Project Haystack

Milan Milenkovic on behalf of Haystack, in collaboration with John Petze and Brian Frank
milan@iotsense.com

Project Haystack: Official Positioning

The role of Project-Haystack is to bring together a community of industry constituents in a collaborative open-source effort to develop a standardized approach to representing and using metadata across a wide range of applications. Project-Haystack standardizes semantic data models and Web services to access and communicate those models, with the goal of making it easier to unlock value from the vast quantity of data being generated by the smart devices and equipment systems that permeate our homes, buildings, factories, and cities. Applications include automation, control, energy, HVAC, lighting, and other environmental systems.

The Project-Haystack vision is to streamline the use of data from the Internet of Things including, but not limited to, building and energy systems, by creating a standardized approach to defining "data semantics" and related services and API's to consume and share the data and its semantic descriptors. Project Haystack makes it easier to unlock value from the vast quantity of data being generated by smart devices by making data "self-describing".

The Project-Haystack initiative was started in 2011 to directly address this need. It is structured as an open source, community-driven initiative modeled on the open source specifications found in the software and IT industry. With a common methodology for defining metadata, and a common vocabulary or taxonomy (libraries of community defined tags) to give meaning to the data collected by smart devices and systems, we can more easily take advantage of a wide range of software tools to visualize, analyze and generate value from operational data.

Motivation (The Problem)

For example, if we obtain a data item from a building automation system (BAS) and it has a value of 77.6 we can't do any effective analysis until we understand whether it is 77.6 degrees F, or degrees C, or PSI, RPM, or kW, etc. "Units" therefore is one good example of an essential descriptor we need in order to understand and use our data.

If all we know are the units (Degrees F for example), we still don't know much about the significance of the value 77.6. If it's a zone temperature it might be a bit warm for occupants. If it's a return air temperature it's right where we want it to be.

Let's say the sensor with the value of 77.6 is named *zn3-wwfl4* with following properties of interest for building management:

- Is a zone temperature
- Is an exterior zone
- Is south facing
- Is supplied by a VAV box
- Is served by AHU-1
- Is operated on occupancy schedule #1, which is 7:30 AM - 6:30 PM
- Has an occupied cooling setpoint of 74 degrees F

Armed with this additional information, we could determine that a value of 77.6 is not proper for 9:00 AM on a weekday - it's too hot and will lead to occupant complaints. What enabled us to make that determination, however, was a significant amount of information about the meaning of the specific sensor. That information is typically not recorded in the control system (or any single location), and is not available across buildings and systems in any consistent "machine readable" format.

Haystack Metadata Tagging: an Example

Given our earlier example, a record representing the point zn3-wwfl4 with its associated metadata might look like the following:

```
id:           150a3c6e-bef0ee0e        (RecId)
dis:          zn3-wwf14                (Str)
sensor:       ✓                        (Marker)
air:          ✓                        (Marker)
temp:         ✓                        (Marker)
unit:         °F                       (Str)
curVal:       77.60 °F                 (Number)
equipRef:     Carytown AHU-4           (Ref)
siteRef:      Carytown                 (Ref)
tz:           New_York                 (Str)
zone:         ✓                        (Marker)
vav:          ✓                        (Marker)
floor:        4                        (Number)
scheduleRef:  occSchedule-1            (Ref)
```

The “point” has a unique ID and then is described by a series of tags. In our example, we have the following:

- **dis**: The display name we want associated with the sensor point. This could be the original descriptor from the system supplying the data, or it could be a new name/descriptor.
- **sensor**: a marker tag that tells us this is a sensor point (versus a control point).
- **air**: a tag that tells us this sensor is measuring air.
- **temp**: a tag telling us this sensor is measuring temperature (versus pressure for example.
- **unit**: a tag that tells us the units of measure.
- **curVal**: a tag that holds the most current value supplied by the system.
- **equipRef**: a tag that associates this point with a specific piece of equipment.
- **siteRef**: a tag that tells us the site (building) this point is associated with.
- **tz**: a tag that tells us the time zone of the site.
- **zone**: a tag that tells us this is a zone (space) in a building.
- **vav**: a tag that tells us that this sensor is measuring a zone supplied by a VAV box.
- **floor**: a tag that tells us which floor of the building the sensor is located.
- **scheduleRef**: a tag that tells us which occupancy schedule is associated with the area this sensor is measuring.

The indicators in parenthesis tell us the type of tag: recID: RecID - a unique identifier, Number, String, Marker, and Reference, which points to an associated entity.

With these tags it is now possible for both humans and machines to interpret the meaning of this point. This enables streamlined value creation. For example, a graphical visualization software application could interpret this information to automatically generate a graphical view of the sensor within the appropriate zone. An analytics application could automatically apply appropriate rules to this sensor to identify improper conditions.

An Open-Source, Community-Driven Approach

Project Haystack is operated as an open source project, community-driven initiative modeled on the open source efforts found in the software industry, which makes it easy for anyone to get involved. Anyone can easily take advantage of the work of Project Haystack and contribute to it. All collaboration is done on the discussion forum at http://project-haystack.org/. Anyone can contribute on the forum by signing up on the web site.

Domain experts in a given area such as chillers, data centers, or refrigeration can join or start a discussion. Equipment manufacturers who would like to see specific tag models for their products are also a great source of input. All of the work done by Project Haystack is easily available to the industry. It can be downloaded without even registering an account on the web site. There is no cost or obligation associated with using Project-Haystack techniques, tagging libraries and open source reference implementations.

Project Haystack is More Than Metadata Tagging

It's important to note that Haystack is more than one thing. First, it's the data modeling methodology—the simple, flexible tagging approach that can be used in media from Excel spreadsheets and CSV text files, to data tables in embedded devices, XML representations, web services and others.

Second, it's the consensus developed tagging libraries (taxonomies) published and made available for download and use (at no cost). You can find all of the tagging libraries developed by the Project-Haystack community here: http://project-haystack.org/tag

Third, it's the REST style communication services designed to exchange Haystack tags between applications.

Fourth, it's the software reference implementations and complementary applications being developed by various community members and companies. As of the date of publication these included:

- Haystack Java Toolkit: lightweight J2ME compliant client and server implementation
- NHaystack: Niagara module to add Haystack tagging and the Haystack REST API
- Haystack CPP: C++ Haystack client and server implementation
- Haystack Dart: client library for Dart programming language
- NodeHaystack: node.js client/server implementation
- PyHaystack: Python Haystack toolkit

You can find links to download all of these software reference implementations here: http://project-haystack.org/download

A Flexible Extensible Model that Can Be Used Beyond Consensus Approved Tags

Project Haystack methodology can be used beyond just the consensus approved tags available at any point in time. Projects and products can add custom tags to represent information important to their needs without requiring submission or approval. Applications that are "Haystack tag-aware" can be easily extended to interpret new community approved tags and custom tags as they are added. Numerous companies have started their adoption of Haystack by developing their own haystack compatible tagging libraries and then returned that work to the community over time.

What Will the World Look Like When We Have Standard Tagging Models to Describe Building, Equipment and Device Data?

Today, using building system data beyond the system generating it requires significant time and effort because these contextual models must be built by hand. This means that the opportunity to use data to reduce energy use, and improve operational efficiency and performance is being stifled by the labor costs associated with manual mapping of system data and the inconsistency of those efforts. Project Haystack defines this common vocabulary so that we can build models of our buildings and systems enabling us to more efficiently derive value from all the data our building automation systems and smart devices are producing. With Project Haystack models we can transition from a manual process to an automated process and unlock the value of our operational data.

With metadata "tagging" a software applications can now automatically find and interpret the data they need to provide value to the user. Two specific examples of the benefits of metadata tagging being accomplished today include:

- Software today can automatically generate equipment graphics and system views simply by interpreting tags on the control system data. Hours of manual graphics assembly is thereby eliminated reducing project cost and increasing value creation.

- With proper metadata, analytics applications can quickly consume data from equipment systems and interpret patterns in operational data to identify faults, deviations, and trends that can be addressed to improve efficiency and insure proper operation of equipment systems.

Project Haystack enables a future where a push of a button can turn data into true intelligence reducing the cost of intelligent building systems and enabling operational teams to better understand and improve the operation and performance of buildings.

More Information:

Complete information on Project haystack can be found at www.project-haystack.org. You can contact members of the board of directors at: projecthaystackinfo@gmail.com