A Vision for 223P, Brick, and Project Haystack

Gabe Fierro

January 5, 2022

It is important to remember that interoperability is not a goal, but it means. Our goal is to facilitate the deployment of energy efficient applications turning buildings into grid interactive assets [1]. This requires the nimble application of data to such tasks, across a diverse spectrum of heterogeneous environments. *Structured metadata* is an enabling technology which reduces the inherent complexity of interfacing with such diverse resources, making it possible for individuals or small teams to scale their effort across hundreds, thousands for even millions of buildings.

[1] Harry Bergmann, Cory Mosiman, Avijit Saha, Selam Haile, William Livingood, Steve Bushby, Gabe Fierro, Joel Bender, Michael Poplawski, Jessica Granderson, et al. Semantic interoperability to enable smart, grid-interactive efficient buildings. Technical report, Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States), 2020

The set of metadata models proposed to address this task has steadily grown over the past decade. The design, implementation, and modes of intended use for each of these models depend on the perspectives and experiences and concerns of their intended users and originating communities. At the same time, the conversation around building metadata consistently and misleadingly pits these models against each other as if they are direct competitors. This ultimately presents a confusing story to the intended adopters and consumers of building metadata: which model to choose or invest in? The question I aim to address in this document is: how can we *simplify* the story around standardized building metadata while (a) *preserving existing investments* in metadata and software, and (b) *reducing or eliminating* redundant effort between fragmented metadata development communities?

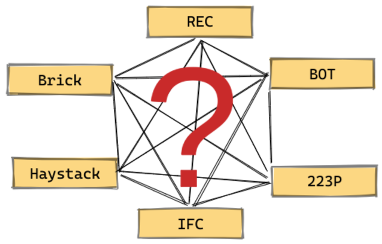

Figure 1: Is one path forward to enable translation between all competing models?

The diversity of data driven tasks for buildings suggests that there is no universal metadata schema or data model that solves all problems. This is well documented, albeit indirectly, in the literature. It has also been proved through the mirror existence of multiple metadata standards: if existing data models solve all the problems, then there would not be as much of a need to develop and explore additional models. Indeed, we can look at metadata models for buildings and building data as addressing particular shortcomings in their predecessors and contemporaries:

- IFC [2] addresses the lack of standard exchange for digital geometry
- Haystack [3] addresses the lack of structure in BMS point labels
- Brick [4] addresses the lack of standardized semantic information in Haystack
- RealEstateCore [5] addresses the lack of property management

[2] Industry Foundation Classes (IFC). `https://www.buildingsmart.org/standards/bsi-standards/industry-foundation-classes/`

[3] Project Haystack. `https://project-haystack.org`

[4] Brick Schema. `https://brickschema.org`

[5] RealEstateCore. `https://realestatecore.io`

metadata in other ontologies

- 223P addresses the lack of machine readable and verifiable descriptions of the topology of building subsystems.
- BTO [6] makes it possible to refer to generic IFC spatial concepts inside the RDF framework

[6] Building Topology Ontology (BOT). `https://w3c-lbd-cg.github.io/bot/`

GIVEN THESE NATURAL DIVISIONS and the distributed nature of the individuals, teams, and communities working on metadata for buildings, I think it makes the most sense to focus on how these models can relate and connect to one another. This is explicitly against defining a unification scheme by which the metadata in any one data model can be fully expressed in any other data model (Figure 2).

Instead, let us focus on where existing metadata models shine and build the glue that ensures that other complementary data models can make use of that metadata when needed, and where appropriate. This will require reducing the scope of some metadata models [7] and migrating concepts from some models to others.

The intent is to define and produce a cohesive "metadata stack" in which the boundaries between metadata models are clearly and rigorously defined. This will disambiguate which metadata models should be used for which tasks, and how the metadata models may be used together.

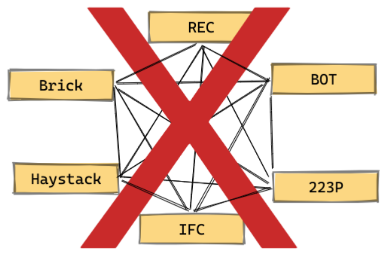

Figure 2: Full interoperability between all available building metadata standards is unrealistic and unnecessary

[7] (or at least encouraging those communities to do so)

Separation of Concerns

We must be principled in assigning responsibility to each of the participating metadata models in order to eliminate (or at least reduce) unnecessary redundancy and to clarify the role of each model. To this end, I propose the following “facets” to organize the participating metadata models:

- **Data**: What data sources and sinks exist, how to access them, their properties and immediate context. “Application-facing”.
- **Control**: How data is processed or leveraged to produce control decisions
- **Network Representation**: How devices, objects, data are represented and accessed on the network
- **Topology and Composition**: How assets are connected and placed within the building, how the building is composed

Each metadata model should bound its scope to one of these facets, save for the concepts that are necessarily shared between models in order to permit interoperability between them.

Modeling Control

There is a tremendous amount of diversity in how control sequences are expressed, represented and executed in buildings. Standardizing a language for control is far from a trivial task, and metadata models seeking to describe any aspect of this domain should be very careful. Luckily, there is already a candidate standard in this space.

The Control Description Language [8] is

> a declarative language that can be used to express control sequences using block-diagrams. It is designed in such a way that it can be used to conveniently specify building control sequences in a vendor-independent format, use them within whole building energy simulation, and translate them for use in building control systems.

[8] Control Description Language (CDL). `https://obc.lbl.gov/specification/cdl.html`, a

There are nascent implementations of standard sequences of operations in CDL available today [9]. While integration between CDL and other metadata models is still under-explored, it should be an eventual priority to develop a way of referencing *between* other metadata models and CDL-based specifications of control.

[9] ASHRAE G36 Sequences of Operation in CDL. `https://simulationresearch.lbl.gov/modelica/releases/v5.0.0/help/Buildings_Controls_OBC_ASHRAE_G36_PR1.html`, b

For example, consider a simple PID loop for temperature control (Figure 3). From CDL's perspective, the inputs (the current temperature readings from one or more sensors), outputs (heating coil valve positions, damper positions) and parameters/targets (temperature setpoints and/or deadbands) are all specified directly. Bereft of any context beyond an unstructured label, it is up to the developer or commissioner of the control sequence to ensure that the proper I/O points are linked to the CDL process. No where in CDL is there any conceptualization of "where" (in a logical *or* physical sense) those I/O points are and how they relate to the rest of the building.

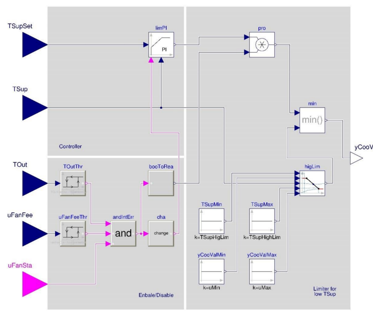

Figure 3: Controller for a cooling-coil valve, not temperature control, but you get the idea

Rather than shoehorning these descriptions into CDL, why not instead link CDL's representation of those I/O points to a richer, semantic representation in another metadata model?

Modeling Data

A metadata model for building data should facilitate the discovery and access of data to enable data-driven software. Data-driven software wants to leverage data sources in buildings — sensors, setpoints, statuses, alarms, etc. — to enact change or achieve insight into the operation, health and performance of the building.

An effective design of a metadata model for building data should prioritize the abstraction presented to the developer. In essence: simple data tasks should be straightforward, and unique or complicated data tasks should be possible. This will likely require abstracting away much of the complexity inherent to the building.

Such abstractions are already present in both literature and practice in the form of context-dependent but generally understood point labels. ASHRAE's Guideline 36 standard contains lists of required points for families of control sequences and FDD rules. The point names, such as "VAV Box Damper Position" and "Occupancy Sensor", are never formally or explicitly defined anywhere in the standard, yet they can be understood by a controls engineer within a specific context — the building they happen to be working on. Over time, point naming conventions and fixed name lists helped to standardize the technical understanding of a particular concept (Figure 4); however, the lack of standardization and structured information among such conventions prevented interoperability.

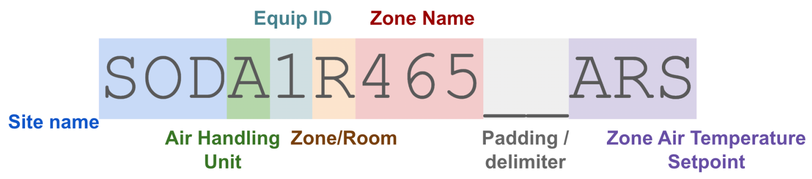

Figure 4: A sample point naming convention for a real building

Metadata efforts Brick and Haystack were both born out of a need to standardize and structure the semantic information informally encoded in BMS point labels. Where they differ is in how they approached the abstraction presented to the developer: Haystack chose to emphasize an intuitive, easy-to-use interface with a great deal of flexibility; Brick chose to emphasize standardizing the semantics of data sources and their context. These two abstractions are not mutually exclusive; they can, and should, be combined by taking the strengths of each to complement the weakness of the other.

I am eliding a discussion of the respective designs of Brick and Haystack and their respective strengths and weaknesses. I refer the reader to Fierro et al. [2019] and Fierro [2021] for a detailed discussion.

Figure 5 illustrates one possible vision. Using techniques established in Fierro et al. [2019], we can establish a mapping between Brick concepts and sets of Haystack tags and use that mapping to *serve Haystack tags directly off of a Brick model*. This has three benefits:

- Brick's concept definitions and formal semantics enable consistency in Haystack modeling. Community conventions such as the Utah tagging document and more recent standardization efforts (Haystack 4 protos) can be formalized using Brick and incorporated into linters or other verifying software.
- Existing Haystack software – backend, frontend and applications – can work on top of Brick models with little-to-no modification. Existing Haystack models can even be "retconned" into a Brick model to ensure correctness and consistency.

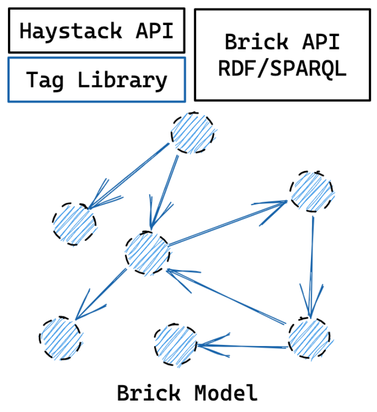

Figure 5: Use of a "tag library" exposes tags to a Haystack API implementation while directly from a Brick RDF graph.

- Because they are both built on RDF and SHACL, it will be easier to translate 223P to Brick than it will be to translate 223P to Haystack. Serving Haystack over a Brick model allows Haystack to be indirectly interoperable with 223P

I have already begun prototyping this idea using mapping tables like Table reftab:mapping-example which get compiled into SHACL rules which can be evaluated by any open-source or commercial SHACL engine which supports SHACL-AF.

Brick Concept	Haystack Tags
brick:Air_Temperature_Sensor	{air, temp, sensor, point}
brick:VAV	{vav, equip}

Table 1: Tabular mapping between Brick classes and Haystack tag sets

Key to this approach is treating Brick classes as the *canonical definition of these concepts*. The Brick concept organization will need to be patched with concepts and protos from Haystack and likely sustain some light reorganization to account for organizational differences.

One example is that Haystack defines an FCU as a subclass of AHU; Brick defines an FCU as a subclass of Terminal Unit — not an AHU.

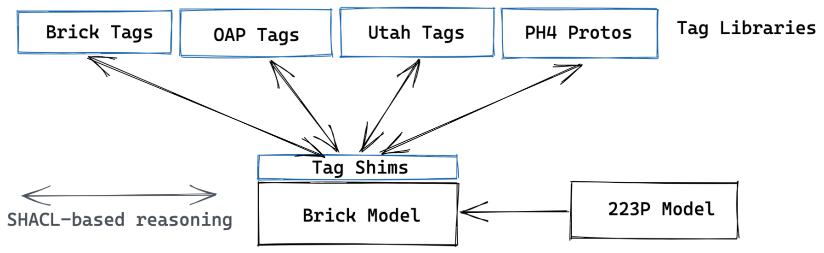

Figure 6: Illustrating representing the same metadata across the Ontology Alignment Project tags, Haystack 4 protos, Utah tag sets, and Brick's existing tags. This is all handled automatically.

Figure 6 illustrates how the SHACL-based tag shim I have developed allows Haystack metadata to be translated between different tag interpretations using Brick as the intermediary.

Modeling Topology and Composition

Even though Brick and Haystack both define many of the concepts named in point lists and application specifications, they lack the detailed and high-fidelity descriptions of buildings and building subsystems that are required for certain advanced use cases. Much of this detail is available through AEC models such as IFC and Revit but these are (a) vary widely in quality and consistency, despite the existence of IFC MVD and other specification languages[10], and (b) are awkward to create, extend and interact with.

[10] I once worked with an IFC model where all of the lighting fixtures were modeled as toilets.

ASHRAE's proposed 223P model is well-suited to addressing these issues. It has explicit conceptualizations of the connections and connection points between devices, allowing software to reason about the complex topology within and between devices in buildings. Like Brick, its modeling of properties and other data in the building is based on QUDT [11]. 223P's level of detail makes it uniquely positioned to model complex building subsystems that are not cleanly or completely handled in Brick or Haystack: lead/lag configurations, split ducts, the position of sensors or other equipment within complex pipework, and so on.

[11] Quantity, Unit, Dimension and Type Ontology. `http://www.qudt.org/`

223P is a *necessarily complex model*; buildings are complicated, and 223P can model most if not all of that complexity. Brick and Haystack are abstractions of that model — they remove detail in favor of simplifying the model that the model consumer must understand and query. Brick and Haystack will not be able to support all of the applications that 223P is able to, but they will make it easier to author common applications that do not require all of the detail supplied by 223P

Abstraction is the simplifying removal of nonessential information.

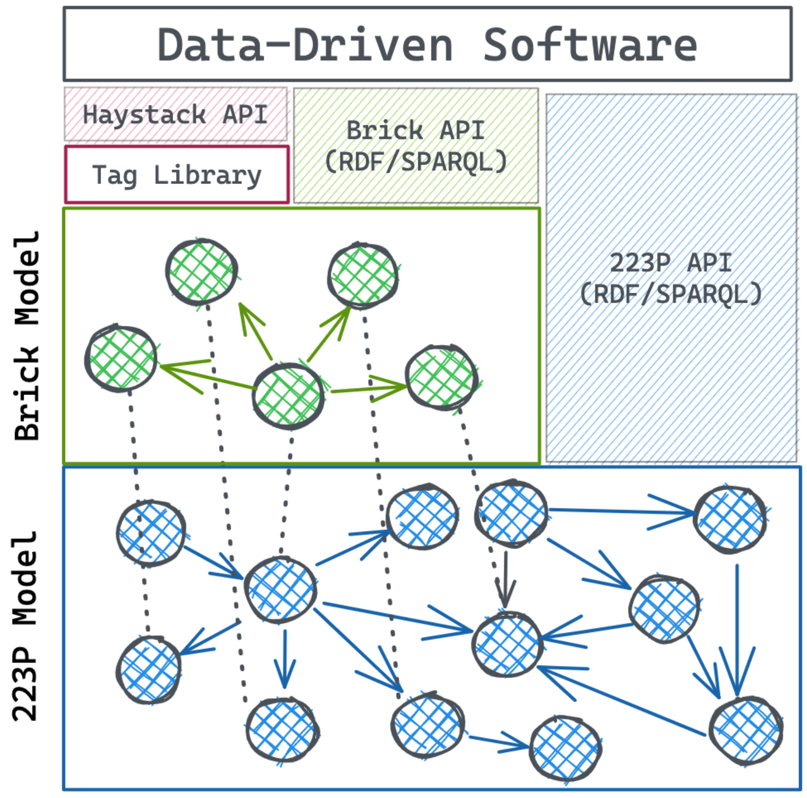

Figure 7: An ontology "stack" showing how Haystack, Brick and 223P can present different abstractions over the same building.

Figure 7 illustrates a hypothetical "stack" for how software could choose to interact with the same building through *either* Haystack, Brick or 223P depending on how much detail and fidelity is required. Just as Haystack tags are derivable from Brick models, Brick models could be derived from 223P models. A Brick model does not capture all of the detail in a 223P model, but it can provide a simplified interface — including intuitive contextualized point names like `Exhaust_Air_Temperature_Sensor` — that facilitates the development of data-driven software.

Modeling Network Representation

The final facet of the building is the network/instrumentation view. This is the structured representation of the control/monitoring network used to interact with the building: BACnet, LonTalk, OPC, Modbus, MQTT, etc. There is ongoing work on introducing an RDF-based model of BACnet networks[12], and this work could be extended to other protocols.

[12] This is being utilized in Brick already: `https://docs.brickschema.org/metadata/external-representations.html`

Modeling the network makes it possible for other digital representations of the building to refer to the metadata required to access a device, object, point, or data source.

Other Ontologies

I have left out of the above discussion the other building ontologies in the space: REC, BOT and others. For other domain-oriented ontologies (like REC), there are ongoing conversations for how to allocate definition of the necessary concepts among the existing communities. Brick and RealEstateCore are currently discussing how Brick could provide all of the equipment definitions and REC could provide all of the location definitions; this would eliminate the current duplicate modeling efforts between the two communities.

Upper ontologies like BOT[13] can be easily incorporated into domain ontologies. Interoperability is relatively easy to achieve here due to how generic the upper ontologies are, and is not a major concern.

[13] Not really an upper ontology, but more generic than a domain ontology

Conclusion

This document presents a possible vision for combining 223P, Brick and Project Haystack into a cohesive stack for supporting building applications. The stack emphasizes interoperability through clean interfaces between participating metadata models. By defining automated derivations between the metadata models, it is possible for software to choose the appropriate abstraction for its intended task.

References

Building Topology Ontology (BOT). `https://w3c-lbd-cg.github.io/bot/`.

Brick Schema. `https://brickschema.org`.

Control Description Language (CDL). `https://obc.lbl.gov/specification/cdl.html`, a.

ASHRAE G36 Sequences of Operation in CDL. `https://simulationresearch.lbl.gov/modelica/releases/v5.0.0/help/Buildings_Controls_OBC_ASHRAE_G36_PR1.html`, b.

Project Haystack. `https://project-haystack.org`.

Industry Foundation Classes (IFC). `https://www.buildingsmart.org/standards/bsi-standards/industry-foundation-classes/`.

Quantity, Unit, Dimension and Type Ontology. `http://www.qudt.org/`.

RealEstateCore. `https://realestatecore.io`.

Harry Bergmann, Cory Mosiman, Avijit Saha, Selam Haile, William Livingood, Steve Bushby, Gabe Fierro, Joel Bender, Michael Poplawski, Jessica Granderson, et al. Semantic interoperability to enable smart, grid-interactive efficient buildings. Technical report, Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States), 2020.

Gabe Fierro, Jason Koh, Yuvraj Agarwal, Rajesh K Gupta, and David E Culler. Beyond a house of sticks: Formalizing metadata tags with brick. In *Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pages 125–134, 2019.

Gabriel Tomas Fierro. *Self-Adapting Software for Cyberphysical Systems*. PhD thesis, University of California, Berkeley, 2021.