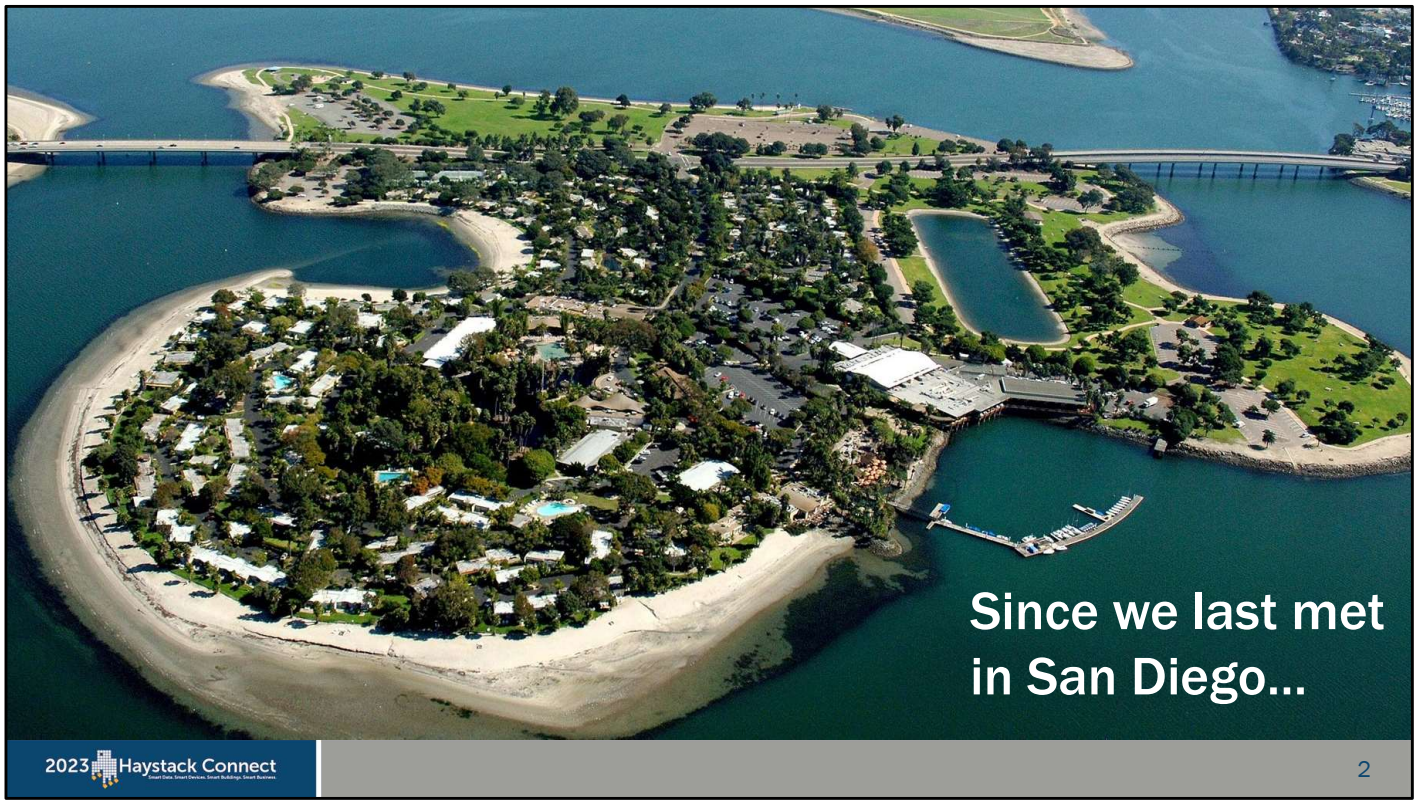
Haystack in Practice – The End-User Perspective & Real-World Applications

Haystack 4 Tag Dictionary in Niagara

Stephen Holicky | Product Director | Tridium

Eric Anderson | Software Engineer | Tridium

2023 Haystack Connect
Smart Data. Smart Devices. Smart Buildings. Smart Business.


Since we last met
in San Diego...


niagara4
2015
4.0
2016
4.1
4.2
2017
4.3
2018
4.4
4.6
2019
4.7
4.8
2020
4.9
2021
4.10
4.11
2022
4.12
2023
4.13
2023 Haystack Connect

HAYSTACK IN YOUR WORKFLOW

Tag Based Graphics

HTML5 Tag Manager

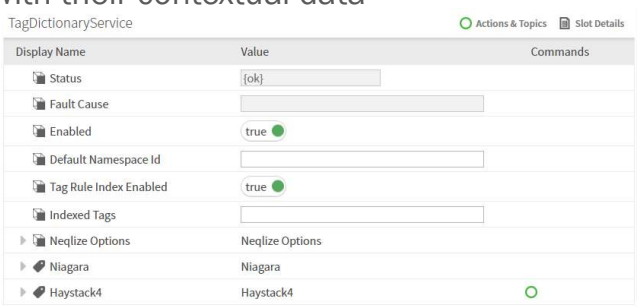
HAYSTACK IN YOUR WORKFLOW

Tag Based Logic Builder

Haystack 4 Support

HAYSTACK 4 SUPPORT

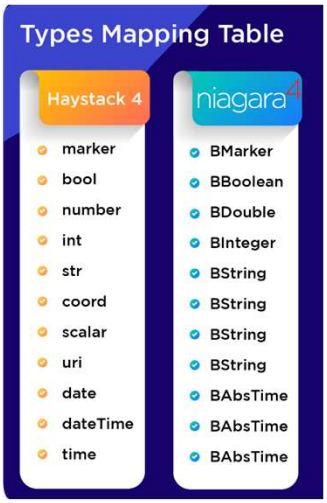
Haystack 4 enriched the taxonomy and an ontology to capture machine-readable relationships of things along with their contextual data



HAYSTACK TAG DICTIONARY

Aligns Niagara Tagging with the latest Haystack technologies.

Updatable in-place and without a module update from Tridium.



EASY MIGRATION

Haystack 4 items equivalent to the Haystack 3 versions are added during migration action.

Haystack 4 Tag Dictionary in Niagara

DEEP DIVE

PRIOR ART

- Haystack Connect 2022 Schedule
  - Niagara Haystack 4 Tag Dictionary on Vimeo
- Connections Magazine - Sep 2022
- TridiumTalk: Niagara 4.13 Feature Preview - Haystack 4

Information about Haystack 4 support in Niagara is already in a few places. There is a link to the video of my presentation at last year's Haystack Connect available at haystackconnect.org. There was an article in the September 2022 Connections magazine. And, this past April, I did a TridiumTalk on Haystack 4- you can find a link to that on the events page at tridium.com.

DEEP DIVE

- Haystack 3 in Niagara recap
- Haystack 4 in Niagara summary
- Migrating from 3 to 4
- Demo

Today, I'll do a quick recap of how Haystack 3 was supported in Niagara and then summarize the support for Haystack 4. Then, I'll talk about support for creating Haystack 4 tags, tag groups, and relations based on your existing Haystack 3 items. Finally, I'll show a quick demo of these things.

HAYSTACK 3 IN NIAGARA

- Significant post-processing
  - tags.csv + extra columns
  - Equip point lists ⟶ equip.csv
- Complex updating and customization
  - Tags and equip import files

Support for Haystack 3 was a little cumbersome. Updated Project Haystack definition files had to be processed before they could be used by Niagara. Extra columns had to be added to the tags.csv and all the equip point lists had to be combined into an equip.csv. To update or change items in the dictionary, users had to create tag and/or equip import files. It was all very complex.

HAYSTACK 4 IN NIAGARA

- Based mostly on unaltered defs and protos.json
  - 4.13 contains 3.9.14
  - Can update dictionary in-place with newer versions
- Supplemented by config file
  - Tagging convenience features
  - Can update dictionary with custom version

For Haystack 4, we tried to do as much as possible with the unaltered definition files. We'll be shipping the latest with each Niagara release but if new versions come out between releases, the dictionary should be able to consume them. There is an extra but minimal configuration file that provides support for some tagging convenience features. This file can also be customized by users and fed to a deployed dictionary.

TAGS

- All defs outside lib:ph that can be mapped to a Niagara type

Haystack 4	Niagara
marker	BMarker
bool	BBoolean
number	BDouble
int	BInteger
str, coord, scalar, uri	BString
date, dateTime, time	BAbsTime

The tags in the dictionary are imported from Project Haystack's defs.json. Most, if not all, defs outside of the lib:ph library are imported if they can be mapped to a Niagara tag value type.

lib:ph TAGS

- The entity and geoPlace defs
- Subtypes of entity
- Defs that are a tagOn entity or geoPlace or one of their subtypes
  - geoCity is tagOn geoPlace
  - kind, tz, unit, etc. are tagOn point, a subtype of entity
- Defs listed in the Niagara configuration file: min, max, input, and output

From the lib:ph library, we import the entity and geoPlace defs explicitly. Then, we import any defs that are a "tagOn" those defs or their subtypes. For example, the geoCity def is imported because it's a tag on the geoPlace def. Likewise, defs such as kind, time zone, and unit are imported because they are tags on the point def, which is a subtype of the entity def. Finally, we import a few additional tags specified in that Niagara configuration file.

CHOICE TAGS

- DynamicEnum tag for each choice def
- EnumRange based on choice values
  - Subtypes of the choice def's "of" def
  - Or, subtypes of the choice def
- Tag rule implies a marker tag or tag group based on the enum value

In Haystack 4, there are choice defs that define an exclusive relationship between a set of marker tags or tag groups. In Niagara, we model these as DynamicEnum tags with an EnumRange based on the choice values in the defs.json. There are two ways choice values are determined in Haystack 4. First, the subtypes of the choice def's "of" def. Second, the subtypes of the choice def itself. When one of these choice values is selected on the DynamicEnum tag, a tag rule implies the corresponding marker tag.

TAG GROUPS

- Conjuncts from defs.json
- Protos from protos.json
  - Exclude those that are already a tag or conjunct tag group: equip, point, space, humidifier-equip, networking-device, etc.

Tag groups in the Haystack 4 dictionary come from the conjuncts in defs.json and the protos in protos.json. Protos that are already a tag or conjunct tag group are not added again.

RELATIONS

- String value tag added for each ref def subtype
- Relation added for each subtype except the ID def
- Use of Niagara relations preferred
- Exported as tags by nhaystack

For every def that is a subtype of the ref def, we have added a String value tag. The value of these tags could be manually set to the ID of the parent or substance source entity. Instead of that, however, we recommend using Niagara relations. For every subtype except for ID, we have added a relation to the dictionary. When using the nhaystack module, these relations are exported as tags with their value set to the ID of the relation's endpoint.

SMART RELATIONS

- equipRef implied on descendant proxy points of a component tagged with "equip"
- equip tag no longer implied on all BDevices
- spaceRef implied on sub-equips and equip points
- siteRef implied on sub-spaces, sub-equips, and equip points
- systemRef implied on sub-equips

As with the Haystack 3 dictionary, we've included some smart relations to assist with relating points to equips to spaces to sites. Note that the equip tag is not implied on BDevices in the Haystack 4 dictionary as it was in the Haystack 3 dictionary. We've more recently added the systemRef smart relation- if a parent equip has a systemRef relation to a component with the system tag, a systemRef relation will be implied from any sub-equips to that system component.

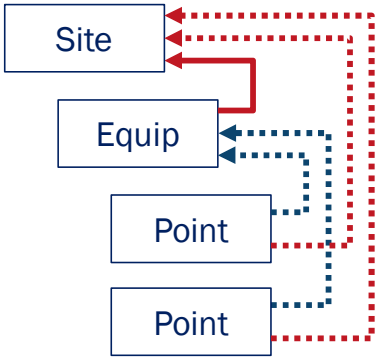
SITE-EQUIP-POINT

In this example, a component with the equip tag has a siteRef relation to a site component.

SITE-EQUIP-POINT

EquipRef relations are implied from the descendant proxy points to that equip component.

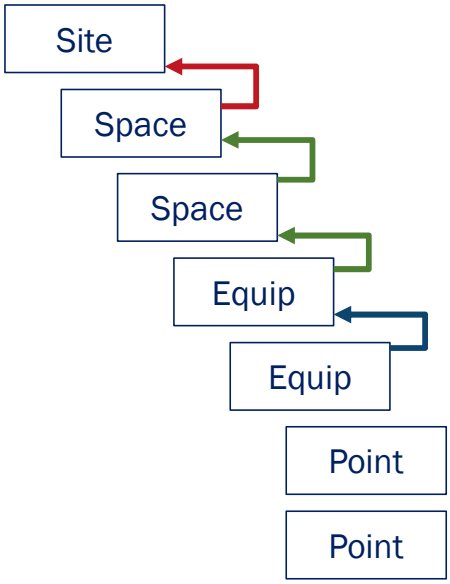
SITE-EQUIP-POINT



h4:siteRef direct	←
h4:siteRef implied	←
h4:equipRef implied	←

Additionally, a siteRef relation is implied from the equip points to that equip's site component.

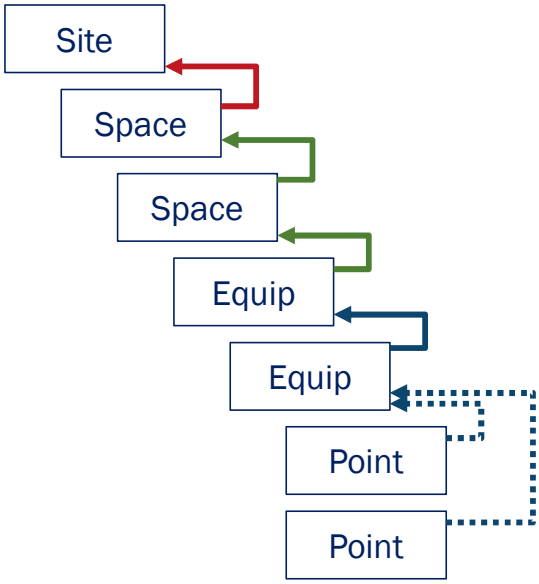
SITE-SPACE-SPACE-EQUIP-EQUIP-POINT



h4:siteRef direct	
h4:siteRef implied	
h4:spaceRef direct	
h4:spaceRef implied	
h4:equipRef direct	
h4:equipRef implied	

In this example, a sub-equip has an equipRef relation to a parent equip, the parent equip has a spaceRef relation to a space, that space has a spaceRef relation to a parent space, and the parent space has a siteRef relation to a site.

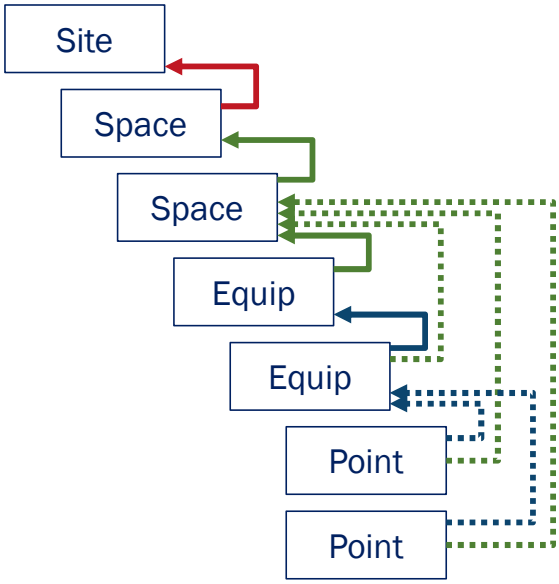
SITE-SPACE-SPACE-EQUIP-EQUIP-POINT



h4:siteRef direct	
h4:siteRef implied	
h4:spaceRef direct	
h4:spaceRef implied	
h4:equipRef direct	
h4:equipRef implied	

As before, the equip points get an implied equipRef relation to the sub-equip.

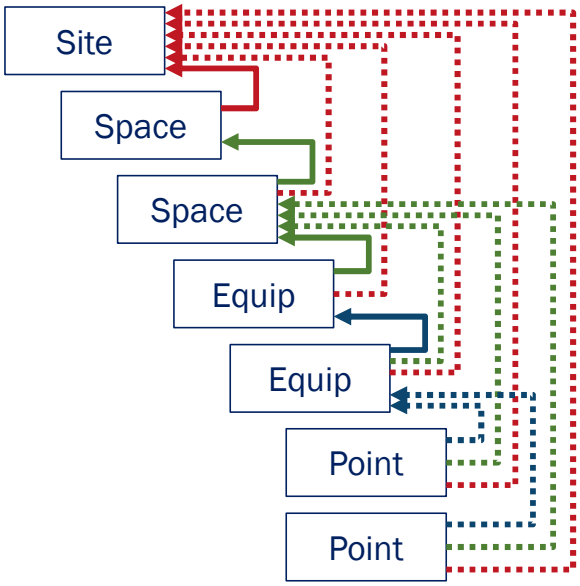
SITE-SPACE-SPACE-EQUIP-EQUIP-POINT



h4:siteRef direct	
h4:siteRef implied	
h4:spaceRef direct	
h4:spaceRef implied	
h4:equipRef direct	
h4:equipRef implied	

The sub-equip and its equip points get an implied spaceRef relation to the parent equip's space.

SITE-SPACE-SPACE-EQUIP-EQUIP-POINT



h4:siteRef direct	
h4:siteRef implied	
h4:spaceRef direct	
h4:spaceRef implied	
h4:equipRef direct	
h4:equipRef implied	

And, the equip points, equips, and spaces get an implied siteRef relation to the parent space's site.

CUSTOM TAG RULES

- SiteRef, SpaceRef, EquipRef smart relations
- Imply simple tags
  - point on BControlPoints
  - bacnet on BBacnetNetworks
- Imply smart tags
  - unit value copied from units facet
  - id value: nspace:{stationName}~slot:{slotPath}

The smart relations just described are implied using custom tag rules specified in the extra Niagara configuration file. There are also custom tag rules that imply simple tags, such as the point tag on BControlPoints and the bacnet tag on BBacnetNetworks. There are custom tag rules that imply smart tags, such as the unit tag whose value comes from the units facet and the id tag. In the Haystack 4 dictionary, the generated ID value is a combination of the station name and slot path.

CHOICE TAG RULES

- "of" def subtypes
- Choice def subtypes

Another set of tag rules implies the choice value marker tags.

INHERITANCE TAG RULES

- Based on the def type inheritance tree
- For example, water ⟶ liquid ⟶ fluid ⟶ substance ⟶ phenomenon

The final set of tag rules are based on the def type inheritance tree in Haystack 4 and generated based on defs.json. For example, water is a liquid is a fluid is a substance is a phenomenon. When the water tag is applied to a component, either directly or implied, the liquid, fluid, substance, and phenomenon tags are also implied.

MIGRATING HAYSTACK 3

- Action to add Haystack 4 equivalents for direct tags, tag groups, and relations
- Haystack 3 items not removed
- Not yet migrating tag-based bindings or tag rules
- Mostly copying tag name and value

To support the transition from Haystack 3, the Haystack 4 dictionary includes an action that adds Haystack 4 equivalents for direct Haystack 3 tags, tag groups, and relations. The Haystack 3 items will stay in-place. We are not yet adding Haystack 4 equivalents to tag rules or modifying the NEQL queries in tag-based PX bindings. Mostly, the Haystack 3 tag names and values are simply copied.

MIGRATION EXCEPTIONS

- project-haystack.org/doc/docHaystack/Changes3to4
- Encoded in haystack3MigrationConfig.csv
- Alternate configuration file may be specified

However, as documented on this Project Haystack page, there are some changes between 3 and 4. These exceptions are captured in a migration configuration file. Users can create and use their own migration file if necessary.

MIGRATING TAGS AND RELATIONS

- Renaming: ahuRef ⟶ airRef
- Tag to multiple tags: dew ⟶ air, dewPoint
- Tag to tag group: co2 ⟶ co2-concentration
- Query based: apparent ⟶
  - if temp or weatherPoint: air, feelsLike
  - else: apparent
- Removed: connection, device1Ref, device2Ref, reheating

For tags and relations, the exceptions are handled in different ways. Some are only renames, such as ahuRef to airRef. Some tags in Haystack 3 became multiple tags in Haystack 4 such as the dew tag for which air and dewPoint are added. Other tags became tag groups in Haystack 4, such as co2 to the conjunct co2-concentration. In a few cases, a query is required to pick the Haystack 4 equivalents. For example, “apparent” in Haystack 3 if paired with “temp” or “weatherPoint” becomes “air” and “feelsLike” in Haystack 4. Finally, some tags and relations have been removed from Haystack 4 and no equivalent is added.

MIGRATING TAG GROUPS

- Use Haystack 4 tag group with additional point tag
  - discharge-air-temp-sensor ⟶ discharge-air-temp-sensor-point
- Explicit mapping in migration configuration file
  - energy-net-sensor ⟶ total-net-ac-elec-active-energy-sensor-point

For tag groups, there were no identical equivalents. For most Haystack 3 tag groups, the closest matching tag group in Haystack 4 only differs by the point tag. In that case, that closest matching tag group is added. For example, the Haystack 4 discharge-air-temp-sensor-point tag group is added for the discharge-air-temp-sensor Haystack 3 tag group. For other tag groups, there is an explicit mapping in the migration configuration file. For example, the Haystack 4 total-net-ac-elec-active-energy-sensor-point tag group is added for the energy-net-sensor Haystack 3 tag group.

Haystack 4 Tag Dictionary in Niagara

DEMO

Haystack 4 Tag Dictionary in Niagara

Q&A