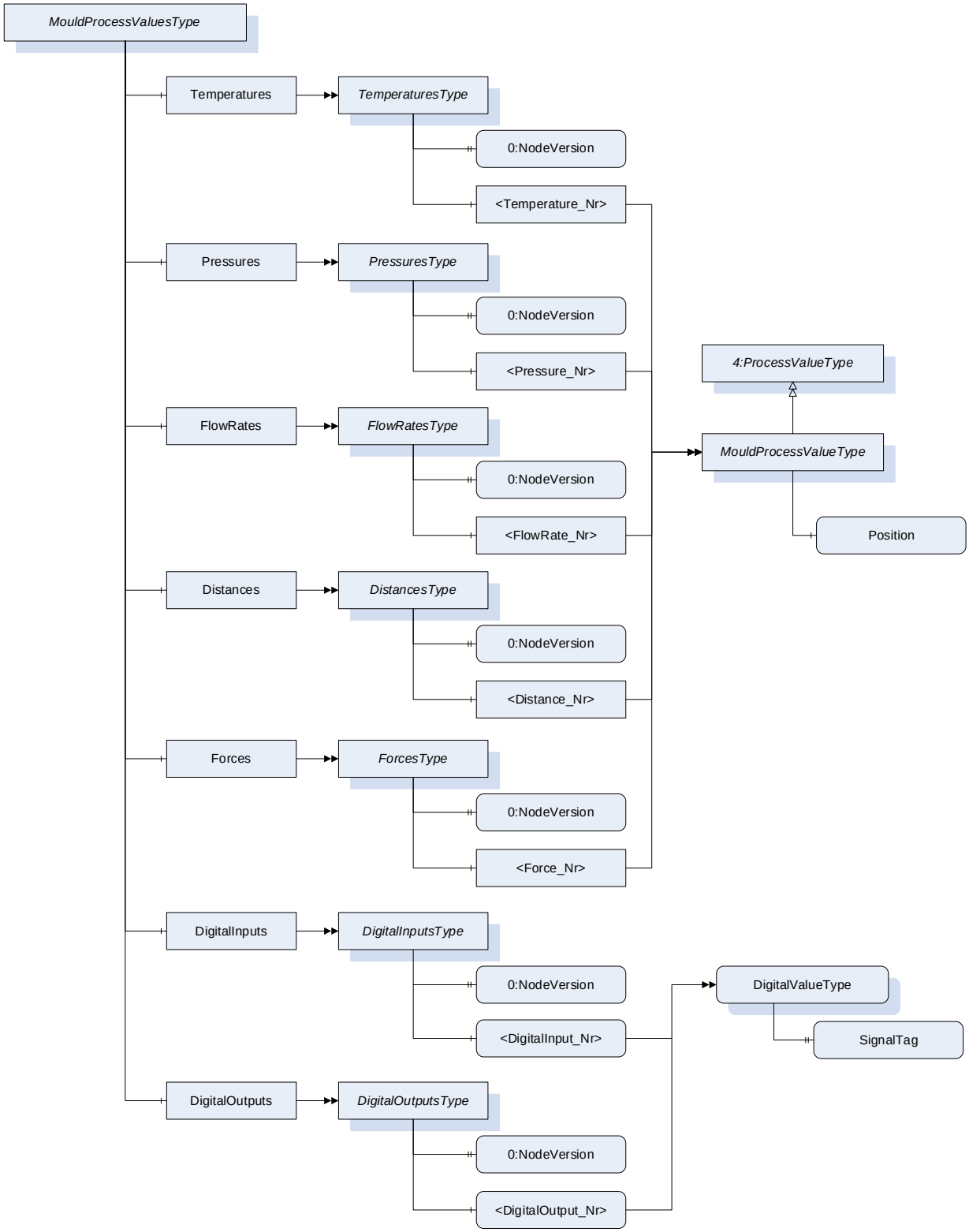

Figure 5 – MouldProcessValuesType Overview

Table 19 – MouldProcessValuesType Definition

Attribute	**Value**				
BrowseName	MouldProcessValuesType				
IsAbstract	False				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Other**
Subtype of 0:*BaseObjectType* defined in OPC 10000-5					
0:HasComponent	Object	Temperatures		TemperaturesType	O
0:HasComponent	Object	Pressures		PressuresType	O
0:HasComponent	Object	FlowRates		FlowRatesType	O
0:HasComponent	Object	Distances		DistancesType	O
0:HasComponent	Object	Forces		ForcesType	O
0:HasComponent	Object	DigitalInputs		DigitalInputsType	O
0:HasComponent	Object	DigitalOutputs		DigitalOutputsType	O
Conformance Units					
OPC 40082-5 Basic					

12.2 Temperatures

Provides information about all monitored temperatures in the mould. The *TemperaturesType* is defined in chapter 13.

12.3 Pressures

Provides information about all monitored pressures in the mould. The *PressuresType* is defined in chapter 14.

12.4 FlowRates

Provides information about all monitored flow rates in the mould. The *FlowRatesType* is defined in chapter 15.

12.5 Distances

Provides information about all monitored distances in the mould. The *DistancesType* is defined in chapter 16.

12.6 Forces

Provides information about all monitored forces in the mould. The *ForcesType* is defined in chapter 17.

12.7 DigitalInputs

Provides information about all digital inputs in the mould. The *DigitalInputsType* is defined in chapter 18.

12.8 DigitalOutputs

Provides information about all digital outputs in the mould. The *DigitalOutputsType* is defined in chapter 19.

13 TemperaturesType

13.1 TemperatureType definition

The *TemperaturesType* is a container object, which provides information about the monitored temperatures in the mould and is formally defined in Table 20.

Table 20 – TemperaturesType Definition

Attribute	**Value**				
BrowseName	TemperaturesType				
IsAbstract	False				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Other**
Subtype of 0:*BaseObjectType* defined in OPC 10000-5					
0:HasProperty	Variable	0:NodeVersion	0:String	0:PropertyType	M, RO
0:HasComponent	Object	<Temperature_Nr>		MouldProcessValueType	MP
Conformance Units					
OPC 40082-5 Temperatures					

The <*Temperature_Nr*> provides a placeholder for the Temperatures. When instances are created the *BrowseNames* shall be “<Temperature_Nr>” where ”Nr” is a three-digit number with leading zeros, starting with “001”. The *MouldProcessValueType* is defined in 20.

14 PressuresType

14.1 PressuresType definition

The *PressuresType* is a container object, which provides information about the monitored pressures in the mould and is formally defined in Table 21.

Table 21 – PressuresType Definition

Attribute	**Value**				
BrowseName	PressuresType				
IsAbstract	False				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Other**
Subtype of 0:*BaseObjectType* defined in OPC 10000-5					
0:HasProperty	Variable	0:NodeVersion	0:String	0:PropertyType	M, RO
0:HasComponent	Object	<Pressure_Nr>		MouldProcessValueType	MP
Conformance Units					
OPC 40082-5 Pressures					

The <*Pressure_Nr*> provides a placeholder for the Pressures. When instances are created the *BrowseNames* shall be “<Pressure_Nr>” where ”Nr” is a three-digit number with leading zeros, starting with “001”. The *MouldProcessValueType* is defined in 20.

15 FlowRatesType

15.1 FlowRatesType definition

The *FlowRatesType* is a container object, which provides information about the monitored flow rates in the mould. This might be flow rates of injected material, but also flow rates of coolants or hydraulic fluids. It is formally defined in Table 22.

Table 22 – FlowRatesType Definition

Attribute	Value				
BrowseName	FlowRatesType				
IsAbstract	False				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Other**
Subtype of 0:*BaseObjectType* defined in OPC 10000-5					
0:HasProperty	Variable	0:NodeVersion	0:String	0:PropertyType	M, RO
0:HasComponent	Object	<FlowRate_Nr>		MouldProcessValueType	MP
Conformance Units					
OPC 40082-5 FlowRates					

The <*FlowRate_Nr*> provides a placeholder for the FlowRates. When instances are created the *BrowseNames* shall be “<FlowRate_Nr>” where ”Nr” is a three-digit number with leading zeros, starting with “001”.

The *MouldProcessValueType* is defined in 20.

16 DistancesType

16.1 DistancesType definition

The *DistancesType* is a container object, which provides information about the monitored distances in the mould. Examples for distances include breathing, offset, slider position. It is formally defined in Table 23.

Table 23 – DistancesType Definition

Attribute	Value				
BrowseName	DistancesType				
IsAbstract	False				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Other**
Subtype of 0:*BaseObjectType* defined in OPC 10000-5					
0:HasProperty	Variable	0:NodeVersion	0:String	0:PropertyType	M, RO
0:HasComponent	Object	<Distance_Nr>		MouldProcessValueType	MP
Conformance Units					
OPC 40082-5 Distances					

The <*Distance_Nr*> provides a placeholder for the Distances. When instances are created the *BrowseNames* shall be “<Distance_Nr>” where ”Nr” is a three-digit number with leading zeros, starting with “001”.

The *MouldProcessValueType* is defined in 20.

17 ForcesType

17.1 ForcesType definition

The *ForcesType* is a container object, which provides information about the monitored forces in the mould and is formally defined in Table 24.

Table 24 – ForcesType Definition

Attribute	**Value**				
BrowseName	ForcesType				
IsAbstract	False				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Other**
Subtype of 0:*BaseObjectType* defined in OPC 10000-5					
0:HasProperty	Variable	0:NodeVersion	0:String	0:PropertyType	M, RO
0:HasComponent	Object	<Force_Nr>		MouldProcessValueType	MP
Conformance Units					
OPC 40082-5 Forces					

The <*Force_Nr*> provides a placeholder for the Forces. When instances are created the *BrowseNames* shall be “<Force_Nr>” where ”Nr” is a three-digit number with leading zeros, starting with “001”.

The *MouldProcessValueType* is defined in 20.

18 DigitalInputsType

18.1 DigitalInputsType definition

The *DigitalInputsType* is a container object, which provides information about the digital inputs of the mould and is formally defined in Table 25.

Table 25 – DigitalInputsType Definition

Attribute	**Value**				
BrowseName	DigitalInputsType				
IsAbstract	False				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Other**
Subtype of 0:*BaseObjectType* defined in OPC 10000-5					
0:HasProperty	Variable	0:NodeVersion	0:String	0:PropertyType	M, RO
0:HasComponent	Variable	<DigitalInput_Nr>	0:Boolean	DigitalValueType	MP
Conformance Units					
OPC 40082-5 DigitalInputs					

The <*DigitalInput_Nr*> provides a placeholder for the DigitalInputs. When instances are created the *BrowseNames* shall be “<DigitalInput_Nr>” where ”Nr” is a three-digit number with leading zeros, starting with “001”.

The DigitalValueType is defined in 21.

19 DigitalOutputsType

19.1 DigitalOutputsType definition

The *DigitalOutputsType* is a container object, which provides information about the digital outputs in the mould and is formally defined in Table 26.

Table 26 – DigitalOutputsType Definition

Attribute	**Value**				
BrowseName	DigitalOutputsType				
IsAbstract	False				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Other**
Subtype of 0:*BaseObjectType* defined in OPC 10000-5					
0:HasProperty	Variable	0:NodeVersion	0:String	0:PropertyType	M, RO
0:HasComponent	Variable	<DigitalOutput_Nr>	0:Boolean	DigitalValueType	MP
Conformance Units					
OPC 40082-5 DigitalOutputs					

The <*DigitalOutput_Nr*> provides a placeholder for the DigitalOutputs. When instances are created the *BrowseNames* shall be “<DigitalOutput_Nr>” where ”Nr” is a three-digit number with leading zeros, starting with “001”.

The DigitalValueType is defined in 21.

20 MouldProcessValueType

20.1 MouldProcessValueType definition

The *MouldProcessValueType* is the *ObjectType* used for the process values of the mould in the *MouldProcessValuesType*. It is formally defined in Table 27. It is a Subtype of the *ProcessValueType*, which is defined in OPC 40001-2 and adds information about the position of the measurement point.

For further information regarding the name and description of the measuring point, the *Node Attributes DisplayName* and *Description* and the mandatory *Property SignalTag* in the *ProcessValueType.*shall be used.

Table 27 – MouldProcessValueType Definition

Attribute	**Value**				
BrowseName	MouldProcessValueType				
IsAbstract	False				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Other**
Subtype of 4:ProcessValueType defined in 40001-2					
0:HasComponent	Variable	Position	0:3DCartesianCoordinates	0:3DCartesianCoordinatesType	O,RO
Conformance Units					

20.2 Position

The optional *Variable Position* provides the position of the measured value in cartesian coordinates.

The coordinate system shall be right-handed.

The origin of the coordinate system should be centered in the first parting plane of the mould. The Z-axis should point in the direction of the injection of the first component, while the X-axis shall point towards the control panel of the machine. Figure 6 shows an example of the orientation of the coordinate system.

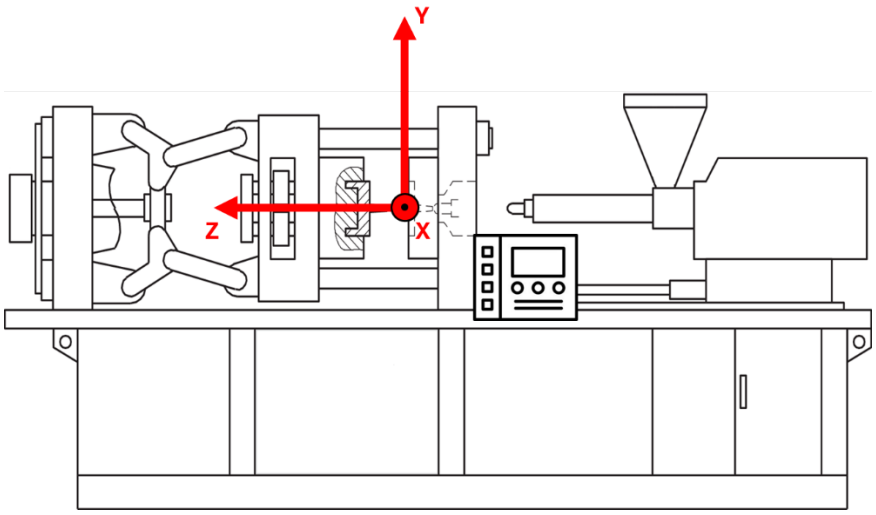

Figure 6 – Coordinate system orientation

The *3DCartesianCoordinatesType* is defined in OPC 10000-5. For the *InstanceDeclaration* the *ModelingRules* of the *Property LengthUnit* are overridden to mandatory.

Table 28 – MouldProcessValueType Additional Subcomponents

BrowsePath	References	NodeClass	BrowseName	DataType	TypeDefinition	Other
Position	0:HasProperty	Variable	0:LengthUnit	0:EUInformation	0:PropertyType	M

21 DigitalValueType

21.1 DigitalValueType definition

The *DigitalValueType VariableType* is used for the digital inputs and digital outputs of the mould and provides the digital values along with meta information. The *DigitalValueType* is formally defined in Table 29.

Table 29 – DigitalValueType Definition

Attribute	Value				
BrowseName	DigitalValueType				
IsAbstract	False				
ValueRank	Scalar				
DataType	0:Boolean				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Other**
Subtype of 0:*BaseDataVariableType* defined in OPC 10000-5					
0:HasProperty	Variable	SignalTag	0:String	0:PropertyType	M
Conformance Units					

For further information regarding the name and description of the Digital input or output, the *Node Attributes DisplayName* and *Description* shall be used.

21.2 SignalTag

The SignalTag is defined by IRDI as: ABB271#007 which states “defines alphanumeric character sequence uniquely identifying a measuring or control point.”

Note: The *SignalTag* description is the same as in the *SignalType* of OPC 30081: Process Automation Devices – PADIM without direct reference in this specification.

22 MaintenanceInformationType

22.1 MaintenanceInformationType definition

The *MaintenanceInformationType* provides information about all maintenance activities related to the mould and is formally defined in Table 30. Each instance of the *<Maintenance_Nr> placeholder* represents a maintenance activity category (examples: lubrication, inspection…).

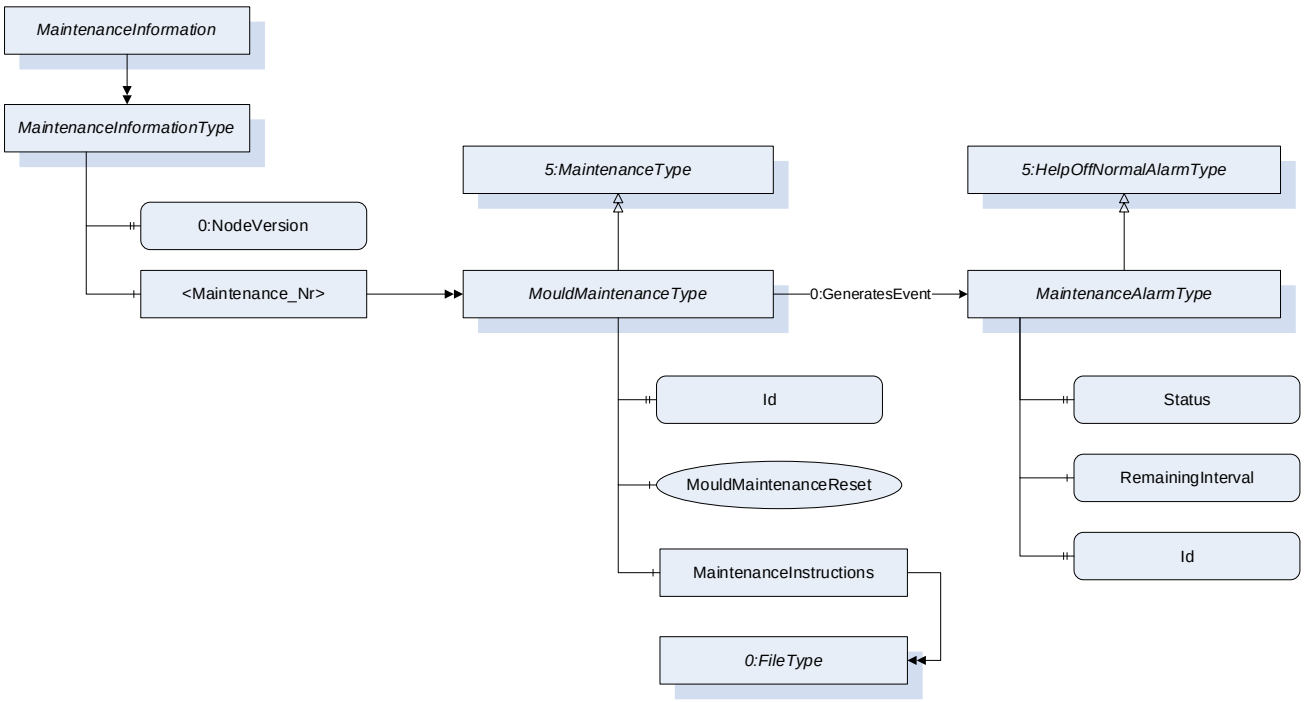

Figure 7 – MaintenanceInformationType Overview

Table 30 – MaintenanceInformationType Definition

Attribute	**Value**				
BrowseName	MaintenanceInformationType				
IsAbstract	False				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Other**
Subtype of the 0:*BaseObjectType* defined in OPC 10000-5					
0:HasProperty	Variable	0:NodeVersion	0:String	0:PropertyType	M, RO
0:HasComponent	Object	<Maintenance_Nr>		MouldMaintenanceType	MP
Conformance Units					
OPC 40082-5 Maintenance					

When instances for *Maintenance* are created, the *BrowseNames* shall be “<Maintenance_Nr>” where Nr is a three-digit number with leading zeros, starting with “001”. The *MouldMaintenanceType* is defined in 23.

23 MouldMaintenanceType

23.1 MouldMaintenanceType definition

The *MouldMaintenanceType* provides information about a single maintenance activity related to the mould and is formally defined in Table 31. The *MouldMaintenanceType* is a subtype of the *MaintenanceType* defined in OPC 40083.

For identification of the maintenance activity (examples: lubrication, inspection…) the *Node Attributes DisplayName* and *Description* shall be used.

Table 31 – MouldMaintenanceType Definition

Attribute	Value				
BrowseName	MouldMaintenanceType				
IsAbstract	False				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Other**
Subtype of the 5:*MaintenanceType* defined in OPC 40083					
0:HasProperty	Variable	Id	0:String	0:PropertyType	M,RO
0:HasComponent	Method	MouldMaintenanceReset			M
0:HasComponent	Object	MaintenanceInstructions		0:FileType	O
0:GeneratesEvent	ObjectType	MaintenanceAlarmType	Defined in 24		
Conformance Units					
OPC 40082-5 Maintenance					

23.2 Id

Unique Id of the maintenance activity.

23.3 MouldMaintenanceReset

The *Method MouldMaintenanceReset* is used to indicate that the maintenance activity has been carried out. Although the *Method Reset* is mandatory in the *MaintenanceType* of OPC 40083, the *Method MouldMaintenanceReset* shall be used when performing maintenance activities at the mould.

The signature of this *Method* is specified below. Table 32 and Table 33 specify the *Arguments* and *AdressSpace* representation, respectively.

Signature

```
MouldMaintenanceReset (
   [in]    0:String          OperatorName,
   [in]    0:String          MachineManufacturer,
   [in]    0:String          MachineSerialNumber,
)
```

Table 32 – MouldMaintenanceReset Method Arguments

Argument	Description
OperatorName	Name of the operator that has carried out the maintenance
MachineManufacturer	Manufacturer of the machine on which was mounted during maintenance
MachineSerialNumber	Serial number of the machine on which was mounted during maintenance

Table 33 – MouldMaintenanceReset Method AddressSpace Definition

Attribute	Value				
BrowseName	MouldMaintenanceReset				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**ModellingRule**
0:HasProperty	Variable	0:InputArguments	0:Argument[]	0:PropertyType	M

23.4 MaintenanceInstructions

The *MaintenanceInstructions Object* provides any file with further information to support the maintenance activity (e.g. Maintenance instructions with pictures and graphics).

24 MaintenanceAlarmType

24.1 MaintenanceAlarmType definition

The *MaintenanceAlarmType* is active when the maintenance *Status* is DUE or WARNING. It is a *SubType* of the *HelpOffNormalAlarmType* (defined in OPC 40083) and is formally defined in Table 34.

e.g.: HelpText = *AdditionalInformation* string.

Table 34 – MaintenanceAlarmType Definition

Attribute	**Value**				
BrowseName	MaintenanceAlarmType				
IsAbstract	True				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Other**
Subtype of the 5:HelpOffNormalAlarmType defined in OPC 40083					
0:HasComponent	Variable	Status	5:MaintenanceStatusEnumeration	0:BaseDataVariableType	M
0:HasComponent	Variable	RemainingInterval	0:Double	0:AnalogUnitType	O
0:HasProperty	Variable	Id	0:String	0:PropertyType	M
Conformance Units					
OPC 40082-5 Maintenance					

25 MouldCycleInformationEventType

25.1 MouldCycleInformationEventType definition

The *MouldCycleInformationEventType* provides information about one production cycle of the mould. The *MouldCycleInformationEvent* is triggered with every Mould closed signal.

Table 35 – MouldCycleInformationEventType Definition

Attribute	**Value**				
BrowseName	MouldCycleInformationEventType				
IsAbstract	True				
References	**Node Class**	**BrowseName**	**DataType**	**TypeDefinition**	**Other**
Subtype of 0:*BaseEventType* defined in OPC UA Part 5					
0:HasComponent	Variable	CycleQuality	5:CycleQualityEnumeration	0:BaseDataVariableType	O
0:HasComponent	Variable	CavityCycleQuality	5:CavityCycleQualityEnumeration[]	0:BaseDataVariableType	O
0:HasComponent	Variable	ActualCycleTime	0:Duration	0:BaseDataVariableType	M
0:HasComponent	Variable	AverageCycleTime	0:Duration	0:BaseDataVariableType	O
0:HasComponent	Variable	JobCycleNumber	0:UInt64	0:BaseDataVariableType	M
0:HasComponent	Variable	MouldLifetimeCycleNumber	0:UInt64	0:BaseDataVariableType	M
0:HasComponent	Variable	MouldCycleState	0:BitFieldMaskDataType	0:OptionSetType	M

25.2 CycleQuality

The *CycleQuality Property* gives information on the quality of the whole cycle. The *CycleQualityEnumeration* is defined in EUROMAP 83.

If the machine is able to evaluate the cycle quality of each cavity, detailed information can be given in the *CavityCycleQuality* Property.

25.3 CavityCycleQuality

The *CavityCycleQuality Property* gives information on the quality of the cycle for each cavity if the machine is able to evaluate this. The *CavityCycleQualityEnumeration* is defined in EUROMAP 83. This *Variable* is a one-dimensional array with the same size as *ActiveCavities.*

25.4 ActualCycleTime

The actual cycle time of the last production cycle.

25.5 AverageCycleTime

The average cycle time of the last production cycles. The choice of sample size and the type of averaging is up to the user.

25.6 JobCycleNumber

The cycle number of the current job. This is the value of the *JobCycleCounter* (defined in 11.5) in the considered cycle.

25.7 MouldLifetimeCycleNumber

The total cycle number of the mould. This is the value of the *MouldLifetimeCycleCounter* (defined in 11.4) in the considered cycle.

25.8 MouldCycleState

The *MouldCycleState* provides information about the mould state in the considered production cycle.

The *MouldCycleState Variable* is from *DataType BitFieldMaskDataType* and from *VariableType OptionSetType.* This *BitFieldMaskDataType* represents a bit mask up to 32 bits where individual bits can be written without modifying the other bits. Each array element of the mandatory *OptionSetValues Property* of the *OptionSetType VariableType* contains either the human-readable representation for the corresponding bit used in the option set or an empty *LocalizedText* for a bit that has no specific meaning. The order of the bits of the bit mask maps to a position of the array, i.e. the first bit (least significant bit) maps to the first entry in the array, etc.

The first 16 bits (starting with the least significant bit) of the bit mask are defined or reserved for future use in this specification. The last 16 bits can be defined by the server. Additional information is listed in Table 36.

The states defined in this variable may all occur simultaneously. This version of the specification defines the following states: Production, Warning, Error, Maintenance.

Table 36 – MouldCycleState OptionSetValues Additional Information

Bit	Value as *LocalizedText*	Description
0	Production	The considered cycle was a production cycle
1	Warning	A warning occurred in the considered cycle. Usually does not lead to termination of the production cycle
2	Error	An error occurred in the considered cycle, Usually leads to termination of the production cycle
3	Maintenance	The considered cycle was a maintenance cycle
4-15	(empty)	Reserved for future versions of this specification
16-31	(empty)	Defined by server

26 Profiles and Conformance Units

This chapter defines the corresponding profiles and conformance units for the OPC UA Information Model for OPC 40082-5. *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.*

26.1 Conformance Units

This chapter defines the corresponding *Conformance Unit* for OPC 40082-5.

Table 37 – Conformance Units for OPC 40082-5

Category	Title	Description
Server	OPC 40082-5 Basic	Support of *Mould_InterfaceType* and all mandatory child elements giving information on the dosing system and its status. There is at least one instance of the *Mould_InterfaceType* in the *Machines Object*.
Server	OPC 40082-5 Maintenance	Support of *MaintenanceInformationType* and all mandatory child elements giving information on the maintenance activities on the mould.
Server	OPC 40082-5 Users	Support of *UserLoginType* to enable automatic login of the machine user onto the mould controller.
Server	OPC 40082-5 Temperatures	Support of *TemperaturesType* and *MouldProcessValueType* to monitor temperature process values of the mould.
Server	OPC 40082-5 Pressures	Support of *PressuresType* and *MouldProcessValueType* to monitor pressure process values of the mould.
Server	OPC 40082-5 FlowRates	Support of *FlowRatesType* and *MouldProcessValueType* to monitor flow rate process values of the mould.
Server	OPC 40082-5 Distances	Support of *DistancesType* and *MouldProcessValueType* to monitor distance process values of the mould.
Server	OPC 40082-5 Forces	Support of *ForcesType* and *MouldProcessValueType* to monitor force process values of the mould.
Server	OPC 40082-5 DigitalInputs	Support of *DigitalInputsType* and *DigitalValueType* to monitor the digital inputs of the mould.
Server	OPC 40082-5 DigitalOutputs	Support of *DigitalOutputsType* and *DigitalValueType* to monitor the digital outputs of the mould.

26.2 Profiles

26.2.1 Profile list

The following tables specify the facets available for *Servers* that implement the OPC 40082-5 Information Model companion specification.

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

Table 38 lists all Profiles defined in this document and defines their URIs.

Table 38 – Profile URIs for OPC 40082-5

Profile	URI
OPC 40082-5 Basic Server Profile	http://opcfoundation.org/UA-Profile/PlasticsRubber/Moulds/Server/Basic

26.2.2 Server Facets

26.2.2.1 Overview

The following sections specify the *Facets* available for *Servers* that implement the OPC 40082-5 companion specification. Each section defines and describes a *Facet* or *Profile*.

26.2.2.2 OPC 40082-5 Basic Server Profile

Table 39 – OPC 40082-5 Basic Server Profile

Group	Conformance Unit / Profile Title	Mandatory / Optional
Server	0:Embedded Server 2017 (defined in OPC 10000-7)	M
Server	0:ComplexType Server Facet (defined in OPC 10000-7)	M
Server	0:Standard Event Subscription Server Facet (defined in OPC 10000-7)	M
Server	0:Method Server Facet (defined in OPC 10000-7)	M
Server	2:BaseDevice Server Facet (defined in OPC 10000-100)	M
Server	0:A & C Alarm Server Facet (defined in OPC 10000-7)	O
OPC 40082-5	OPC 40082-5 Basic	M
OPC 40082-5	OPC 40082-5 Maintenance	O
OPC 40082-5	OPC 40082-5 Users	O
OPC 40082-5	OPC 40082-5 Temperatures	O
OPC 40082-5	OPC 40082-5 Pressures	O
OPC 40082-5	OPC 40082-5 FlowRates	O
OPC 40082-5	OPC 40082-5 Distances	O
OPC 40082-5	OPC 40082-5 Forces	O
OPC 40082-5	OPC 40082-5 DigitalInputs	O
OPC 40082-5	OPC 40082-5 DigitalOutputs	O

27 Namespaces

27.1 Namespace Metadata

Table 40 defines the namespace metadata for this document. The *Object* is used to provide version information for the namespace and an indication about static *Nodes*. Static *Nodes* are identical for all *Attributes* in all *Servers*, including the *Value Attribute*. See OPC 10000-5 for more details.

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

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

Table 40 – NamespaceMetadata Object for this Document

Attribute	**Value**	
BrowseName	http://opcfoundation.org/UA/PlasticsRubber/Moulds/	
Property	**DataType**	**Value**
NamespaceUri	String	http://opcfoundation.org/UA/PlasticsRubber/Moulds/
NamespaceVersion	String	RC 1.0.0
NamespacePublicationDate	DateTime	2024-09-01
IsNamespaceSubset	Boolean	False
StaticNodeIdTypes	IdType []	0
StaticNumericNodeIdRange	NumericRange []	
StaticStringNodeIdPattern	String	

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

27.2 Handling of OPC UA Namespaces

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

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

Table 41 provides a list of mandatory and optional namespaces used in an OPC 40082-5 OPC UA *Server*.

Table 41 – Namespaces used in a OPC 40082-5 Server

NamespaceURI	Description	Use
http://opcfoundation.org/UA/	Namespace for *NodeIds* and *BrowseNames* defined in the OPC UA specification. This namespace shall have namespace index 0.	Mandatory
Local Server URI	Namespace for nodes defined in the local server. This namespace shall have namespace index 1.	Mandatory
http://opcfoundation.org/UA/DI/	Namespace for *NodeIds* and *BrowseNames* defined in OPC 10000-100. The namespace index is *Server* specific.	Mandatory
http://opcfoundation.org/UA/Machinery/	Namespace for *NodeIds* and *BrowseNames* defined in OPC UA for Machinery – Part 1: Basic Building Blocks (OPC 40001-1). The namespace index is *Server* specific.	Mandatory
http://opcfoundation.org/UA/Machinery/ProcessValues/	Namespace for *NodeIds* and *BrowseNames* defined in OPC UA for Machinery – Part 2: Process Values (OPC 40001-2). The namespace index is *Server* specific.	Mandatory
http://opcfoundation.org/UA/PlasticsRubber/ GeneralTypes/	Namespace for *NodeIds* and *BrowseNames* defined in OPC 40083. The namespace index is server specific.	Mandatory
http://opcfoundation.org/UA/PlasticsRubber/Moulds/	Namespace for *NodeIds* and *BrowseNames* defined in this document. The namespace index is *Server* specific.	Mandatory
Vendor specific types	A *Server* may provide vendor-specific types like types derived from *ObjectTypes* defined in this document in a vendor-specific namespace.	Optional
Vendor specific instances	A *Server* provides vendor-specific instances of the standard types or vendor-specific instances of vendor-specific types in a vendor-specific namespace. It is recommended to separate vendor specific types and vendor specific instances into two or more namespaces.	Mandatory

Table 42 provides a list of namespaces and their indices used for *BrowseNames* in this document. The default namespace of this document is not listed since all *BrowseNames* without prefix use this default namespace.

Table 42 – Namespaces used in this document

NamespaceURI	Namespace Index	Example
http://opcfoundation.org/UA/	0	0:EngineeringUnits
http://opcfoundation.org/UA/DI/	2	2:DeviceClass
http://opcfoundation.org/UA/Machinery/	3	3:MachineIdentificaionType
http://opcfoundation.org/UA/Machinery/ProcessValues/	4	4:ProcessValueType
http://opcfoundation.org/UA/PlasticsRubber/GeneralTypes/	5	5:MachineInformationType

Annex A
(normative)

OPC 40082-5 Namespace and mappings

A.1 NodeSet and supplementary files for OPC 40082-5 Information Model

The OPC 40082-5 *Information Model* is identified by the following URI:

http://opcfoundation.org/UA/PlasticsRubber/Moulds/

Documentation for the NamespaceUri can be found here.

The *NodeSet* associated with this version of specification can be found here:

https://reference.opcfoundation.org/nodesets/?u=http://opcfoundation.org/UA/PlasticsRubber/Moulds/&v=1.0.0&i=1

The *NodeSet* associated with the latest version of the specification can be found here:

https://reference.opcfoundation.org/nodesets/?u=http://opcfoundation.org/UA/PlasticsRubber/Moulds/&i=1

Supplementary files for the OPC 40082-5 *Information Model* can be found here:

https://reference.opcfoundation.org/nodesets/?u=http://opcfoundation.org/UA/PlasticsRubber/Moulds/&v=1.0.0&i=2

The files associated with the latest version of the specification can be found here:

https://reference.opcfoundation.org/nodesets/?u=http://opcfoundation.org/UA/PlasticsRubber/Moulds/&i=2