HEXAGON
PPM

Intergraph Smart Data Validator

Customization Guide

Version 2018 (3.0)
September 2017

Copyright

U.S. Government Restricted Rights Legend

Use, duplication, or disclosure by the government is subject to restrictions as set forth below. For civilian agencies: This was developed at private expense and is "restricted computer software" submitted with restricted rights in accordance with subparagraphs (a) through (d) of the Commercial Computer Software - Restricted Rights clause at 52.227-19 of the Federal Acquisition Regulations ("FAR") and its successors, and is unpublished and all rights are reserved under the copyright laws of the United States. For units of the Department of Defense ("DoD"): This is "commercial computer software" as defined at DFARS 252.227-7014 and the rights of the Government are as specified at DFARS 227.7202-3.

Unpublished - rights reserved under the copyright laws of the United States.

Intergraph Corporation
305 Intergraph Way
Madison, AL 35758

Documentation

Documentation shall mean, whether in electronic or printed form, User's Guides, Installation Guides, Reference Guides, Administrator's Guides, Customization Guides, Programmer's Guides, Configuration Guides and Help Guides delivered with a particular software product.

Other Documentation

Other Documentation shall mean, whether in electronic or printed form and delivered with software or on Intergraph Smart Support, SharePoint, or box.net, any documentation related to work processes, workflows, and best practices that is provided by Intergraph as guidance for using a software product.

Terms of Use

a. Use of a software product and Documentation is subject to the Software License Agreement ("SLA") delivered with the software product unless the Licensee has a valid signed license for this software product with Intergraph Corporation. If the Licensee has a valid signed license for this software product with Intergraph Corporation, the valid signed license shall take precedence and govern the use of this software product and Documentation. Subject to the terms contained within the applicable license agreement, Intergraph Corporation gives Licensee permission to print a reasonable number of copies of the Documentation as defined in the applicable license agreement and delivered with the software product for Licensee's internal, non-commercial use. The Documentation may not be printed for resale or redistribution.

b. For use of Documentation or Other Documentation where end user does not receive a SLA or does not have a valid license agreement with Intergraph, Intergraph grants the Licensee a non-exclusive license to use the Documentation or Other Documentation for Licensee’s internal non-commercial use. Intergraph Corporation gives Licensee permission to print a reasonable number of copies of Other Documentation for Licensee’s internal, non-commercial use. The Other Documentation may not be printed for resale or redistribution. This license contained in this subsection b) may be terminated at any time and for any reason by Intergraph Corporation by giving written notice to Licensee.

Disclaimer of Warranties

Except for any express warranties as may be stated in the SLA or separate license or separate terms and conditions, Intergraph Corporation disclaims any and all express or implied warranties including, but not limited to the implied warranties of merchantability and fitness for a particular purpose and nothing stated in, or implied by, this document or its contents shall be considered or deemed a modification or amendment of such disclaimer. Intergraph believes the information in this publication is accurate as of its publication date.

The information and the software discussed in this document are subject to change without notice and are subject to applicable technical product descriptions. Intergraph Corporation is not responsible for any error that may appear in this document.

The software, Documentation and Other Documentation discussed in this document are furnished under a license and may be used or copied only in accordance with the terms of this license. THE USER OF THE SOFTWARE IS EXPECTED TO MAKE THE FINAL EVALUATION AS TO THE USEFULNESS OF THE SOFTWARE IN HIS OWN ENVIRONMENT.

Intergraph is not responsible for the accuracy of delivered data including, but not limited to, catalog, reference and symbol data. Users should verify for themselves that the data is accurate and suitable for their project work.

Limitation of Damages

IN NO EVENT WILL INTERGRAPH CORPORATION BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL INCIDENTAL, SPECIAL, OR PUNITIVE DAMAGES, INCLUDING BUT NOT LIMITED TO, LOSS OF USE OR PRODUCTION, LOSS OF REVENUE OR PROFIT, LOSS OF DATA, OR CLAIMS OF THIRD PARTIES, EVEN IF INTERGRAPH CORPORATION HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

UNDER NO CIRCUMSTANCES SHALL INTERGRAPH CORPORATION'S LIABILITY EXCEED THE AMOUNT THAT INTERGRAPH CORPORATION HAS BEEN PAID BY LICENSEE UNDER THIS AGREEMENT AT THE TIME THE CLAIM IS MADE. EXCEPT WHERE PROHIBITED BY APPLICABLE LAW, NO CLAIM, REGARDLESS OF FORM, ARISING OUT OF OR IN CONNECTION WITH THE SUBJECT MATTER OF THIS DOCUMENT MAY BE BROUGHT BY LICENSEE MORE THAN TWO (2) YEARS AFTER THE EVENT GIVING RISE TO THE CAUSE OF ACTION HAS OCCURRED.

IF UNDER THE LAW RULED APPLICABLE ANY PART OF THIS SECTION IS INVALID, THEN INTERGRAPH LIMITS ITS LIABILITY TO THE MAXIMUM EXTENT ALLOWED BY SAID LAW.

Export Controls

Intergraph Corporation's commercial-off-the-shelf software products, customized software and/or third-party software, including any technical data related thereto ("Technical Data"), obtained from Intergraph Corporation, its subsidiaries or distributors, is subject to the export control laws and regulations of the United States of America. Diversion contrary to U.S. law is prohibited. To the extent prohibited by United States or other applicable laws, Intergraph Corporation software products, customized software, Technical Data, and/or third-party software, or any derivatives thereof, obtained from Intergraph Corporation, its subsidiaries or distributors must not be exported or re-exported, directly or indirectly (including via remote access) under the following circumstances:

a. To Cuba, Iran, North Korea, the Crimean region of Ukraine, or Syria, or any national of these countries or territories.

b. To any person or entity listed on any United States government denial list, including, but not limited to, the United States Department of Commerce Denied Persons, Entities, and Unverified Lists, the United States Department of Treasury Specially Designated Nationals List, and the United States Department of State Debarred List (https://build.export.gov/main/ecr/eg_main_023148).

c. To any entity when Customer knows, or has reason to know, the end use of the software product, customized software, Technical Data and/or third-party software obtained from Intergraph Corporation, its subsidiaries or distributors is related to the design, development, production, or use of missiles, chemical, biological, or nuclear weapons, or other un-safeguarded or sensitive nuclear uses.

d. To any entity when Customer knows, or has reason to know, that an illegal reshipment will take place.

e. Any questions regarding export/re-export of relevant Intergraph Corporation software product, customized software, Technical Data and/or third-party software obtained from Intergraph Corporation, its subsidiaries or distributors, should be addressed to PPM's Export Compliance Department, 305 Intergraph Way, Madison, Alabama 35758 USA or at exportcompliance@intergraph.com. Customer shall hold harmless and indemnify PPM and Hexagon Group Company for any causes of action, claims, costs, expenses and/or damages resulting to PPM or Hexagon Group Company from a breach by Customer.

Trademarks

Intergraph®, the Intergraph logo®, Intergraph Smart®, SmartPlant®, SmartMarine, SmartSketch®, SmartPlant Cloud®, PDS®, FrameWorks®, I-Route, I-Export, ISOGEN®, SPOOLGEN, SupportManager®, SupportModeler®, SAPPHIRE®, TANK, PV Elite®, CADWorx®, CADWorx DraftPro®, GTSTRUDL®, and CAESAR II® are trademarks or registered trademarks of Intergraph Corporation or its affiliates, parents, subsidiaries. Hexagon and the Hexagon logo are registered trademarks of Hexagon AB or its subsidiaries. Microsoft and Windows are registered trademarks of Microsoft Corporation. MicroStation is a registered trademark of Bentley Systems, Inc. Other brands and product names are trademarks of their respective owners.

Contents

Preface

This document contains information about customizing the configuration of Intergraph Smart® Data Validator. This document is intended for system administrators and users with the correct permissions, who are configuring and customizing the installation of Smart Data Validator.

Sample customization code

Sample customization code is supplied as part of the standard installation process for all subjects discussed in this document. The sample code can be found at the following location:

```
<drive>:\Program Files (x86)\Smart\SDV\2015\CustomExamples\
```

SmartPlant® Foundation assemblies can be found at the following location:

```
<drive>:\Program Files(x86)\SmartPlant\Foundation\2014\Server\Bin
```

★ IMPORTANT Ensure that all referenced assemblies are in place before starting the customization. For more information on specific code examples, see *Sample Custom Code* (see "*Custom validation report writer*" on page 18).

SECTION 1

Customizing the Validation Report Writer

Validation report writers are used to generate validation reports in a given format when validation errors occur in the imported data. There are two validation report writers delivered with Smart Data Validator, a text validation writer and Microsoft Excel validation report writer; both are supplied with the installation.

If you want to produce a validation report in some other format, you can implement your own writer by using the following steps.

Location of sample customization code

Sample customization code for the validation writer can be found in the following location.

```
<drive>:\Program Files (x86)\Smart\SDV\2015\ CustomExamples
\Intergraph.VTL.Server.CustomValidationReportWriters\Intergraph\VTL\Ser
ver\CustomValidationReportWriters
```

NOTE Ensure that all referenced assemblies are in place before starting the customization.

Configure the validation report definition

1. Set the path of the custom validation report writer assembly. For example:

   ```
   VTLCustomReportWriterDllPath="D:\CustomDlls\CustomValidationReportWr
   iter.dll".
   ```

2. Set the following two properties on the **IVTLValidationReportDefinition** interface.

   a. **VTLValidationReportType -** This property specifies the validation report type. This type uses the VTLValidationReportTypes enum list, which contains:

   - e1TextReport - a text report enum.
   - e1ExcelReport - a Microsoft Excel report enum.
   - e1CustomReport - a custom report enum.

   NOTE To generate the report in a custom format use the e1CustomReport enum entry.

   b. **VTLCustomReportWriterDllPath** - This property specifies the path to the custom validation report writer assembly. You can use the full path or relative path to the custom assembly. For example:

   - **Full path** - `"D:\CustomDlls\CustomValidationReportWriter.dll"`
   - **Relative Path** - `"CustomValidationReportWriter.dll"`

NOTE The relative path must be based on the Smart Data Validator site Bin directory.

For example:

```
<drive>:\SmartPlant Foundation 2014 Server
Files\Web_Sites\SDVTestServer\Bin.
```

3. Use the following schema to create a new **VTLJob Definition** and **VTLValidationReportDefinition.**

   TIP This sample can also be found in the **ModelFiles** directory in the **CustomExamples** folder.

```
  <VTLValidationReportDefinition>
    <IObject UID="CustomValidationReportDefinition"
Name="CustomValidationReportDefinition" Description="Customized
Example Validation Report" />
    <IVTLValidationReportDefinition
VTLValidationReportType="e1CustomReport"
VTLCustomReportWriterDllPath=" D:\CustomDlls
\Intergraph.VTL.CustomValidationReportWriters.dll" />
  </VTLValidationReportDefinition>
```

Create a sample validation report writer

1. Create a .NET class library and add a reference to Intergraph.VTL.Server.dll located at:

   `<drive>:\Program Files (x86)\Smart\SDV\2015\Server`

2. Implement the **IVTLValidationReportWriter** interface by overriding all **MustOverride** methods.

   NOTES

   - You need to have a parameterless constructor.
   - This returns a report generated path so that the report can be attached to the validation job.

Sample custom writer implementations

The following sample customized validation writers can be found in the custom code folder location.

`<drive>:\Program Files (x86)\Smart\SDV\2015\CustomExamples`

CustomValidationReportWriter

This sample custom validation report writer class defines the steps involved in customizing the validation report writer.

The custom validation report writer class can be found in the following location:

```
<drive>:\Program Files
(x86)\Smart\SDV\2015\CustomExamples\Intergraph.VTL.Server.Custom\Interg
raph.VTL.Server.Custom\Intergraph\VTL\Server\CustomValidationReportWrit
ers\CustomValidationReportWriter.vb
```

CustomExcelReportWriter

This sample custom Excel validation report writer class defines the steps involved in customizing the existing Excel report writer by setting the properties of the Excel writer.

```
<drive>:\Program Files
(x86)\Smart\SDV\2015\CustomExamples\Intergraph.VTL.Server.Custom\Interg
raph.VTL.Server.Custom\Intergraph\VTL\Server\CustomValidationReportWrit
ers\ CustomExcelReportWriter.vb
```

NOTE You can customize the Excel report writer by setting the properties of the Excel validation report writer. For example, **SummarySheetStartingRow**, **SummarySheetStartingColumn**, **ReportTitleBGColor**, and **ColumnHeaderBGColor**.

Using the customized writers

Once you have customized the report writer, you can create a new VTLValidationReportDefinition that has the following properties:

- **VTLValidationReportType** - Custom report type
- **VTLCustomReportWriterDllPath** - Path to custom assembly

NOTES

- You can use the relative path of the custom assembly if the assembly resides in the Bin directory of the site.
- You must relate the new report definition to required job definitions, so that all corresponding jobs generate validation reports in the new custom format.

SECTION 2

Custom Target System Adapter

A target system adapter manages the communication between a Smart Data Validator system and an application that receives the exported data. The SmartPlant Foundation target system adapter uses Windows Communication Foundation (WCF), which can be implemented on a Windows platform using the .NET language.

A sample project using a custom target system adapter implementation is supplied to help you with your implementation. The supplied project hosts the WCF service inside a windows forms application, so Internet Information Services (IIS) is not required.

Target system adapter

Smart Data Validator connects to the selected target system through a target system adapter. When your target system is based on SmartPlant Foundation, you can use the target system as the basis to create your import mappings, validation rulesets, and export mappings. An indicator on the target system object indicates whether the target system supports validation and object queries, which are used to build the mappings. Other target systems can also be used just for creating the export mappings.

Custom target system

The custom sample solution installed with Smart Data Validator includes two examples of custom target systems. One custom target system is driven by a Microsoft Windows interface, and the other is a HTTP hosted service. Both of the custom target systems have examples of how to implement the following interfaces.

IVTLTargetSystemValidationService

This interface is a part of the target system adapter and allows Smart Data Validator to determine if an object already exists or if a value on an object matches the same value as found in the target system.

IVTLTargetSystemDataExportService

This part of the target system adapter loads the transformed data into the target system application as part of the export process. By the time the data has been exported from Smart Data Validator, it has already been transformed into the correct format to match the target system application. The transformed data has also been exported in the correct order, simplifying the implementation for the target system adapter for most applications. For example, a relationship between two objects is not sent on the first object, but transmitted on the second object.

IVTLTargetSystemQueryService

The target system adapter allows Smart Data Validator to query the property values of objects in the target system application for the computed columns and other similar features so that the Smart Data Validator can match the imported data properties to the target system properties.

IVTLTargetSystemSchemaQueryService

The retrieval of schema objects from the Smart Data Validator target system application allows the export mappings to be created.

SECTION 3

Custom Import Mapping Functions

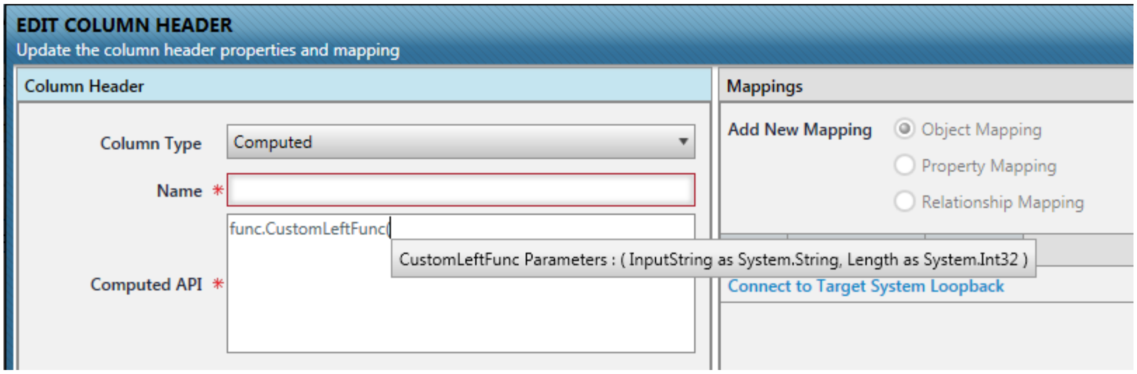
Computed columns

During the import mapping process, you can specify a **Computed** column type. This computed column type has a set of functions available to it as part of Smart Data Validator.

An example of how to add to the list of functions and implement some specific logic is provided in the **Intergraph.VTL.Server.ImportMappingFunctoids** custom assembly, which is located in the custom code folder:

```
<drive>:\Program Files (x86)\Smart\SDV\2015\CustomExamples\
```

The following example shows the computed column function available in the custom sample.



EDIT COLUMN HEADER

Update the column header properties and mapping

Column Header

Column Type: Computed

Name *

Computed API * func.CustomLeftFunc(

CustomLeftFunc Parameters : (InputString as System.String, Length as System.Int32)

Mappings

Add New Mapping ● Object Mapping ○ Property Mapping ○ Relationship Mapping

Connect to Target System Loopback

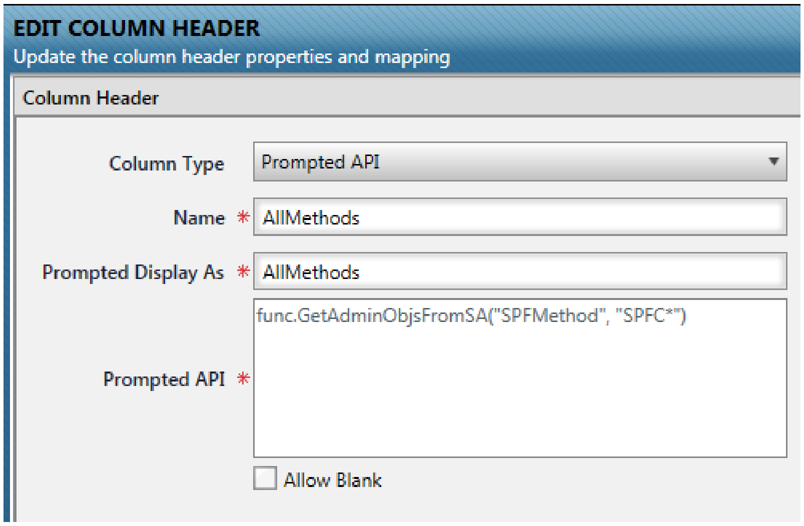
Prompted API

During the import mapping process, you can specify a **Prompted API** column type, which has a set of functions available to it as part of Smart Data Validator.

An example of how to add to the list of functions and implement some specific logic is provided in a custom assembly named **Intergraph.VTL.Server.ImportMappingFunctoids** in the custom code folder located at:

```
<drive>:\Program Files (x86)\Smart\SDV\2015\CustomExamples\
```

The following example shows using the sample **Prompted API** column available in the custom sample.



EDIT COLUMN HEADER
Update the column header properties and mapping

Column Header

Column Type: Prompted API

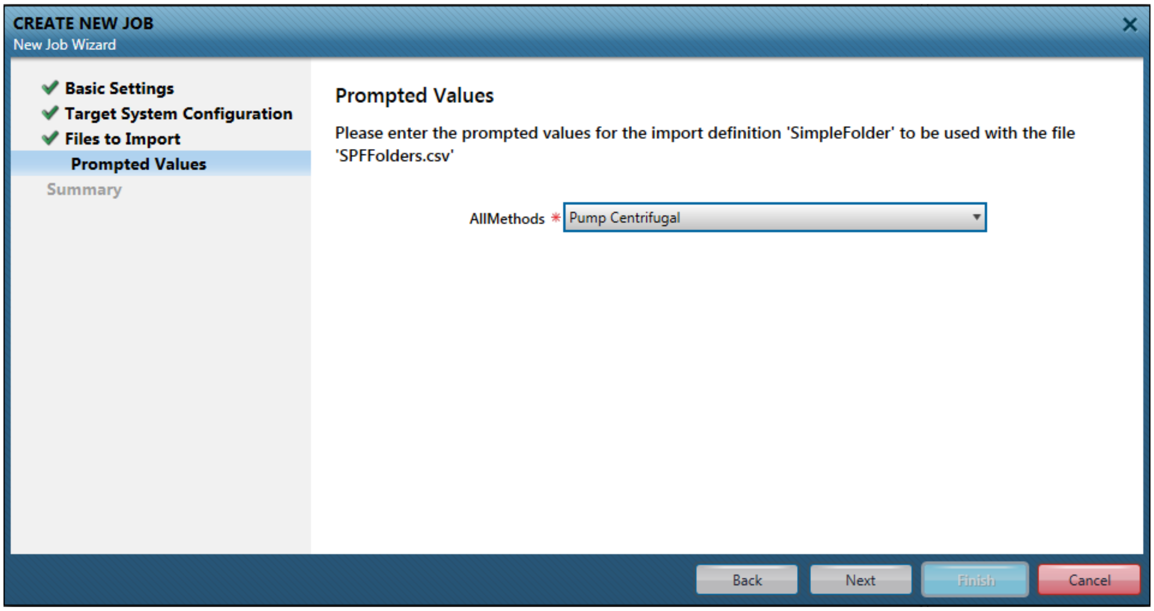
Name *: AllMethods

Prompted Display As *: AllMethods

Prompted API *: func.GetAdminObjsFromSA("SPFMethod", "SPFC*")

□ Allow Blank

The following example shows creating a new job with a **Prompted API** column.



CREATE NEW JOB
New Job Wizard

Prompted Values

Please enter the prompted values for the import definition 'SimpleFolder' to be used with the file 'SPFFolders.csv'

AllMethods *: Pump Centrifugal

Configure the custom function assembly

The custom function assembly is installed with the Smart Data Validator user interface and must be located in a custom directory found in the Smart Data Validator staging area site, such as:

```
<drive>:\SmartPlant Foundation 2014 Server
Files\Web_Sites\SDVTestServer\Bin\Custom
```

SECTION 4

Custom Validation Rules

The custom function assembly is installed with the Smart Data Validator application. You can find the custom function assembly in a custom directory in the Smart Data Validator staging area site, such as:

```
<drive>:\Program Files (x86)\Smart\SDV\2015\Server
Files\Web_Sites\SDVTestServer\Bin\Custom
```

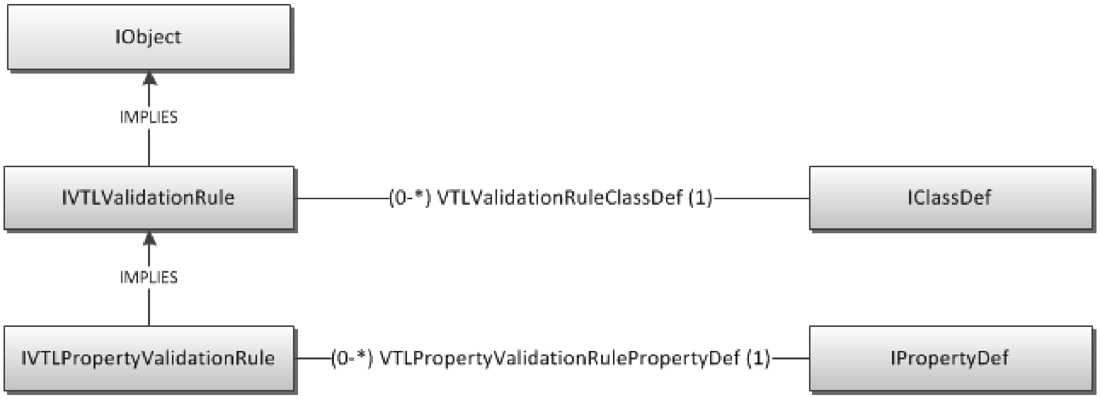
General rule schema

Every validation rule interface definition in Smart Data Validator implies the **IVTLValidationRule**.

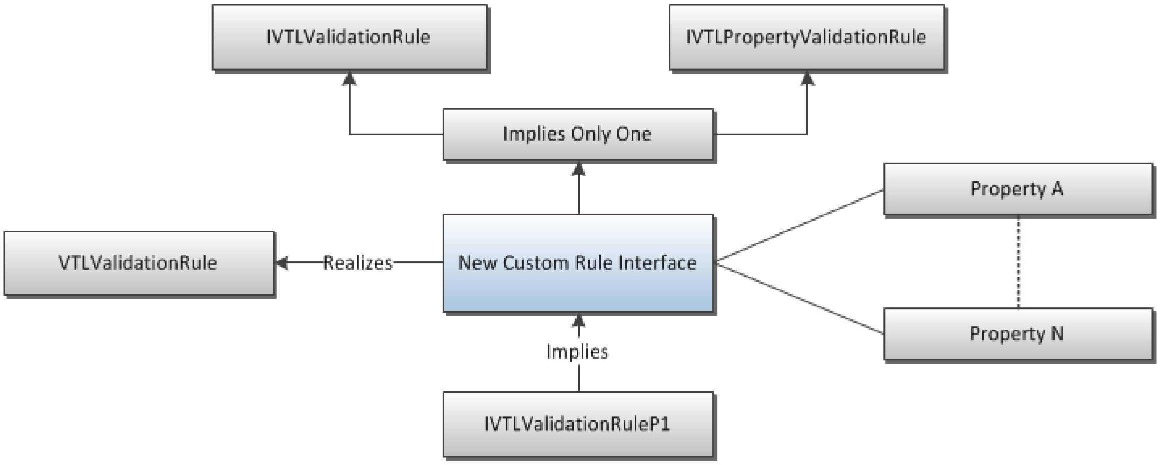
- If the rule validates a class definition, then the rule interface implies the **IVTLValidationRule**.
- If the validation rule is related to a property definition to validate a property, the rule interface implies the **IVTLPropertyValidationRule** first, which then implies the **IVTLValidationRule**.

Each new rule interface must be implied by the **IVTLValidationRule** interface definition, so that the new rule can be discovered when creating rules. Any property definition can be used on the rule, as long as it is scoped by a default SDV value. If the rule is set up correctly, the rule set section of the Smart Data Validator Administration module displays the new rule and allows dynamic entry of the property values.

The following image is an example of the schema structure for rules:


IObject
IMPLIES
IVTLValidationRule
(0-*) VTLValidationRuleClassDef (1)
IClassDef
IMPLIES
IVTLPropertyValidationRule
(0-*) VTLPropertyValidationRulePropertyDef (1)
IPropertyDef

The following image is an example of a valid rule.



Exact string rule code example

The exact string rule needs to be created in the code with the interface name exactly matching the name of the interface in the schema. The matching name allows it to be referenced in the code and allows the code that runs the rules to dynamically find the new rule. The rule interface does not need to have any properties or methods related as the interface name is the most important requirement.

The class that implements this exact string rule must realize the **IVTLValidationRuleExecution** interface, which is a generic interface, both without a type and with the rule interface.

```
Public Class VTLHasExactStringValidationRule
        Implements IVTLValidationRuleExecution,
IVTLValidationRuleExecution(Of IVTLHasExactStringValidationRule)
```

The class must be exported through the Managed Exstensibility Framework (MEF). For example,

```
<Export(GetType(IVTLValidationRuleExecution)),
PartCreationPolicy(CreationPolicy.NonShared)>
```

For a rule to report that an object or property has failed validation, itmust add a validation object to a collection and return that collection from the ExecuteValidation function. You need to realize the **IValidationObjectCreation** interface to get a class that creates validation objects.

SECTION 5

Customizing Workflows

Smart Data Validator is delivered with three out-of-the-box workflow templates. Each of these workflows includes the basic steps for importing, validating, and exporting data. Additionally, they automate the removal of database tables in the staging area and landing tables in the target system, where data is temporarily stored during export before it is loaded into the database. These workflows can be used as delivered or customized to meet your specific work processes.

- **Import Validate Export** - This basic workflow imports information into the staging area, validates the data, exports the data to the target system, and removes the temporary staging data.
- **Import Validate Implicit Delete Export** - This workflow imports, validates, and exports data, and removes the temporary staging data. It also creates an implicit deletes report that must be approved before the export operation can be completed.
- **Import Validate Export Delete Job** - This workflow imports, validates, and exports data, and removes the temporary staging data. It also removes the job from the user interface once it is complete.

Additionally, Smart Data Validator includes a number of process steps that you can use to customize the delivered workflow templates or to create custom workflow templates. You can use the workflow functionality in SmartPlant Foundation to modify the delivered workflow templates or create new ones for use in Smart Data Validator. To use a custom workflow in Smart Data Validator, use SmartPlant Foundation to relate the workflow template to the IVTLJob interface.

Workflow process steps

The following process steps are delivered in the Smart Data Validator layer, and can be used to help customize the workflows to fit your own business requirements.

- **Import** - Runs the import of the supplied data to the staging area database using a defined job definition.
- **Validation** - Runs the validation rules on the imported data and generates a validation report. The validation report must be approved before the workflow can continue.
- **Implicit Delete Report** - Indicates progress of the implicit delete report creation by SmartPlant Foundation by displaying the percentage complete. Once the implicit delete report is complete, it must be approved before the workflow can continue.
- **Implicit Delete** - Runs the implicit delete functionality on the data export from the staging area to the target system database.
- **Export** - Transfers the data to the target system and loads the data to the database.
- **JobStatusUpdate** - When implemented, provides a mechanism to change the workflow status of the job between steps. For example, see the initial step in each of the out-of-the-box workflows.

- **StageCleanUp** - Deletes all the tables that the import created in the Smart Data Validator staging database and can be included at the end of each delivered workflow. The job is still visible in the user interface after the tables have been cleaned.
- **StageCleanUpDeleteJob** - Deletes all the tables from the staging database and also deletes the job itself, so it disappears from the user interface. This step can be included at the end of a workflow template in the place of the StageCleanUp step.
- **TargetCleanUp** - When an export to SmartPlant Foundation is completed without any errors, this step removes the scheduled task and delete the landing tables used to hold the exported data before it was loaded into the database. However, if the export fails, then the task and data are left in the target system. In that case, this step can be run to remove the data.

NOTE Using the TargetCleanUp step in a retry path could cause unwanted results when a SmartPlant Foundation target system is involved. For more information on retry, see *Retry a failed implicit delete process* or *Retry a failed export process* in the *Smart Data Validator Job Management Guide.*

For more information on creating or modifying workflows in SmartPlant Foundation, see *Workflows* in the *SmartPlant Foundation Desktop Client User's Guide* and *How to Configure the Workflow Model.*

SECTION 6

Custom validation report writer

Sample customization code for a custom validation writer is supplied with Smart Data Validator, as part of the installation for the custom validation report writer. The custom validation writer sample code can be found in the following location:

```
<drive>:\Program Files (x86)\Smart\SDV\2015\
CustomExamples\Intergraph\SDV\Server\Custom\
Intergraph\SDV\Server\CustomValidationReportWriters
```

★ **IMPORTANT** Ensure that all referenced assemblies are in place before starting the customization. The Intergraph.VTL.Server.dll is a mandatory assembly for the custom validation writer, as it contains the interface that has to be implemented to customize the report writer. This assembly is found in the following location:

```
<drive>:\Program Files (x86)\Smart\SDV\2015\Server
```

Glossary

actions

An indicator of what Smart Data Validator will do with the object data in the validation and export process, such as update and delete. These kinds of operations vary, depending on whether the column header is mapped to an object, a property, or a relationship.

auto-generate

An option to automatically generate validation rule definitions, export mappings, and rules, based on actions. The validation rules are used to validate imported and exported data and the export mappings ensure that the column headers match the objects and properties found in the target system.

brownfield

An existing project or area that has constraints imposed due to prior work and contains existing data.

cardinality

A setting on a relationship definition that specifies how many instances of a relationship are valid for the objects at the end of the relationship. For example, a tag cannot exist without a primary classification relationship, and also cannot have more than one primary classification.

column headers

They are used in Smart Data Validator as the basis for mapping new data in columns to match an existing structured database.

configuration tree

A representation in a tree list, which may include plant, areas, units, and projects, that indicates the structure in which the data is stored in SmartPlant Foundation.

CSV file

A comma-separated value (CSV) file, which stores tabular data in plain-text format.

data files

A job can process multiple input files and therefore use multiple input mappings. Import mappings are defined on the job definition.

export mapping

The process that maps the imported objects, classes, and properties to the existing structure of the objects, classes, and properties found in a target system.

export process

The Export process uses a defined mapping, based on the structure of the target system, to manage the loading of the data into the final destination system.

function

A computed functional code run at import, where the output value of a function depends only on the arguments that are input to the function.

greenfield

A project or area that is completely new and does not have any constraints imposed by prior work or existing data.

implicit delete

A component process used by Smart Data Validator, where a user can decide to implicitly delete or terminate a group of objects from the target system, because they are no longer in the supplied input data submission.

import definition

A defined mapping of imported file objects, properties, and relationships from existing column headers to objects, properties, and relationships found in the staging area database during the import process.

import process

The process that manages the import of data in the data file or files to the staging area database using a defined mapping.

inverted CSV file

See *raw attribute format* (on page 21).

job

A defined object that carries information for the progress of data through a selected workflow in Smart Data Validator Job Management, such as when data is imported, validated, and exported to a target system.

job definition

A combined set of components configured for the import, validation, and export of data to a specific target system or multiple target systems.

mapping

A defined process where data is correlated from existing column headers for the objects, properties, and relationships in the imported data to the correct column headers for the objects, properties, and relationships found in another database, such as the staging database or target system. For more information, see *import definition* (on page 20) and *export mapping* (on page 19).

object weighting

A process that emphasizes the contribution of an object in a set of data to a final effect or result, thereby deciding its weight in the analysis. This affects the order in which the data is exported, helping to achieve the desired result.

query definition

A comma separated list of properties that can be entered in the **Target System Query Definition** field, which identifies the object in the target system along with the class definition. These properties can be retrieved by navigating the relationships.

raw attribute format

Where the data in a CSV file is organized in a vertical format and each row contains properties and relationships for the same object. The data in a standard CSV file is typically organized horizontally. Also referred to as an inverted CSV file.

relationship definition

A defining object that relates items together in a database which are stored in different tables.

relationship property

A property on the link interface which is part of the relationship definition. The link interface allows properties to be created on the relationship itself.

rules

A logical formula used to evaluate and verify whether the data in an imported record meets the requirements and data standards specified for the target database system.

rulesets

A combined set of rules that can be run as a set during job processing.

SDV

Smart Data Validator, the software used for importing data into the staging area, validating the data, and/or exporting data to a target system.

staging

The staging or staging area or staging database used is the first part of a combined set of processes in Smart Data Validator, where imported data is held and validated before being exported to a Smart Data Validator site.

target system

The system or database used as the destination for exported data that has been validated before export. It is also used as the basis for creating import mappings, implicit delete, and validation rules.

terminate

The action of changing an object's status to terminated without removing it from the database. Terminating objects, instead of deleting them, allows you to continue to see the history of the object after termination.

UID

Unique Identifier is used to uniquely identify an object within a system. UID definitions must be set on all objects that Smart Data Validator exports from the staging area if the unique key or query definition is left blank. The staging area UID definitions must match the UID definitions in the target system.

unique key

A set of values guaranteed to be unique for each object in a relation, and can be used to identify objects in the target system.

URL

A Uniform resource locator is a web address that contains a specific character string that references a resource.

validation process

A process that evaluates the imported data against a defined set of rules to ensure the validity of the data.

validation rule

A logical formula used to evaluate the imported data in one or more fields to determine whether it matches the existing criteria and hierarchy found in the target system database. There are two types of validation rules:

1. Rules that determine if the data is valid for the target system schema (these rules can be autogenerated.
2. Rules that determine if the data meets specific business criteria, such as naming format.

Index