

Technical Document

Tagging Guide

November 29, 2015

niagara⁴

Tagging Guide

Tridium, Inc.

3951 Westerre Parkway, Suite 350
Richmond, Virginia 23233
U.S.A.

Confidentiality

The information contained in this document is confidential information of Tridium, Inc., a Delaware corporation ("Tridium"). Such information and the software described herein, is furnished under a license agreement and may be used only in accordance with that agreement.

The information contained in this document is provided solely for use by Tridium employees, licensees, and system owners; and, except as permitted under the below copyright notice, is not to be released to, or reproduced for, anyone else.

While every effort has been made to assure the accuracy of this document, Tridium is not responsible for damages of any kind, including without limitation consequential damages, arising from the application of the information contained herein. Information and specifications published here are current as of the date of this publication and are subject to change without notice. The latest product specifications can be found by contacting our corporate headquarters, Richmond, Virginia.

Trademark notice

BACnet and ASHRAE are registered trademarks of American Society of Heating, Refrigerating and Air-Conditioning Engineers. Microsoft, Excel, Internet Explorer, Windows, Windows Vista, Windows Server, and SQL Server are registered trademarks of Microsoft Corporation. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Mozilla and Firefox are trademarks of the Mozilla Foundation. Echelon, LON, LonMark, LonTalk, and LonWorks are registered trademarks of Echelon Corporation. Tridium, JACE, Niagara Framework, NiagaraAX Framework, and Sedona Framework are registered trademarks, and Workbench, WorkPlaceAX, and AXSupervisor, are trademarks of Tridium Inc. All other product names and services mentioned in this publication that is known to be trademarks, registered trademarks, or service marks are the property of their respective owners.

Copyright and patent notice

This document may be copied by parties who are authorized to distribute Tridium products in connection with distribution of those products, subject to the contracts that authorize such distribution. It may not otherwise, in whole or in part, be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form without prior written consent from Tridium, Inc.

Copyright © 2015 Tridium, Inc. All rights reserved.

The product(s) described herein may be covered by one or more U.S. or foreign patents of Tridium.

Contents

About this guide	5
Document change log	5
Related documents	5
Chapter 1 Tagging Overview.....	7
License requirements	7
Tagging process.....	7
Chapter 2 Common Tagging Tasks	9
Creating a tagged device	9
Adding Ad Hoc tags.....	10
Removing a tag	11
Add tags to objects in the Discovered pane.....	11
Add tags in the Database pane	12
Adding a tag group to a component	13
Adding a tag to an existing tag group	14
Adding tags using Batch Editor.....	15
Editing tags in a template	16
View Implied Tags using Edit Tags dialog	17
Viewing implied tags using Spy view	18
Selecting or Exiting Tag Mode (Manager views).....	19
Exporting and importing tag dictionaries	20
Creating and exporting a new tag dictionary	20
Editing a tag dictionary exported to CSV.....	21
Importing a tag dictionary in CSV format.....	24
Chapter 3 Tagging reference	27
Online tagging vs. offline tagging	27
About the Edit Tags dialog	27
Components in the tagdictionary module.....	29
About tags.....	29
Tag Dictionaries	30
About the Tag dictionary Service	36
Plugins in the tagdictionary module	39
Tag Dictionary Manager view.....	39
Glossary	41
Index.....	43

About this guide

This guide explains to the Systems Integrator how to use the Niagara Tagging feature.

Document change log

Updates (changes and additions) to this document are listed below.

- Updated: November 29, 2015
 - Added information on changes supporting the Tag Dictionary Service functionality for Niagara 4.1. The following topics have been modified as described.
 - Creating a tagged device: edited content in steps 4 and 5. Results info explains that tags in added tag groups are replaced with an implied relation.
 - Editing tags in a template: added note to step 5.
 - About the Edit Tags dialog: deleted duplicate content in 2nd paragraph and added note at end that describes changed handling of tags in tag groups.
 - Tag Definitions: added note that about data policy in tag definitions.
 - Tag Group Definitions: added content that describes changed handling of tags in tag groups, data policies, and other instances.

The following topics have been added to this guide:

- Adding a Tag Group to a component
- Adding a tag to an existing Tag Group
- Data Policies
- Tag Group Monitor
- Initial release document: August 31, 2015

Related documents

Following documents provide information related to using tags.

- Hierarchies Guide
- Relations Guide
- Templates Guide

Chapter 1 Tagging Overview

Topics covered in this chapter

- ◆ License requirements
- ◆ Tagging process

Adding tags to your data model can streamline the process of setting up a system, especially large or “enterprise” systems. Instead of manually mapping data into the application point by point, trend by trend, systems integrators can use tags to facilitate the process. Tag information can also facilitate and improve search results and hierarchical navigation design.

If you add tags to station objects using standard Tag Dictionaries then other applications can discover station content without having to understand the naming convention used by the installer or system integrator. Typical station tagging might include things such as: networks, devices, points, control blocks, and more. You can also map all of these example objects to domain-specific semantic entities such as, buildings, systems, equipment to further indicate how they relate to each other.

License requirements

The **tags** license is required to use the **TagDictionaryService** and tag dictionaries on a station. The **Dictionary.limit** attribute limits the number of tag dictionaries available for the system. Any dictionaries added above the limit for the license will be in fault. When a dictionary is in fault, the tags in that dictionary are not available in the **Edit Tags** dialog. By default, you are limited to the first two tag dictionaries. However, the **Dictionary.limit** attribute is configurable on the license in the same manner as are device limits.

For more licensing information, see licensing topics in the Niagara 4 Platform Guide.

Tagging process

Before adding tags, make sure that you have a plan and that you have the dictionaries that you need to complete the process.

The basic process for tagging involves the following

1. Identify your purpose. Possible uses could be one or more of the following examples:
 - Enterprise structure navigation. In this case you may want to focus on using tags that include geographical.
 - Systems maintenance views. In this case you may need to use tags that include device or equipment information.
 - End user navigation. In this case you may use functionally related tags.
2. Make sure you have the dictionaries you need.

In many cases, the Niagara dictionary may be sufficient. The Niagara dictionary is in the **TagDictionaryService** folder by default in a new station. You can add the Haystack dictionary from the **haystack** palette, if needed. You can also create Ad Hoc tags or create your own custom dictionary if you need to.

NOTE: For new stations, it may be true that you need only the Niagara and Haystack Smart Tag Dictionaries. However, you can reduce or eliminate your tagging efforts by looking for Smart Tag Dictionaries developed by the Niagara community. These are likely to be the best option for stations created in NiagaraAX and migrated to Niagara 4.

3. Add tags to your components.

You can add tags one at a time or you can use Tag Groups to add multiple tags with each Add action. You can add tags during or after a discovery process and you can also use the Batch Editor to add tags.

Chapter 2 Common Tagging Tasks

Topics covered in this chapter

- ◆ Creating a tagged device
- ◆ Adding Ad Hoc tags
- ◆ Removing a tag
- ◆ Add tags to objects in the Discovered pane
- ◆ Add tags in the Database pane
- ◆ Adding a tag group to a component
- ◆ Adding a tag to an existing tag group
- ◆ Adding tags using Batch Editor
- ◆ Editing tags in a template
- ◆ View Implied Tags using Edit Tags dialog
- ◆ Viewing implied tags using Spy view
- ◆ Selecting or Exiting Tag Mode (Manager views)
- ◆ Exporting and importing tag dictionaries

The following sections include descriptions of some common ways to use tagging.

Creating a tagged device

You can add Direct Tags to a device (or other station objects) to provide additional semantic information. You may add more than one type of tag to a device, in order to support multiple hierarchical navigation schemes. You can also use Tag Groups to add a predefined collection of tags to the device in a single add action.

Prerequisites:

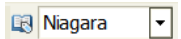
- One or more installed tag dictionaries. If necessary, add required tag dictionaries to the **TagDictionaryService**.

NOTE: If tagging offline, it is possible that no dictionaries are available. In that situation the system searches for tag dictionaries in alternate locations.

This task describes how to use the **Edit Tags** dialog box to add individual tags or tag groups from an installed tag dictionary.

Step 1 Right-click on the device that you want to tag and select **Edit Tags** from the popup menu.




Step 2 In the **Edit Tags** dialog box, select a dictionary from the option list in the top left corner



TIP: In the Search field, you can use a shortcut to designate the dictionary. Type **hs:** for Haystack, **n:** for Niagara, and similarly for other dictionaries.

The top half of the dialog box shows a list of tags available from the selected dictionary.

Step 3 Use the filter fields as needed to limit the number of tags displayed. For example:

- Type in the **Search** field  to filter by tag name. Tags are filtered immediately as you type.
- Select an option from the option list  **Show All**  to filter based on validity options (Show All, Valid Only, or Best Only).

Step 4 Add any number of tags to suit your needs (such as, n:device, hs:geoState, my:bldgRef, etc.) using either of the following methods:

- To add an individual tag from a tag dictionary, select one or more tags in the **Tag Dictionary** (upper) pane and click **Add Tag** to assign the selected tag(s) to the device
- To add a predefined collection of tags from a tag dictionary, in the **Tag Dictionary** (upper) pane in the dialog, scroll down to **Tag Groups** and select a tag group, and click **Add Tag** to assign the selected collection of tags at once.

The assigned individual tags and added tag groups are listed on the **Direct Tags** tab in the lower half of the dialog.

Step 5 Edit any tag value fields, as appropriate, and click the **Save** button to save the added tag assignments.

Starting in Niagara 4.1, when using the **Edit Tags** dialog any added TagGroups display differently than in the prior release. When adding a TagGroup to any component the added tag group displays on the **Direct Tags** tab as an Ord to the TagGroup itself. After saving the added tag assignments, when you reopen the **Edit Tags** dialog you will see the set of individual tags in that TagGroup display on the **Implied Tags** tab. The reason for this is that the Tag Group Monitor detects the presence of individual tags that are included in a tag dictionary's TagGroupDefinition and it replaces those tags with an `n:tagGroup` relation from the component to the corresponding tag dictionary's TagGroupInfo tags.

Step 6 **Optional:** For tags that have Ord type values (such as "hs:siteRef"), refer to the following steps as an example of how to add a link to your tag.

- a. Click the option list arrow located to the right of the tag value field.
- b. Select the appropriate link type from the options menu.
- c. Browse to the desired link and select it.
- d. Select the `Handle` option and click **OK**.

The device is now tagged.

Adding Ad Hoc tags

You can add Ad Hoc Tags to any station object to provide additional semantic information without using an installed tag dictionary. Ad Hoc tags are tags that you create directly from the **Edit Tags** dialog box. These tags are not found in any tag dictionary.

Ad Hoc tags are useful for development or testing purposes, allowing you to test without adding or modifying tag dictionaries and without using the tags that are already in use by active production applications. However, when applying tags that will be used by applications, best practice is to use tags from standardized tag dictionaries that are applied system-wide.

Step 1 Right-click on the component that you want to tag and select **Edit Tags** from the popup menu.

Step 2 In the **Edit Tags** dialog box, and without making any selections click the **AddTag** button.

The **Add Tag** dialog box appears.

Step 3 In the **TagId** field, enter a new tag name using the following syntax: `namespace:tagname`.

For example, `my:datalogs`. For best practices, use a consistent naming convention. Also, it is important to use a namespace that does not conflict with that of other installed tag dictionaries.

Step 4 In the **Type** field use the option list to select the tag type from the options available.

NOTE: In your ad hoc tag, do not use a namespace that is identical to an existing tag dictionary. For example DO NOT use `hs:`, `n:`, or other namespace characters that would conflict with existing tag dictionaries.

For example, for an Ad Hoc tag with the TagId `my:datalogs` you could select a type called `baja:String` to determine that the tag value be a String type of data.

Step 5 Click the **OK** button to assign the tag to your selected component.

The new tag is added and appears in the **Direct Tags** table in lower half of the dialog box.

- Step 6 Edit any tag value fields (such as String, Ord, etc.), as appropriate, and click the **Save** button to save the tag assignments.

The component is now tagged.

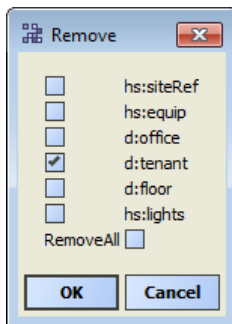
Removing a tag

You can remove direct tags from individual station objects using the **Remove** dialog box.

NOTE: This task does not apply to implied tags.

- Step 1 Right-click on the object that you want to edit and choose **Edit Tags** from the menu.
 Step 2 Click the **Remove Tag** button from the **Edit Tags** dialog box.

The Remove dialog box displays, showing a list of all the Direct tags that are assigned to the selected component.



- Step 3 Select the individual tags that you want to remove or choose **Remove All** and then click the **OK** button.

The selected tags are removed from the table listing under the Direct Tags tab in the lower half of the dialog.

NOTE: Check that the appropriate tags are now listed in the **Edit Tags** lower pane, under the Direct Tags tab. Your deletions are not complete until you click the **Save** button. Click the **Cancel** button if you want revoke the delete action.

- Step 4 Click the **Save** button to complete the task.

The tag is removed from the selected object.

Add tags to objects in the Discovered pane

Tagging is integrated into the **Station Manager** and **Point Manager** views to make it easier to tag editable stations or points in the **Discovered** pane, before adding them to the **Database** pane.

Prerequisites: Device or Point Manager view is active with Tag Mode selected. Points or devices are discovered and listed in the **Discovered** pane.

Tagging during discovery is optional but it is a convenient way to add metadata as you add points or devices. This task describes how to add tags only, it does not describe all point or device fields that need to be reviewed or edited during an add process.

- Step 1 From the **Point or Station Manager** view, in the **Tag Dictionary** pane, select the desired tag dictionary.
 Step 2 Select one or more discovered objects in the **Discovered** pane.
 Step 3 In the **Tag Dictionary** Pane, select one or more tags to add and click the **Add** button.

Step 4 In the **Add** dialog box, check that the appropriate tags are added (table in top pane) and add values to any tags that have editable fields.

Step 5 Click the **OK** button and verify that your tags appear with the desired objects in the **Database** pane.

Devices or points are added to the station with tags.

Add tags in the Database pane

Tagging is integrated into the **Station Manager** and **Point Manager** views to make it easier to tag editable stations or points that are in the **Database** pane.

Prerequisites:

- **Device Manager** or **Point Manager** view is active with **Tag Mode** selected.
- Points or devices are listed in the **Database** pane.

Tagging objects that are in the Database pane of a manager view is a convenient way to add metadata to your points or devices. This task only describes how to add tags, it does not describe point or device fields that may be edited from the manager view.

Step 1 From the **Point** or **Station Manager** view, in the **Tag Dictionary** pane, select the desired tag dictionary.

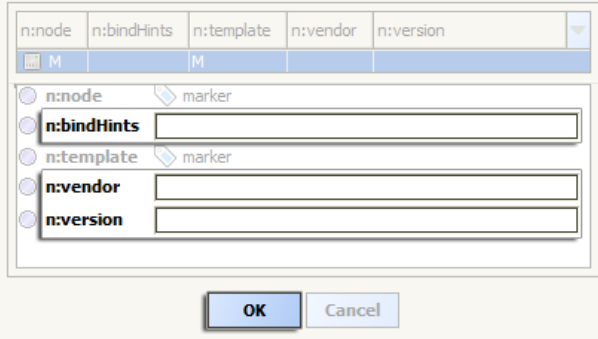
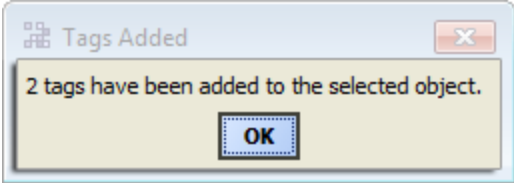
Step 2 Select one or more objects in the **Database** pane.

Step 3 In the **Tag Dictionary** Pane, select one or more tags to add and click the **TagIt** button.

Depending on the type of tag you are adding, one of the following happens:

- If one or more tags have a value field, a **Tags Edit** dialog box opens and displays all fields.
- If no tags have value fields, a **Tags Added** dialog box displays a confirmation message indicating how many tags are added.

Step 4 Depending on the type of tags you have added, do one of the following:

Option	Description
<p>Edit tag values and click OK in the Tags Edit dialog box.</p>	 <p>If this dialog box displays, then you have tag values to edit. Edit the value fields and click OK.</p>
<p>Click OK in the Tags Added dialog box.</p>	 <p>If this dialog box displays, no tag values are available. Click the OK button.</p>

Adding a tag group to a component

Adding a Tag Group lets you add a predefined collection of tags to a component in a single action. Typically tags are in a tag group because it is common for each of the tags to be assigned to the same component. The **Device Manager** and **Point Manager** views of a driver, and the **Edit Tags** dialog are the primary methods for adding a tag group to a component. This procedure describes how to use the **Edit Tags** dialog to add a tag group.

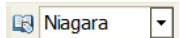
Prerequisites:

- One or more installed tag dictionaries. If necessary, add required tag dictionaries to the **TagDictionaryService**.

NOTE: If tagging offline, it is possible that no dictionaries are available. In that situation the system searches for tag dictionaries in alternate locations.

Step 1 Right-click on the component that you want to tag and select **Edit Tags** from the popup menu.

Step 2 In the **Edit Tags** dialog box, select a dictionary from the option list in the top left corner



The top half of the dialog box shows a list of individual tags available from the selected dictionary.

Step 3 Scroll down to see the list tag groups in the dictionary.

TIP: Use the Search and the Filter fields as needed to limit the number of tags displayed.

Step 4 Select the desired tag group and click **Add Tag** to assign the selected collection of tags.

The assigned tag group displays as an Ord on the **Direct Tags** tab (lower half).

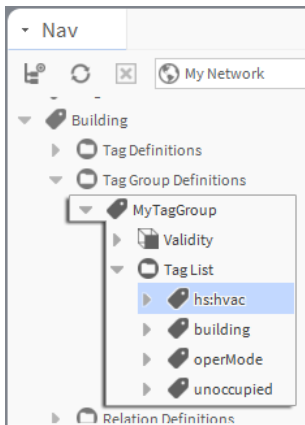
Step 5 Click the **Save** button to save the added tag assignments.

NOTE: Starting in Niagara 4.1, when using the **Edit Tags** dialog any added TagGroups display differently than in the prior release. When adding a TagGroup to any component the added tag group displays on the **Direct Tags** tab as an Ord to the TagGroup itself. After saving the added tag assignments, when you reopen the **Edit Tags** dialog you will see the set of individual tags in that TagGroup display on the **Implied Tags** tab. The reason for this is that the Tag Group Monitor detects the presence of individual tags that are included in a tag dictionary's TagGroupDefinition and it replaces those tags with an `n:tagGroup` relation from the component to the corresponding tag dictionary's TagGroupInfo tags.

Once you save the added tag assignments, the set of tags in the tag group are implied tags on the component.

Adding a tag to an existing tag group

Starting in Niagara 4.1, you can add a tag from a different tag dictionary to an existing tag group. Optionally, you can add a tag from the `tagdictionary` palette. Shown in the following image, the `hs:hvac` tag (copied from the Haystack tagdictionary) is added to MyTagGroup in the Building tagdictionary.



Prerequisites:

- At least two tag dictionaries installed, one of which contains a TagGroup

- Step 1 In the Nav tree, expand the tag dictionary to the TagGroup that you wish to edit.
- Step 2 Expand the second tag dictionary to select a Marker tag to you wish to add to the TagGroup in the first tag dictionary.
- Step 3 Drag (or right-click and Copy) the selected tag, and drop it (or right-click and Paste it) on the Tag-List folder of the Tag Group Definition that you are editing.

NOTE: An alternative is to add a **Marker** tag from the `tagdictionary` palette.

- Step 4 In the **Name** dialog, enter the tag's desired name as a fully qualified tag name including the namespace with the colon separator. For example: `hs:hvac`.

NOTE: There is no verification that the tag name entered is actually defined in a tag dictionary. If it is not defined in a tag dictionary, the added tag is an "ad hoc" tag. While you can certainly use ad hoc tags, the recommended tagging best practice is to use tags that are contained in a standardized tag dictionary that is applied system-wide.

Changes to the tag group are saved automatically.

This namespace overrides the application of the parent dictionary's defined namespace. The added tag automatically becomes an `n:tagGroup` relation (an implied tag) from the component to the corresponding tagdictionary's TagGroupInfo.

Adding tags using Batch Editor

You can add Direct Tags to large numbers of objects using the Program Service, **Batch Editor** view. Use the **Batch Editor** to locate objects that need tagging and use the **Add Tags** button in the **Batch Editor** view to add tags.

Prerequisites:

- One or more installed tag dictionaries. If necessary, add required tag dictionaries to the **TagDictionaryService**.

NOTE: If tagging offline, it is possible that no dictionaries are available. In that situation the system searches for tag dictionaries in alternate locations.

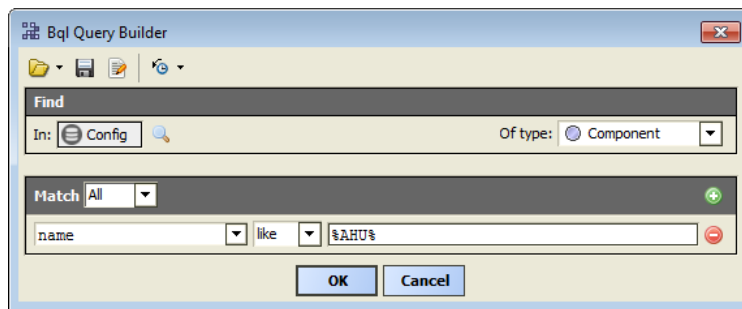
This task describes how to use the **Edit Tags** dialog box to add individual tags or tag groups from a dictionary that is installed in the station Services folder.

NOTE: If you are using tags to support multiple hierarchical navigation schemes, you may add more than one type of tag to a component. You can use Tag Groups for adding multiple tags in a single add action.

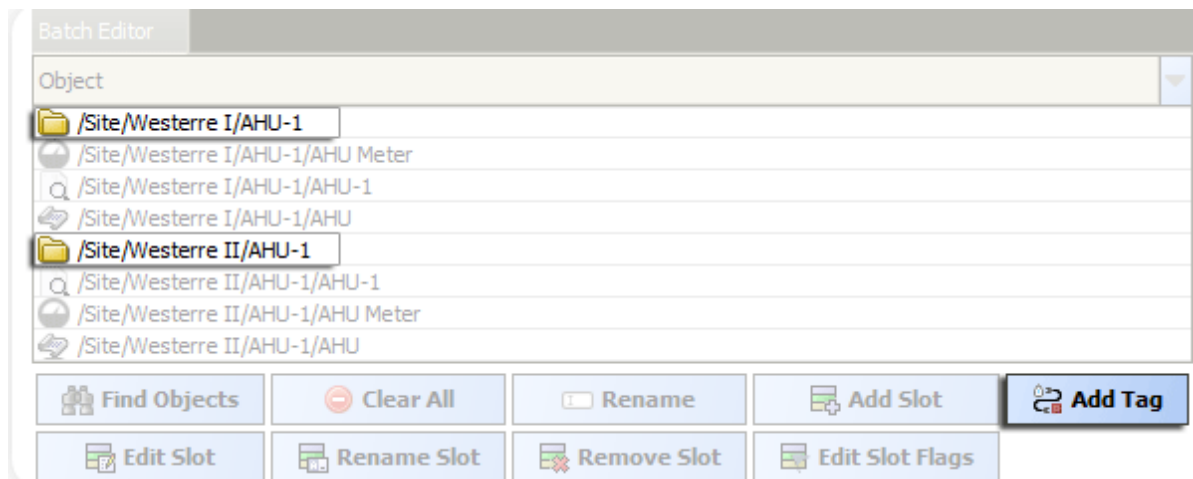
Step 1 In the Station Nav tree, expand the **Station→Config→Services** nodes and double click on the Program Services node.

Step 2 In the **Batch Editor** view, click the **Find Objects** button and use the **Bql Query Builder** to produce a list of objects that you want to tag.

For example, under the Config node you can search for all Component types that have AHU in their name (using parameters similar to the example in the image below).



The search should produce a list of matching components similar to the example in the following image.

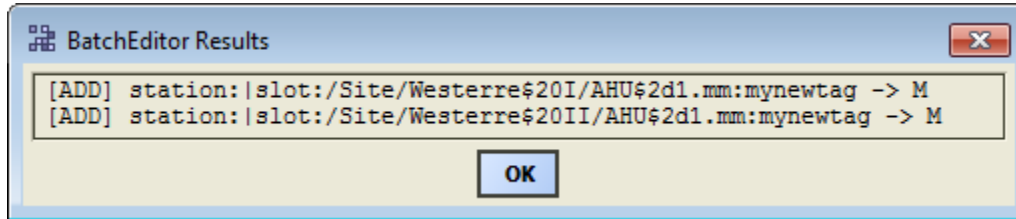


Step 3 Select and remove the unwanted components in the table and then click the **Add Tag** button.

Step 4 Select one or more tags from the **Tag Dictionary** (top) pane (or you may choose to use the **Add Tag** button from this dialog box to apply custom tags) and then click the **AddTag** button to assign the selected tags to all components.

NOTE: Select a Tag Group, if appropriate, to add several tags at once.

The selected tags are added and appear in the **Direct Tags** table in lower half of the dialog box. Finally, the **BatchEditor Results** dialog opens with all tag actions listed.



The components are tagged.

Editing tags in a template

You can edit an existing template to add additional direct tags to the objects in it, or to remove or modify the existing tags. These changes are made on the template **Configuration** tab

Prerequisites:

- An existing template
- One or more installed tag dictionaries

This task describes using the **Template** sidebar to access an existing template and invoke the **Edit Tags** dialog from the template **Configuration** tab.

Step 1 To locate a template, open the **Template** side bar by clicking **Window→Side Bars→Template**



Step 2 In the **Template** side bar, click the pull down menu and select either: **templates** folder or **modules** folder.

NOTE: To see templates stored in a template module, select the **modules** folder and expand the desired module.

Step 3 Double-click on the desired template.

The **Template** view appears displaying the template configuration tabs with the **Template Info** tab selected.

NOTE: A template stored in a module cannot be edited. When you open it, you will see "ReadOnly" in the top left corner of the **Template** view. In order to make changes you must first click **Save As** and save it as a new template in the **templates** folder.

Step 4 Click the **Configuration** tab in the **Template** view, right-click the object you want to change in the left pane, and select **Edit Tags**.

The **Edit Tags** dialog displays.

Step 5 Proceed to add tags, remove tags, or modify values of existing tags and click **Save** to close the **Edit Tags** dialog.

NOTE: Starting in 4.1, when using the **Edit Tags** dialog within the **Template Editor** view, TagGroup tags are assigned to a component as individual direct tags in the template. Upon template deployment, the Tag Group Monitor in the Tag Dictionary Service is notified and the individual tags (of the tag group) are removed from the component and replaced with an `n:tagGroup` relation. The relation, of course, implies the same set of tags.

Step 6 When finished, click **Save** to save your changes to the template.

- Or, click **Save As** to create a new variation of the template with a different filename (leaving the original template unchanged).

NOTE: For more details on using the Template view Configuration tab, refer to the *Template Guide* sections "Creating a template" and "Template reference".

View Implied Tags using Edit Tags dialog

Viewing Implied tags can be useful when designing a custom tag dictionary, to confirm that certain objects are getting the desired tags, or when designing a NEQL query for a hierarchy definition or search. Implied Tags do not appear in an object Property Sheet or other typical views. One way that to view these tags is on the Implied Tags tab in the **Edit Dialog** box.

Prerequisites:

- One or more installed tag dictionaries. If necessary, add required tag dictionaries to the **TagDictionaryService**.

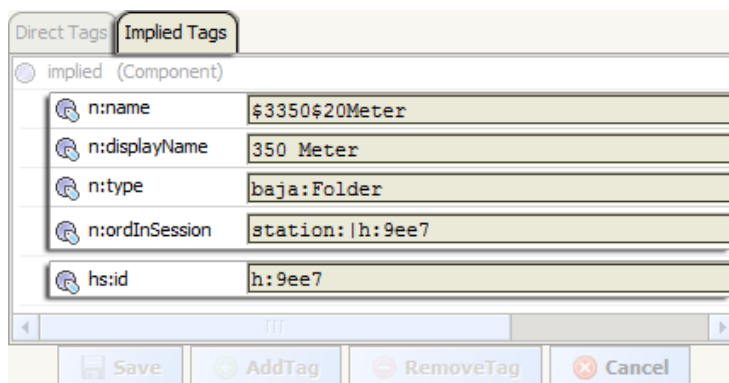
NOTE: If tagging offline, it is possible that no dictionaries are available. In that situation the system searches for tag dictionaries in alternate locations.

Implied tags are automatically assigned to objects by **Smart Tag Rules** in the installed Tag Dictionaries.

Step 1 Right-click the object whose tags you want to examine and choose **Edit Tags** from the popup menu.

Step 2 Select the **Implied Tags** tab in the lower pane of the dialog to view the **Implied Tags**.

For example, in the following image, you can see five tags that are implied based on the dictionary rules for a Component object:



The first four tags are implied from Niagara tag dictionary rules:

- `n:name`
- `n:displayName`
- `n:type`
- `n:ordInSession`

The final tag is implied based on Haystack tag dictionary rules:

- `hs:id`

Viewing implied tags using Spy view

Implied tags and implied relations are automatically assigned to objects by rules in the installed Smart Tag Dictionaries. The implied tags and implied relations do not appear in the Property Sheet view or other more commonly used views. Spy View shows all of the direct and implied tags and relations on an object as well as other detailed data. Although intended to be used for diagnostic purposes, you can use Spy View to identify implied tags and/or relations already assigned to a component. This can be useful when developing hierarchies. Once identified, you can then create queries for those tags/relations in your hierarchy definition.

Prerequisites:

- Connection to your station
- One or more installed tag dictionaries. If necessary, add required tag dictionaries to the **TagDictionaryService**.

NOTE: If tagging offline, it is possible that no dictionaries are available. In that situation the system searches for tag dictionaries in alternate locations.

This procedure describes the how to open the Spy View on a station component to see its implied tags:

NOTE: Invoking the **Edit Tags** dialog is another method for viewing the direct and implied tags assigned to a component.

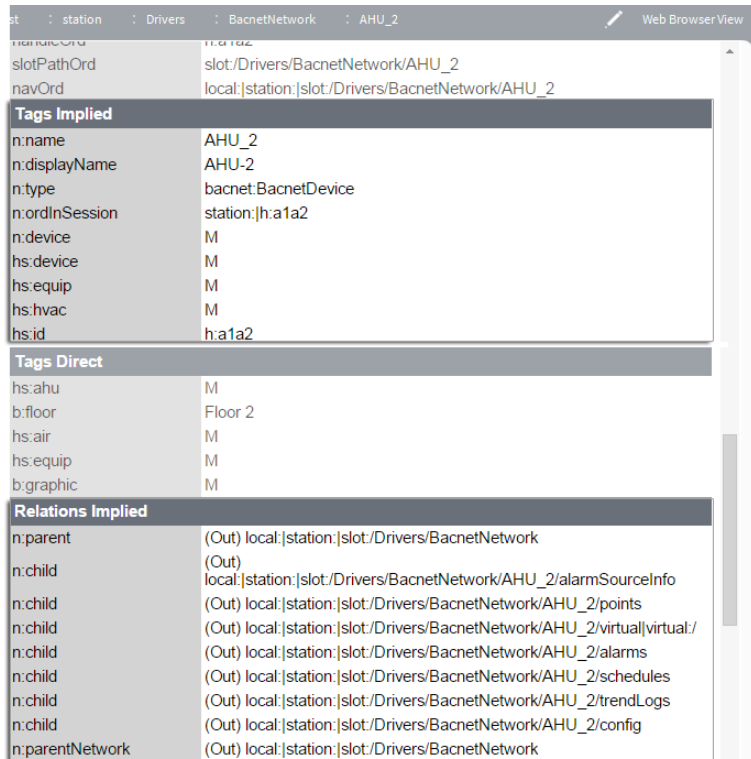
Step 1 In the Workbench Nav tree, right-click on the component of interest and click **Views→Spy Remote** from the popup menu.

Spy information displays in the **Web Browser View**.

Step 2 Scroll down the until you see the **Tags Implied** heading.

The implied tags assigned to the selected component are listed. Scroll up or down to view all of the tags and relations assigned to the component.

The Spy view pictured here lists the different types of tags and relations assigned to the `AHU_2` component.




Selecting or Exiting Tag Mode (Manager views)

Station Manager and Point Manager views have a **Tag Mode** available for adding tags to devices or points as they are added.

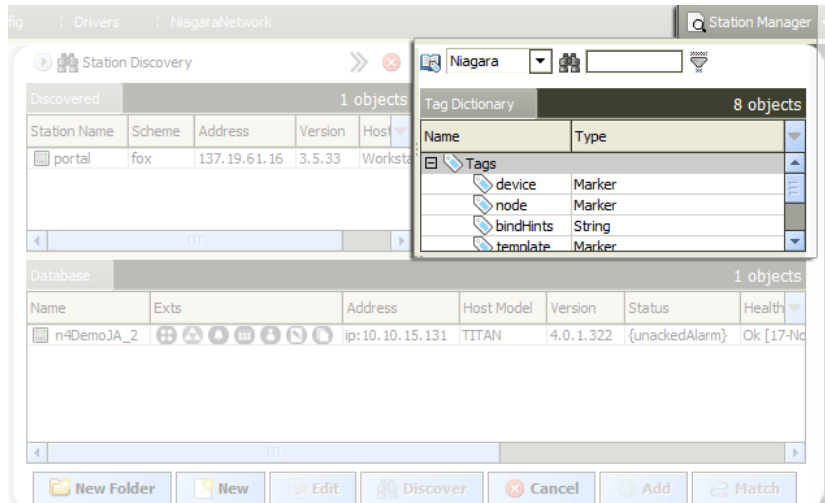
Prerequisites:

- **Tag Mode** is only available in the **Station Manager** or **Point Manager** views.

Tagging is integrated into the driver manager views to help you add tags when devices or points are discovered and added. You can select and exit **Tag Mode** using either the **Manager** menu or the  **Tag** icon in the **Toolbar**.

Step 1 While in the Station Manager or Point Manager view, click **Manager**→**Tag Mode** from the Workbench main menu to select or exit Tag Mode.

When selected, Tag Mode appears as a single pane across the top or as a second pane in the upper pane depending on whether or not you also have Learn Mode selected. The following image shows Tag Mode and Learn Mode selected simultaneously.



Exporting and importing tag dictionaries

The **Tag Dictionary Manager** view provides a method to import and export tag dictionaries (or smart tag dictionaries) in a standard format, CSV file format, compatible with Excel (or other CSV-compatible spreadsheet software). This facilitates creating custom tag dictionaries which you then import to your station. Working in the exported CSV file, you can easily edit the correctly structured tag dictionary, populating it with your custom tag definitions, etc. When finished, simply save the revised CSV file and import it using the **Tag Dictionary Manager** view.

It may suit your purposes to create a custom tag dictionary (or dictionaries) for a specific customer, for an OEM, or for a specific application. You may use a custom tag dictionary as you would other tag dictionaries, to apply tags to objects, create hierarchy definitions, as well as search the station for tagged objects tags

NOTE: By default, the license for the Tag Dictionary Service limits the number of tag dictionaries available for the system to the first two. Any dictionaries added above the limit for the license will be in fault and unusable. However, the `Dictionary.limit` attribute on the license is configurable in the same manner as are device limits.

Creating and exporting a new tag dictionary

Create a new custom tag dictionary and export it to CSV file format. Using this method of creating a new tag dictionary results in a correctly structured, "empty" tag dictionary, which you can then export for editing.

Prerequisites:

- tag license
- TagDictionaryService installed
- Open station connection (online or offline)

Step 1 In the Nav tree, double-click the TagDictionaryService to open the **Tag Dictionary Manager** view (or use the right-click menu to open this view).

Step 2 Click **New** and in the **New** dialog, select one of the following options and click **OK**.

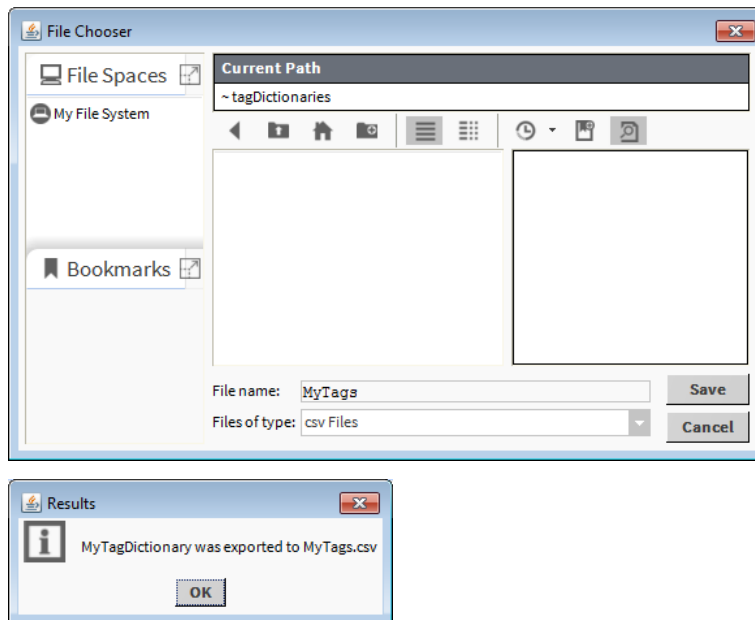
- Tag Dictionary ("simple" tag dictionary with no implied tags)
- Smart Tag Dictionary (supports implied tags)

Step 3 In the 2nd **New** dialog, in the **Name** field enter the dictionary name and click **OK**.

Your new custom tag dictionary is listed in the database table in the and also is visible under the TagDictionaryService node in the Nav tree.

Step 4 In the **Tag Dictionary Manager** view, select your new custom tag dictionary and click **Export**.

Step 5 In the **File Chooser** dialog, select a location to save the file, enter the desired file name (as shown), and click **Save**.



The exported structured tag dictionary is "empty" at this point, you can edit the file, as well as use it as a template which you use to develop additional tag dictionaries.

NOTE: You can also export the Niagara or Haystack dictionaries to use as examples.

Editing a tag dictionary exported to CSV

You can open an exported tag dictionary (CSV format) in Microsoft Excel or other CSV-compatible spreadsheet software, or even a text editor. The file is structured correctly, ready for you to enter a namespace and add tag definitions and other types of definitions, as needed.

Prerequisites:

- Tag dictionary exported to CSV format
- CSV-compatible spreadsheet software

Step 1 Open the exported CSV file (which should resemble the one shown here).

	A	B	C	D	E	F	G	H	I	J	K	L
1	namespace											
2							Validity Rules					
3	Section	Rule Name	Group Name	Tag name	Type	Smart Type	hasTags	hasAncestor	isType	hasRelation	hasRelationFilter	Units
4	TagDefinitions											
5	#			name	type	module: class	opt	opt	opt	opt	opt	opt
6	#				AbsTime							
7	#				Boolean							
8	#				Marker							
9	#				Double							
10	#				Float							
11	#				Integer							
12	#				Long							
13	#				Ord							
14	#				RelTime							
15	#				String							
16	#				TimeZone							
17	#				Unit							
18												
19	TagGroupDefinitions											
20	#		groupName				opt	opt	opt	opt	opt	
21												
22	#(tags)			name								
23												
24	RelationDefinitions											
25	#			name	Relation							
26												
27												
28												

Step 2 In cell B1, enter a **Namespace** for this tag dictionary (typically only a few characters, for example: MyTags).

NOTE: Avoid using a namespace that conflicts with that of other installed tag dictionaries, such as "n" (NiagaraTagDictionary) or "hs" (HaystackTagDictionary).

Step 3 Starting under the **Tag Definition** section, in **Row 18**, enter the first of your tagDefinition entries (one per row), be sure to note the following information:

Row	Description
Row 4	Tag Definition section: define the tags for your dictionary in this area.
#	Rows that begin with # are comment lines which show examples or explanatory comments which may be helpful to retain in the file.

Row	Description
Row 5	Row 5 is an example of the pattern to use to define a tagDefinition entry. You must define tag name and tag type.
Row 6–17	<p>Row 6–17 show valid tag types that can be used.</p> <p>Validation Rules columns: Validation rules are optional. These columns are used to define NEQL query predicates that will be used to suggest where the tag can be applied. For a particular tag definition entry, if more than one of these columns have values they will be wrapped in a tagdictionary:And function.</p> <ul style="list-style-type: none"> • hasTag: this tag may be applied if the target component also matches this NEQL tag query. • hasAncestor: This tag may be applied if the target component has an ancestor that matches this neql tag query. • isType: This tag may be applied to the target component is one of these types. Value must be entered in the "module:ClassName" form. If more that one type is entered separated by a space, It will be treated as an "or" function. Example: driver:Device driver:PointFolder: The tag is valid on a BDevice or a BPointFolder. • hasRelation: This tag may be applied if the target component has has this relation and has the tags that are listed in the hasRelationFilter (3.c.v) column. hasRelationFilter: This is a tag filter used with hasRelation (3.c.iv) validation check. <p>Units: Units are optional. This can be used to define a measurement unit to be used for a tag that has a value. The value entered is used to as the unitName argument in the BUnit.getUnit(String unitName). Example: "square foot" for a tag whose value is an area</p>

- Step 4 Under the TagGroupDefinitions section (optional), add a row for each TagGroup, defining a Group-Name that will be used to represent this collection of tags.
- Add one or more tag rows under the TagGroup one row for each tag in the group.
- NOTE:** Tags included in a TagGroup must also be defined in the TagDefinitions section.
- Step 5 Under the RelationDefinitions section (optional), add a row for each relation defining a Relation-Name and enter `Relation` in the Type column.
- Step 6 Under the RuleDefinitions section (SmartTagDictionaries only) define the rules for implied tags and relations.
- Add a row for each TagRule, entering a RuleName and one or more validation rule column values. See information on Validation Rules listed under Step 3.
 - Under the tag rule row, add a row for each implied tag for this rule entering a name, type and a smart type. The smart type should be in the module:class format.
 - If there are any implied tag groups for this rule add a row for each with the GroupName entered in the groupName column.
 - If there are any implied relations for this tag rule, add a row for each with the RelationName in the name column and enter `Relation` in the type column.

Your edited tag dictionary is complete and ready to import. An example of an edited tag dictionary in CSV file format, shown here.

	A	B	C	D	E	F	G	H	I	J	K	L
1	namespace	my										
2							Validity Rules					
3	Section	Rule Name	Group Name	Tag name	Type	Smart Type	hasTags	hasAncestor	isType	hasRelation	hasRelationFilter	Units
4	TagDefinitions											
5	#			name	type		opt	opt	opt	opt	opt	opt
6	#				Abstime							
7	#				Boolean							
8	#				Marker							
9	#				Double							
10	#				Float							
11	#				Integer							
12	#				Long							
13	#				Ord							
14	#				RelTime							
15	#				String							
16	#				TimeZone							
17	#				Unit							
18				building	Marker							
19				area	Double		my:building					square foot
20				outside	Marker							
21				air	Marker							
22				temp	Marker							fahrenheit
23	TagGroupDefinitions											
24	#		groupName				opt	opt	opt	opt	opt	
25	#(tags)			name								
26			oaTemp							control:NumericPoint		
27				outside								
28				air								
29				temp								
30	RelationDefinitions											
31	#			name	Relation							
32				buildingRef	Relation							

- ❶ Namespace = my
- ❷ Tag Definitions — notice tagDefinitions, area and temp are configured with Validity Rules
- ❸ Tag Group Definitions — notice oaTemp tagGroup is configured with a Validity Rule and the group contains three tagDefinitions. (outside, air, and temp).
- ❹ Relation Definitions — defines one Relation, buildingRef

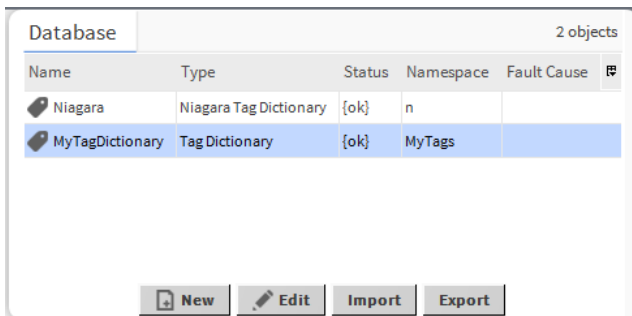
Importing a tag dictionary in CSV format

Prerequisites:

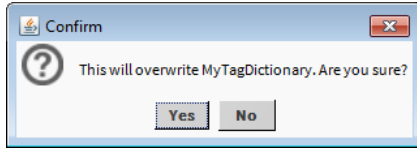
- Tag dictionary in CSV format in your Workbench user home

Step 1 Open the **Tag Dictionary Manager** view of the TagDictionaryService.

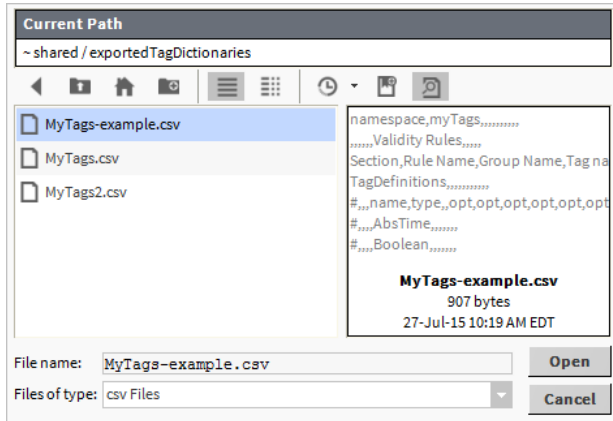
Step 2 Select the tag dictionary to update and click **Import**.



Step 3 In the **Confirm** dialog (alerting you that the Import action will overwrite the selected tag dictionary) click **Yes** to continue.



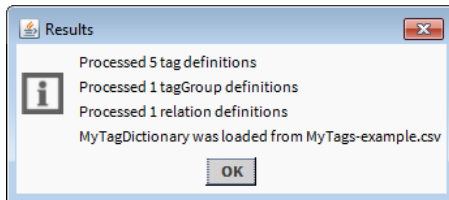
Step 4 In the **File Chooser** dialog, locate and select the CSV file to import, and click **Open**.



NOTE:

By default, the function prompts for a CSV file. This behavior can be modified programmatically.

Step 5 Click **OK** in the **Results** dialog (notifying you that the CSV file imported successfully), as shown.



NOTE: In the event that an error is detected in the CSV file, an **Error** dialog displays indicating the error and its location by row or line number.

Step 6 In the Nav tree, expand the TagDictionaryService node and double-click (or right-click) on the imported/updated tag dictionary to open a **Property Sheet** view. Then review its properties and verify your changes.

Property Sheet

MyTagDictionary (Tag Dictionary)

Status	{ok}
Fault Cause	
Namespace	myTags
Enabled	<input checked="" type="checkbox"/> true
Frozen	<input type="checkbox"/> false

Tag Definitions Tag Info List

- building Marker
- area Double
- outside Marker
- air Marker
- temp Marker

Tag Group Definitions Tag Group Info List

- oaTemp Tag Group Info

Relation Definitions Relation Info List

- buildingRef Relation Info

Refresh Save

Chapter 3 Tagging reference

Topics covered in this chapter

- ◆ Online tagging vs. offline tagging
- ◆ About the Edit Tags dialog
- ◆ Components in the tagdictionary module
- ◆ Plugins in the tagdictionary module

Tagging is a form of semantic modeling that assigns information (one or more tags) to objects. The tag information can help integrators and users significantly when searching for objects, designing system structures or navigating hierarchies.

Tagging can identify a device and indicate where it is physically located. By identifying and locating devices, tags provide a context for the device that can be used in many different ways. When you use tags, you can reduce or eliminate the requirement to manually map objects directly to a desired application.

Online tagging vs. offline tagging

There are three separate scenarios in which you apply tags:

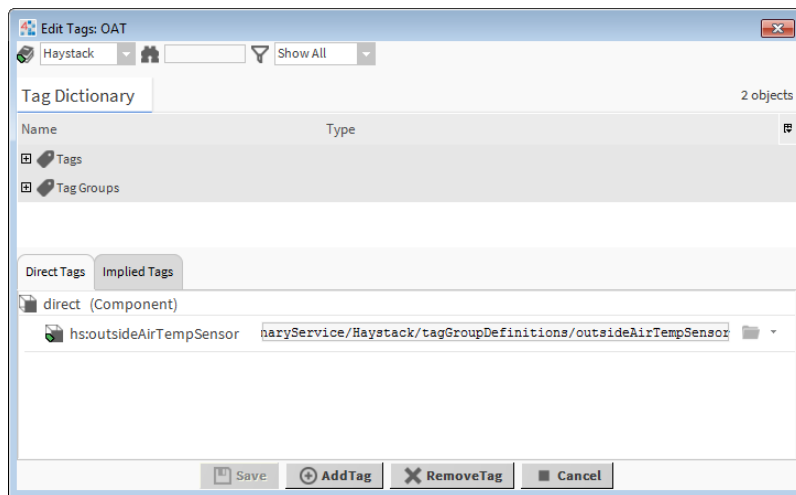
- **Online tagging** — where the installed tag dictionaries in **TagDictionaryService** take effect.
- **Offline tagging in a station with tag dictionaries** — where the installed tag dictionaries in the **TagDictionaryService** take effect. You will also see implied tags in the **Edit** dialog.
- **Offline tagging in a station when no dictionaries are found** — in this case the system searches for tag dictionaries in the following locations:
 - all palettes of installed tag dictionary modules
 - in the `user-home/tagDictionary` folder (where custom tag dictionaries are stored)
 - and also searches for Implied tags

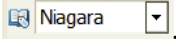
About the Edit Tags dialog

The **Edit Tags** dialog (as well as the station and point manager views) are the primary methods for adding a Tag or a Tag Group to a component. The dialog lets you add individual tags or tag groups to the object, as well as remove them from the object. The lower half of the dialog provides tabs to view the Direct and Implied tags assigned to the object.

Invoke the **Edit Tags** dialog by right-clicking an object and selecting **Edit Tags**.

Figure 1 Edit Tags dialog



In the **Edit Tags** dialog box, select a dictionary from the option list in the top left corner .

TIP: You can use this shortcut to select a dictionary. In the **Search** field type `hs :` for Haystack, `n :` for Niagara, or enter the namespace for another dictionary.

The top half of the dialog box shows a list of tags available from the selected dictionary. Once a tag is assigned to the active object, the tag icon appears dimmed.

To facilitate making selections, the dialog includes filters which help by narrowing the list of tags from which you can choose. This is most useful when selecting from a dictionary containing a huge number of tags, such as the Haystack Tag Dictionary.

Type in the **Search** field  to filter by tag name. Tags are filtered immediately as you type.

- If the list has only a single item, then it is selected by default.
- If a tag name is a subset of another tag, adding a space selects the shorter tag by name.

For example, if you have both "chiller" and "chillerPlant" tags, typing "chiller" shows both tags. Adding a space after "chiller" filters out "chillerPlant" and shows the "chiller" tag only.
- Entering a colon ":" filters for the tag dictionary that has the prefixed namespace.

For example, if you enter "hs:temp" you select the "hs" (Haystack) dictionary and the "temp" tag.

You can also select an option from the option list  to filter based on validity options.

- **Show All:** no filtering applied when this option is selected.
- **Valid Only:** shows just the tags that are valid based on rules defined in the tag dictionary.
- **Best Only:** filters tags in appropriate manager views based on the identity of the component; for example, whether it is a point or device.

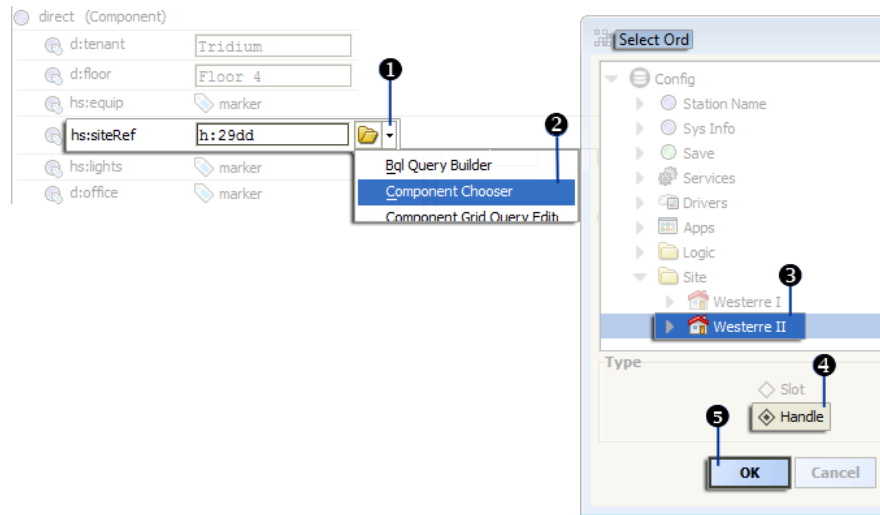
Methods for adding tags include the following:

- Add an individual tag from a dictionary
- Add a Tag Group (predefined collection of tags) from a dictionary
- Add a unique Ad Hoc tag which you create

After clicking **Add Tags**, the selected tags are added and appear in the **Direct Tags** table in lower half of the dialog box.

After editing any tag value fields as needed, click the **Save** button to save the tag assignments.

For tags that have Ord type values (such as "hs:siteRef"), refer to the following image and steps as an example of how to add a link to your tag.



1. Click the option list arrow located to the right of the tag value field.
2. Select the appropriate link type from the options menu.
3. Browse to the desired link and select it.
4. Select the `Handle` option.
5. Click the `OK` button.

NOTE: Starting in Niagara 4.1, when using the **Edit Tags** dialog any added TagGroups display differently than in the prior release. When adding a TagGroup to any component the added tag group displays on the **Direct Tags** tab as an Ord to the TagGroup itself. After saving the added tag assignments, when you reopen the **Edit Tags** dialog you will see the set of individual tags in that TagGroup display on the **Implied Tags** tab. The reason for this is that the Tag Group Monitor detects the presence of individual tags that are included in a tag dictionary's TagGroupDefinition and it replaces those tags with an `n:tagGroup` relation from the component to the corresponding tag dictionary's TagGroupInfo tags.

Components in the tagdictionary module

Component include services, folders and other model building blocks associated with a module. They may be dragged and dropped onto a Property or Wire sheet from a palette.

Components in the `tagdictionary` module are described in the following sections.

About tags

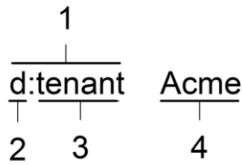
Tags provide additional information to objects in order to make the objects more accessible and flexible for search and system design. Tags also facilitate the design and use of hierarchical organization in a station user interface, whether you are working with an Enterprise Supervisor station or a single controller station.

NOTE: The tags available for use are defined in the tag dictionaries installed on your station.

Tag structure

A tag contains different parts that, together, make the tag useful as additional information on objects in a station. The following diagram shows the four basic parts of a tag.

Figure 2 Parts of a Tag



The following table provides definitions of the different parts of a tag:

Item	Tag Element	Description
1	Tag Id	The Tag Id is comprised of a dictionary and name, generally displayed as two pieces of text separated by a colon (:), as shown in the following example: <code>dictionaryNamespace:name</code> .
2	Tag Dictionary	The dictionary string is used to link or assign a tag to a particular "namespace" (tag dictionary). This is typically a very short string of only a few characters. NOTE: If the dictionary is not defined (empty string), the Id is displayed with just the name.
3	Tag Name	The name string provides the semantic information and is often paired with the Tag Value.
4	Tag Value	A string value assigned to the tag for more information, for example: building name, device name, location, or other.

Types of tags

The following table describes types of tags that may be used on the system:

Tag type	Description
Direct tags	Direct tags are tags that you add intentionally to a component using an installed tag dictionary or an Ad Hoc tag. In its simplest form, a Tag on a component is a component "property", with a non-component value and a metaData flag set. The property name is a string form of the Tag Id. In the Edit Tags dialog box, Direct Tags are listed under the Direct Tags tab.
Implied tags	Implied tags are tags that are not directly stored in the component, but are "implied" by tag rules that are defined in installed Smart Tag Dictionaries. These tags are typically the remapping of existing component properties to the semantic naming convention defined in a Tag Dictionary. In the Edit Tags dialog box, Implied Tags are listed under the Implied Tags tab.
Ad Hoc tags	An Ad Hoc tag, also a direct tag, is one that you create in the Add Tag dialog box just before adding it to a component. Ad Hoc tags are not included in any Tag Dictionary.

Tag Dictionaries

The **TagDictionary** component, found in the **tagdictionary** palette, is the container for a collection of tag definitions, tag group definitions, and relation definitions. The tags in a tag dictionary may be associated with devices, components, and points. Typically, these associations are established when the device is discovered, registered, and fully subscribed but tags can be added to an object at any time. Tags also provide a vocabulary for searching.

About tag dictionaries

NOTE: The tags license is required in order to use the TagDictionaryService and tag dictionaries on a station.

A tag dictionary is used to define:

- The collection of standardized tags with an Id.name, that has semantic meaning for a given domain or namespace. The dictionary also defines tag default values and any validation rules for applying tags.
- The collection of standardized Relation Id's with semantic meaning for a given domain or namespace.
- The collection of standardized grouping of tags (tag groups) that have semantic meaning for a given domain or namespace. The dictionary also defines any validation rules for applying these tag groups.

Types of tag dictionaries

The Niagara dictionary and Haystack dictionary are provided by default, other dictionary types include: Custom tag dictionary and Smart tag dictionary.

Dictionary type	Description
Niagara tag dictionary	The Niagara dictionary is indicated by the <code>n</code> character, followed by a colon character (:). The Niagara dictionary, is a type of Smart Tag dictionary, therefore it applies Implied Tags and Implied Relations to components and links. This allows queries to find these components based on type, linkage, hierarchy or combinations of these. The Niagara tag dictionary is included by default in all stations created using the New Station Wizard .
Haystack tag dictionary	The Haystack dictionary is indicated by the <code>hs</code> character, followed by a colon character (:). The Haystack dictionary is a result of the work of the Open Source Initiative (OSI) hosted on the website http://project-haystack.org .
Custom tag dictionary	You can create and add as many dictionaries as you like to a station Tag Dictionary Service. For example, you may create one or more custom dictionaries for a specific customer, for an OEM, or for a specific application.
Smart tag dictionary	A Smart Tag Dictionary is a tag dictionary containing a list of Tag Rules that determine implied tags and implied relations for each and every object, applying the tags and relations to objects automatically. Technically, these smart tags (Implied Tags and Implied Relations) are never added to the station, and the station size is not increased as a consequence.

Tag dictionary composition

A tag dictionary is composed of the following:

- A unique namespace, normally 1- or 2-characters, such as "n" for Niagara, "hs" for Haystack
- Tag Definitions (contains individual Tag components added to the dictionary)
- TagGroup Definitions (optional -contains individual TagGroups components added to the dictionary)
- Relation Definitions (optional -contains individual Relation components added to the dictionary)
- TagRules (optional -contains individual smart tag definitions and conditions)

TagDictionary properties

Type	Value	Description
Status [component]	text	Read-only field. Indicates the condition of the component at last polling. <ul style="list-style-type: none"> {ok} indicates that the component is polling successfully. {down} indicates that polling is unsuccessful, perhaps because of an incorrect property. {disabled} indicates that the Enable property is set to false. fault indicates another problem.
Fault Cause	text	Read-only field. Indicates why the network, component, or extension is in fault.
Namespace	text string	Descriptive text that reflects the name of the tag dictionary. This is typically a very short string of only a few characters. For example, "n" for NiagaraTagDictionary , "hs" for HaystackTagDictionary .
Enabled [general]	true or false	Activates and deactivates use of the function.
Frozen	true or false	
Tag Definitions	Tag Info List	Container for a collection of standardized tags that have semantic meaning for that namespace (tag dictionary). Each tag in this TagInfoList can be used to add metadata to components, providing additional semantic information. Tags may contain a Validity slot with conditions such as, Always, IsType, etc.
Tag Group Definitions	Tag Group Info List	Container for a collection of TagGroups for a tag dictionary. A tag group provides a structure that lets you add multiple tags to a component, with a single action. Typically tags are in a group because it is common for each of the tags to be assigned to a single component.
Relation Definitions	RelationsInfoList	Container folder that contains a collection of standardized Relation Id's with semantic meaning for that namespace.
validTagRules	Tag Rule List	
validTagGroupRules	Tag Rule List	

Tag Definitions

The Tag Definitions folder in a TagDictionary property sheet contains the collection of standardized tags that have semantic meaning for that namespace (tag dictionary). Each tag in this TagInfoList can be used to add metadata to components, providing additional semantic information. Tags may contain a Validity slot with conditions such as, Always, IsType, etc.

The following implementations of Simple Tag Info are available in the Tags folder of the **tagdictionary** palette. When creating a custom tag dictionary, drag and drop tags from the palette to the Tag Definitions folder in the tag dictionary's property sheet to create the tag definitions for that dictionary.

Tag Type	Value	Description
Marker	Marker (default)	Tag name only. The Marker tag does not require a value. The fact that a component has the tag applied is sufficient to convey semantic information (the tag name).
String	String value	Tag name and string value
Integer	0 (default)	Tag name and numeric value
Long	0 (default)	Tag name and numeric value
Float	0.00 (default)	Tag name and numeric value
Double	0.00 (default)	Tag name and numeric value
Ord	null (default)	Tag name and Ord value

NOTE: Starting in Niagara 4.1, tag dictionaries support adding a single DataPolicy to a TagInfo or a TagGroupInfo component. Also, the TagInfo and TagGroupInfo components have an **Add DataPolicy** action. For details, see "Data Policies".

Tag Group Definitions

Although not required, a tag dictionary may contain tag groups. The Tag Group Definitions folder in a Tag Dictionary property sheet contains the collection of TagGroups for that dictionary. A tag group provides a structure that lets you add multiple tags to a component, with a single action. Typically tags are in a group because it is common for each of the tags to be assigned to a single component.

For example, in the Haystack Tag Dictionary, there is a tag group for "discharge air temp sensor" that contains the following set of individual tags:

- discharge
- air
- temp
- sensor

When you add that tag group to a component all four tags are added to that component with one add action.

The **Device Manager** and **Point Manager** views of a driver, and the **Edit Tags** dialog are the primary methods for adding a Tag Group to a component.

NOTE: When creating or editing a Tag Group, a tagging best practice is to include only those tags that have a corresponding Tag Definition in the parent tag dictionary. One way to guarantee it is to populate the Tag Group's Tag List with tags copied only from the Tag Definitions list in the tag dictionary. Note that in Niagara 4.0, any tag that exists in a Tag Group Definition is *required* to have a corresponding Tag Definition in the same tag dictionary. While in Niagara 4.1 this is no longer required but it is still a recommended best practice.

TagGroup functionality changes for Niagara 4.1

Starting in Niagara 4.1, when using the **Edit Tags** dialog to add a Tag Group, the Tag Group displays in the **Direct Tags** tab as an Ord to the Tag Group itself, and once the change is saved the individual tags of the Tag Group display in the **Implied Tags** tab. This is a change from how Tag Group tags were handled in Niagara 4.0, where using the **Edit Tags** dialog to add a Tag Group resulted in the individual tags displaying in the **Direct Tags** tab.

A Tag Group can contain tags from other tag dictionaries. You can add a tag to a Tag Group that overrides the namespace of the parent tag dictionary. This allows you to define a Tag Group that contains tags from multiple tag dictionaries. Note, that there is no verification that the tag name entered is actually defined in a

tag dictionary. If the Tag Definition does not exist, the added tag is an "ad hoc" tag. It is possible to use ad hoc tags although, a tagging best practice is to include only those tags that have a corresponding Tag Definition in the parent tag dictionary.

You can add a single "Data Policy" to a Tag Group Definitions folder (or to a Tag Definitions folder). A data policy provides additional metadata that can be associated with a tagged component. For more details see "Data Policies".

Converting existing tags to TagGroup Relations

On station startup, the Tag Group Monitor in Tag Dictionary Service scans the station components looking for a complete set of tags from each defined TagGroup in the TagDictionaryService. If a complete set is detected, the Monitor replaces the set of tags on the component with an `n:tagGroup` relation from the component to the TagGroupDefinition. This set of tags then becomes implied tags on the component.

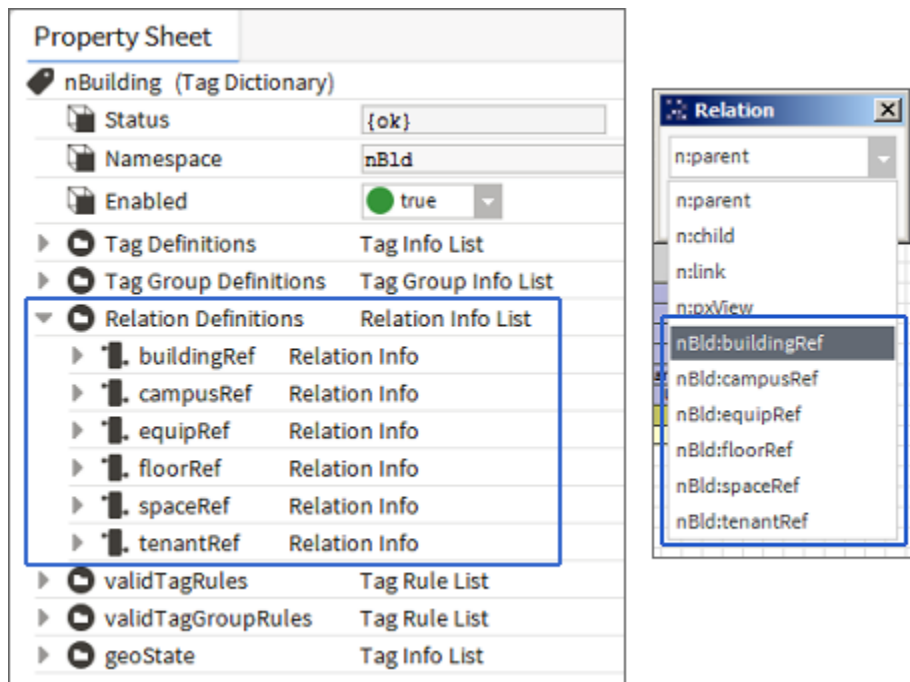
Using the Edit Tag dialog within the Template Editor

When using the **Edit Tags** dialog within the **Template Editor** view, TagGroup tags are assigned to a component as individual direct tags in the template. Upon template deployment, the Tag Group Monitor in the Tag Dictionary Service is notified and the individual tags (of the tag group) are removed from the component and replaced with an `n:tagGroup` relation. The relation, of course, implies the same set of tags.

RelationDefinitions

Tag dictionaries often contain a collection of Relation Definitions (shown in the following image) which are standardized Relation Id's with semantic meaning for that namespace. These relation definitions come into play when adding a relation to a component, in the **Relation** dialog, your choices are limited to the relations that are defined in any of the tag dictionaries installed on your system.

Figure 3 Relation Definitions in custom TagDictionary (left) provide choices seen in the Relation dialog (right)



NOTE: Starting in Niagara 4.1, there is a new implied relation. For devices containing child points that have a **Null Proxy** extension, the `childNullProxyPoint` relation is implied on each of those points.

Tag Rules

The tag rules used in a smart tag dictionary are the primary control mechanism for smart tagging. The tag rules determine the implied tags and relations for each and every entity.

A SmartTagDictionary is a tag dictionary which contains a collection of Tag Rules that define the logic for implying Tag(s) and Relation(s). A TagRule defines a rule that determines when an entity implies a tag or set of tags. Also, a TagRule has a Condition property with a Value property. The value of Condition defines the rule.

TagRules contain three definition lists: Tag List, Tag Group List, and Tag Relation List. When a taggable object is evaluated, eventually the process determines if the object meets the criteria specified on the condition slot of each tag rule. When that happens, if the implied tag does not apply a null value is returned, or it will return a tag (or relation, or tag group) with the value set (if other than a marker tag). Eventually, all of the results from tag rules in all of the smart tag dictionaries, will be merged to form the complete set of implied tags for the object.

The validity slot of a definition (TagInfo, RelationInfo, or TagGroupInfo) is not evaluated in a tag rule or in a tag group definition. It is only evaluated in the tag definitions of a tag dictionary. Any definition that exists in a tag rule or tag group definition is required to have a corresponding tag definition in the tag dictionary.

NOTE: Instructions for creating Tag Rules for a smart tag dictionary is considered to be advanced, developer-level, information and so it is not covered in this document.

Data Policies

Starting in Niagara 4.1, tag dictionaries may include data policies. A data policy provides additional metadata associated with a tag or tag group. The `tagdictionary` palette contains the following DataPolicy components:

- DataPolicy
- BooleanDataPolicy
- EnumDataPolicy
- NumericDataPolicy
- StringDataPolicy

NOTE: Typical tagging operations do not require data policies. The data policy functionality is provided primarily for use by the Niagara 4.1 Analytics engine. For that reason, tag dictionaries may include added data policies.

An **Add Data Policy** action has been added to TagInfo and TagGroupInfo components. Invoking this action prompts you to select a DataPolicy type to add to the selected TagInfo or TagGroupInfo component in a tag dictionary.

You can also add a data policy to TagInfo or TagGroupInfo components by dragging a DataPolicy component from the `tagdictionary` palette and dropping it on the desired TagInfo or TagGroupInfo component.

NOTE: You may not add a data policy if the TagInfo component is a Marker tag, or if the TagInfo or TagGroupInfo component already has a DataPolicy child. Only a single data policy can be added to TagInfo or TagGroupInfo components.

Properties for Data Policy

Name	Value	Description
Min Interval	drop-down with time intervals	The minimum allowed interval when requesting a trend.
Max Interval	drop-down with time intervals	The maximum allowed interval when requesting a trend.

Name	Value	Description
Preferred Time Range	three fields: drop-down list for time period, drop-down list for day of the month, drop-down list for time selector	The default time range of the data when a request does not include a time range.
Preferred Rollup	drop-down list of arithmetic functions	The default rollup of the data when a request does not specify the rollup function.
Preferred Aggregation	drop-down list of arithmetic functions	The default aggregation when a request does not specify the aggregation function.
Units	drop-down list of units of measure	Defines the units of measure for the data reported from the point.
Precision	32 bit (default), 64 bit	Allows you to select 32 bit or 64 bit options for the history data logging. The 64 bit option allows for higher level of precision but consumes more memory.
Totalized		
Trend Required	true (default) or false	When true, all points must have a trend.
Units	drop-down list of units of measure	Defines the units of measure for the data reported from the point.

About the Tag dictionary Service

The Tag Dictionary Service, located in a station's Services directory, is the container for all tag dictionaries installed in the station.

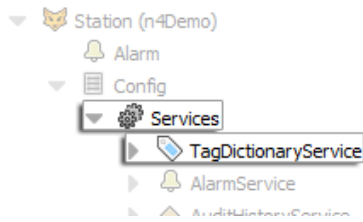
Tag Dictionary Service

The **Tag Dictionary Manager** is the main view for the Tag Dictionary Service. The service has a property for defining a default namespace so that queries that do not specify a namespace are resolved on this default namespace. For example, if you execute a NEQL query for "point" (instead of "n:point"), and the default namespace ID is set to "n", your query returns all objects tagged with "n:point".

Note the following information about the Tag Dictionary Service:

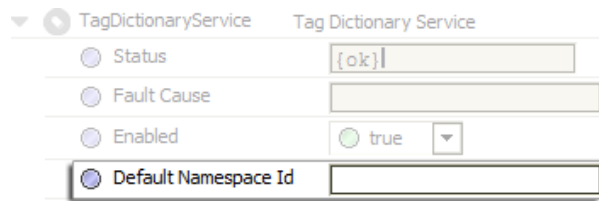
- The **tags** license is required in order to use the TagDictionaryService and tag dictionaries on a station.
- A station can support only one tag dictionary service.
- Neither the tag dictionary service nor any tag dictionaries are strictly required for tagging or for performing NEQL queries. However, without tags or tag dictionaries, a station is not be able to take advantage of most of the functions available from tagging objects. Tags and tag dictionaries are fairly lightweight and therefore are included by default in all stations created by the new station wizard.
- Installed tag dictionaries belong under the **TagDictionaryService**. They are not legal anywhere else in the station.

Figure 4 TagDictionaryService is located in the Services directory



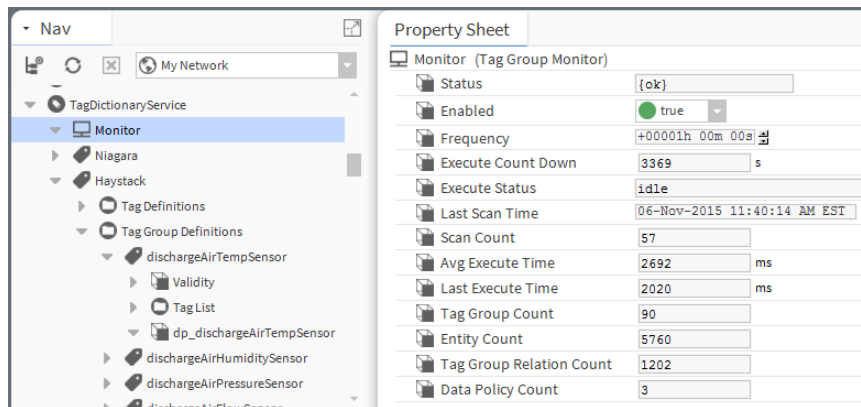
Property	Values	Description
Enabled [general]	true or false	Activates and deactivates use of the function.
Status [component]	text	Read-only field. Indicates the condition of the component at last polling. <ul style="list-style-type: none"> • {ok} indicates that the component is polling successfully. • {down} indicates that polling is unsuccessful, perhaps because of an incorrect property. • {disabled} indicates that the Enable property is set to false. • <i>fault</i> indicates another problem.
Fault Cause	text	Read-only field. Indicates why the network, component, or extension is in fault.
Default Namespace Id	Text string	This TagDictionaryService property provides a field that is used to indicate a default tag namespace (tag dictionary) that is used if there is not a namespace provided as part of a query. For example, if the default namespace is set to "hs", then a search query that includes only "ahu" would return all objects that are tagged with "hs:ahu".

Figure 5 Default Namespace Id property in TagDictionaryService property sheet view



Tag Group Monitor

Starting in Niagara 4.1, the **TagDictionaryService** has a frozen **TagGroupMonitor** slot named **Monitor**. The primary responsibility of this component is to monitor the addition or removal of component tags and the maintenance of **n : tagGroup** relations.



Converting existing tags to TagGroup Relations

On station startup, the Tag Group Monitor in Tag Dictionary Service scans the station components looking for a complete set of tags from each defined TagGroup in the TagDictionaryService. If a complete set is detected, the Monitor replaces the set of tags on the component with an `n:tagGroup` relation from the component to the TagGroupDefinition. This set of tags then becomes implied tags on the component.

Using the Edit Tag dialog within the Template Editor

When using the **Edit Tags** dialog within the **Template Editor** view, TagGroup tags are assigned to a component as individual direct tags in the template. Upon template deployment, the Tag Group Monitor in the Tag Dictionary Service is notified and the individual tags (of the tag group) are removed from the component and replaced with an `n:tagGroup` relation. The relation, of course, implies the same set of tags.

Properties

Name	Value	Description
Status [component]	text	Read-only field. Indicates the condition of the component at last polling. <ul style="list-style-type: none"> {ok} indicates that the component is licensed and polling successfully. {down} indicates that polling is unsuccessful, perhaps because of an incorrect property. {disabled} indicates that the Enable property is set to false. {fault} indicates another problem.
Enabled [general]	true or false	Activates and deactivates use of the function.
Frequency	+00000h 00m 00s	Specifies how frequently the monitor will do a complete scan of the station looking for applied tag group tags.
Execute Count Down	0000s	Seconds until next scan.
Execute Status	idle	
Last Scan Time		Date and time stamp
Scan Count		Number of scans.
Avg Execute Time	0000ms	The average time in milliseconds that it has taken to execute the scan.
Last Execute Time	0000ms	The time in milliseconds of the last scan execute.

Name	Value	Description
Tag Group Count	90	The total number of tag groups found in the installed tag dictionaries.
Entity Count	5706	Total number of entities (components) included in the scan. NOTE: The services folder is not included in the scan.
Tag Group Relation Count	1202	Total number of <code>n:tagGroup</code> relations found.
Data Policy Count	3	Total number of data policies found in the installed tag dictionaries.

Plugins in the tagdictionary module

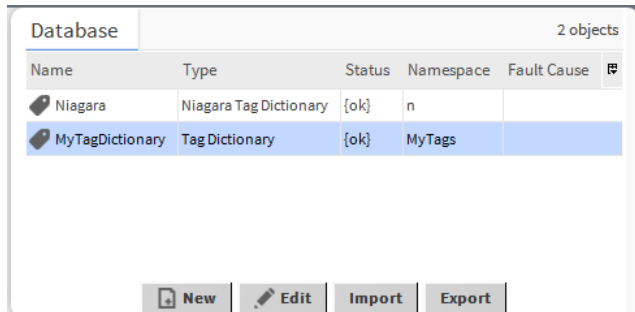
Plugins provide views of components and can be accessed in many ways. For example, double-click a component in the Nav tree to see its default view. In addition, you can right-click on a component and select from its **Views** menu.

For summary documentation on any view, select **Help**→**On View (F1)** from the menu or press **F1** while the view is open.

View(s) in the `tagdictionary` module are described in the following sections.

Tag Dictionary Manager view

Tag Dictionary Manager, the default view for the Tag Dictionary Service, lists all tag dictionaries installed on the station. The view provides functionality for creating, editing, importing and exporting tag dictionaries.



The **New** button at the bottom of the view allows you to create either a new tag dictionary or a new smart tag dictionary which you can then export to a CSV file for editing. You can edit an exported tag dictionary

The **Edit** button at the bottom of the view allows you to change the tag dictionary properties Name, Namespace, and Enabled status.

The **Tag Dictionary Manager** view allows you to **Import** and **Export** tag dictionaries in a standard CSV file format. You can edit an exported tag dictionary in a CSV-compatible spreadsheet program (either online or offline) to add or remove tag definitions, tag groups definitions, relation definitions, as well as validity rules. Afterwards, you may import the edited CSV file, thereby updating the existing tag dictionary.

Glossary

data policy	A data policy provides additional metadata that can be associated with a tagged component. For more details on tags and tagging, see the Tagging Guide.
Haystack	An extensible Semantic Web Browser developed by the Haystack research group at the MIT Computer Science and Artificial Intelligence Laboratory (http://haystack.lcs.mit.edu). The project explores how the Semantic Web data model (a Resource Description Framework — RDF) can be applied by users to better organize, navigate, and retrieve information, both personal and shared (www.w3.org/2005/04/swls/BioDash/Demo/What is Haystack.html).
Haystack tag dictionary	<p>A smart tag dictionary (namespace) containing a collection of tags developed by Project Haystack, which can be used for semantic modeling of building control entities, i.e. site tags, building tags, equipment tags, point tags, geo-location tags, etc.</p> <p>The Haystack dictionary is indicated by the <i>hs</i> character, followed by a colon character (:).</p> <p>The Haystack dictionary is a result of the work of the Open Source Initiative (OSI) hosted on the website http://project-haystack.org.</p>
namespace	A container for a set of names in a naming system. A tag dictionary is a namespace.
Niagara tag dictionary	<p>A tag dictionary (namespace) containing a collection of tags developed for Niagara systems, that are used for semantic modeling of specific building control entities, i.e. networks, devices, equipment, points, sites, buildings, geo-location, histories, etc.</p> <p>The Niagara dictionary, is a type of Smart Tag dictionary, therefore it applies Implied Tags and Implied Relations to components and links. This allows queries to find these components based on type, linkage, hierarchy or combinations of these. The Niagara tag dictionary is included by default in all stations created using the New Station tool.</p> <p>The Niagara dictionary is indicated by the <i>n</i> character, followed by a colon character (:).</p>
semantic information	Metadata used to indicate the purpose of a device, that is, what the device is, what each of its data points means, and how devices are related to each other.
tag	<p>A piece of semantic information (metadata) associated with a device or point (entity) for the purpose of filtering or grouping entities. Tags identify the purpose of the component or point and its relationship to other entities. For example, you may wish to view only data collected from meters located in maintenance buildings as opposed to those located in office buildings or schools. For this grouping to work, the metering device in each maintenance building includes a tag that associates the meter with all the other maintenance buildings in your system.</p> <p>JACEs are associated with Supervisors based on tags; searching is done based on tags.</p> <p>Tags are contained in tag dictionaries. Each tag dictionary is referenced by a unique namespace.</p>

tag dictionary	Tag dictionaries contain a set of tag definitions, and may contain tag group definitions, relation definitions, as well as tag rules for smart tags.
taggable spaces	The implementation of tags for all common data types: components, files, histories and alarms.

Index

A

Ad Hoc tags	29
adding	10
adding	
tag groups	9, 13
tags	9

D

Data Policy	
component	35
properties	35
Direct Tags	29

E

Edit Tags dialog	27
Editing tags in template	16

I

implied relations	
viewing with Spy View	18
implied tags	
viewing with Spy View	18
Implied Tags	29
viewing	17

O

Offline tagging	27
Online tagging	27

R

Relation Definitions	30, 34
Removing a tag	11

S

Smart Tag	
tag rules	30
Smart Tag Dictionary	30
Spy View	18

T

tag	
adding in Database pane	12
adding in Discovered pane	11
adding using Batch Editor	15

Tag Definitions	30, 32
tag dictionaries	20
creating custom	20
custom	30
editing exported	21
exporting	20
importing	24
types of	30
Tag Dictionary	
properties	30
Tag Dictionary Manager view	39
Tag Dictionary Service	36
tag group	
adding	13
Tag Group	
adding	9
adding tag to	14
Tag Group Definitions	30, 33
Tag Group Monitor	38
Tag mode	
exiting	19
in Manager views	11–12
selecting	19
tag rules	
in smart tag dictionary	35
Tagging	
license requirements	7
overview	7
process	7
tags	
types of	29