ASHRAE Standard 205P: Progress towards representation of performance data for HVAC&R equipment

Neal Kruis¹, Timothy McDowell², Charles S. Barnaby³, Tanaya Mankad¹
¹Big Ladder Software, Denver, CO, USA
²Thermal Energy System Specialists, Madison, WI, USA
³Independent Consultant, Moultonborough, NH, USA

Abstract
Sufficiently detailed HVAC equipment performance data are rarely available to building performance analysts. When data are published, there is little consistency among manufacturers, and the formats require error-prone manual manipulation before the data can be used with a specific simulation tool. ASHRAE Standard 205P (provisional), “Representation of Performance Data for HVAC&R and Other Facility Equipment”, defines common data models and serialization formats for facility equipment performance data, allowing automated exchange between equipment manufacturers and simulation models. The intent is for all manufacturers to write common-format data files (called “representations”) that can be processed by all building performance simulation software tools.

This paper describes a data modeling framework and an accompanying toolkit that facilitate the development of representation specifications and ease the adoption of the standard by both manufacturers and software developers.

Key Innovations
1. Standardization of equipment performance data files, called representations
2. Development of an efficient and robust data modeling framework for defining representation specifications
3. Development of a software toolkit to facilitate adoption of Standard 205P representations

Practical Implications
Building performance analysts will be able to model specific HVAC equipment products without substantial effort to convert or manipulate data into the forms used by their simulation software tools.

Introduction
Whole-building simulation models allow prediction of the fuel consumption and carbon emissions of HVAC equipment given information about how the equipment performs under the range of conditions encountered in the simulation. The lack of detailed equipment performance data in ready-to-use forms has hampered the application of building simulation models since their invention in the 1960s and 1970s. Data are often not available or not published in consistent formats, forcing simulation users to resort to time consuming (and error-prone) reformatting or to use generic performance characteristics that are not representative of the actual equipment.

ASHRAE Standard 205P (provisional), “Representation of Performance Data for HVAC&R and Other Facility Equipment”, addresses this deficiency by defining common data models and serialization formats for facility equipment performance data, allowing automated exchange between manufacturers and simulation models. The formats and procedures specified by Standard 205P are developed under ASHRAE and ANSI consensus processes with participation of equipment manufacturers, application software developers, and engineering practitioners. The intent is for all manufacturers to write common-format data files (called “representations”) that can be processed by all building performance simulation software tools. The focus of the standard is the format of the data. The methods of deriving data (e.g., experimentally or through computer models) is beyond the scope of the standard.

ASHRAE Standard 205P recently concluded its second publication public review. The standard itself has changed significantly since an early implementation of the standard was illustrated by Kruis and Barnaby (2018) before the first publication public review. The Standards Project Committee (SPC) responsible for Standard 205P is preparing a draft for a third, and hopefully final, full public review. The following sec-
Standard CBOR Serialization Data Format

Kruis and Barnaby (2018) discussed how Standard 205P focussed on a format-agnostic data model with the idea that the industry would “informally self-standardize around a specific file format”. Some reviewers of the draft standard noted that relying on informal self-standardization could ultimately defeat the objective of the standard. The SPC agreed to define a normative file format for the standard: Concise Binary Object Representation (CBOR) (Bormann, 2018).

CBOR is based on the same underlying data model as JSON (JavaScript Object Notation):

JSON is built on two structures: - A collection of name/value pairs. In various languages, this is realized as an object, record, struct, dictionary, hash table, keyed list, or associative array. - An ordered list of values. In most languages, this is realized as an array, vector, list, or sequence.

These are universal data structures. Virtually all modern programming languages support them in one form or another. It makes sense that a data format that is interchangeable with programming languages also be based on these structures. (JSON.org, 2018)

The major difference between JSON and CBOR is that CBOR serializes the file content in binary form where the data requires less space and can be processed faster. While the FlatBuffers (Google, 2017) format showed some advantages over CBOR (Kruis and Barnaby, 2018), the SPC ultimately decided to use CBOR due to its closer functionality to and compatibility with the more broadly used JSON format.

Normative JSON Schema

The data model descriptions for specific types of equipment (or “representation specifications”) in Standard 205P provide written guidance on how manufacturers should generate representations and how simulation software developers should read representations. However, this prescriptive approach was too informal to enforce compliance with the standard in actual implementations. A schema-based validation is now a normative component of the standard, and is required in order to verify that a given representation is compliant with the standard.

JSON Schema (Wright and Andrews, 2018) provides a “vocabulary to annotate and validate JSON [files]”. Because CBOR adheres to JSON’s underlying data model, JSON Schema can also be used to validate CBOR files. JSON Schema is well-established and currently in the process of preparing Internet-Drafts for standardization within the Internet Engineering Task Force (IETF).

Representation Specifications

At the time of this article, Standard 205P contains representation specifications for seven types of equipment (four more than were described by Kruis and Barnaby (2018)). Each equipment type is assigned a unique representation specification identifier (RSID) as shown in Table 1.

<table>
<thead>
<tr>
<th>RSID</th>
<th>Equipment Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>RS0001</td>
<td>Liquid-Cooled Chiller</td>
</tr>
<tr>
<td>RS0002</td>
<td>Unitary Cooling Air-Conditioning Equipment</td>
</tr>
<tr>
<td>RS0003</td>
<td>Fan Assembly</td>
</tr>
<tr>
<td>RS0004</td>
<td>Air-to-Air Direct Expansion Refrigerant Coil System</td>
</tr>
<tr>
<td>RS0005</td>
<td>Motor</td>
</tr>
<tr>
<td>RS0006</td>
<td>Electronic Motor Drive</td>
</tr>
<tr>
<td>RS0007</td>
<td>Mechanical Drive</td>
</tr>
</tbody>
</table>

A representation file has a well-defined hierarchical structure of data groups:

- RS00XX
  - Metadata
  - Description
    - ProductInformation
    - Ratings
    - Performance
    - PerformanceMap(s)

The “Metadata” data group contains information about a specific representation (e.g., a unique identifier, the corresponding schema version, a timestamp for when the data was created). The “Description” data group contains informative data such as the product manufacturer name and model number, as well as data related to specific ratings (e.g., AHRI). The “Performance” data group contains all data required to simulate the equipment under all operating conditions (not just the operating conditions used for ratings).

Barriers to Standard 205P Adoption

We have already alluded to the four key roles relevant to application of ASHRAE Standard 205P: Representation specification authors develop data models for a specific equipment type. Manufacturers provide representations to Building performance analysts, who then load representations into simulation software models through capabilities implemented by software developers.

While building performance analysts will benefit the most from the standard, by design they are the role...
that needs to concern itself the least with the actual implementation of the standard. It is the representation specification authors, manufacturers, and software developers who will need to become most intimately familiar with the data models and other requirements of Standard 205P.

The sections below discuss some of the barriers to adopting Standard 205P faced by each of the four user roles.

Authoring Representation Specifications

The main body of Standard 205P contains rules for authoring new representation specifications. The development of representation specifications should be accessible to both manufacturers and software developers. Therefore, the definitions of data models should be both human-editable and machine-readable. One advantage of machine-readable data model descriptions is they can be validated against a meta-schema—a programmatic description of the rules for creating a data model.

We also want to avoid the risk of inconsistencies posed by maintaining separate human-editable and machine-readable descriptions of the same data model. Many human-readable formats (e.g., word processors and spreadsheets) do not lend themselves to task automation by software. Many machine-readable formats have rigid, non-intuitive syntax for many non-programmer domain experts.

Publishing Representations

Manufacturers are accustomed to developing system selection software and product catalogues with a smaller subset of the data that is required to fully characterize the operation of the equipment for simulation. Furthermore, the data definitions used in Standard 205P may not align directly with their current data definitions. Building up the infrastructure to automatically produce 205P compliant representations will require some investment in adapting their current software and workflows. Some initial attempts to generate representations may be more manual, and it is important for tools to accommodate human manipulation of representation data.

Obtaining Representations

How manufacturers decide to distribute representations is largely their decision. The manufacturers reserve the right to limit access to their data as they see fit. While the topic of data access is explicitly outside the scope of Standard 205P, providing a means for building performance analysts to request and obtain representations will be critical to successful application of the standard.

Processing Representations

Software developers will also have to invest into the development of routines to read and process 205P compliant representations. Although the data models established by 205P may not map exactly to the current input data models of the simulation software, intermediate data model translation can be developed for specific software to encourage early adoption on by their users (for example, generating curve fits from 205P performance maps).

Ultimately, much of the functionality for reading and processing 205P representations will be similar across simulation tools. Shared libraries can be provided that contain these capabilities to minimize the amount of redundant effort among developers (for example, methods for multi-dimensional interpolation of performance maps can be provided by an open source library such as Btwxt (Big Ladder Software, 2020)).

A New Data Modeling Framework

With aforementioned barriers in mind, we developed a general engineering data modeling framework for organizing data elements and defining their relationships to one another. While many data models already exist in the area of building simulation and HVAC equipment (Green Building XML Schema Inc., 2021; Mosiman et al., 2020), they are tightly coupled to specific serialization formats and schema formats (e.g., XML) rather than the more abstract definitions of a data model. Industry Foundation Classes (IFC) (ISO, 2018) presents an exception with its EXPRESS data modeling language (ISO, 2004); however, the EXPRESS data modeling language introduces its own syntax that is not easily parsed and generally lacks tooling support unlike other established data modeling formats. There is an entire standard for generating XML schema from EXPRESS which notes conflicts and challenges in conforming EXPRESS and XML schemas (ISO, 2007). In the future, it is anticipated that Standard 205P data models can be integrated into broader frameworks such as buildingSMART data dictionaries (bSDD) (buildingSMART International, 2021) to support use in building information modeling (BIM) applications.

While SPC 205P decided to embrace the CBOR and JSON Schema, the underlying data model can support other serialization formats and/or schema formats without losing the relationships among data elements. Standard 205P data models can remain intact as serialization technologies come and go. As previously decided by the SPC, a real implementation of the Standard will have to settle on specific formats, but it will not be locked to those formats as technologies evolve over time.

Many data modeling languages are developed by software engineers and information scientists who adopt jargon that is not natural in the domain adopting this
data model: HVAC engineering and building scientists. One advantage of developing a new framework is we can establish a common vocabulary and introduce new data modeling attributes that help provide context to all roles involved in the data modeling and exchange process. For example, unit systems are essential to communicating engineering data and yet most data modeling languages do not formally support specifications of units.

Ultimately, for a data model to be usable for all user roles of the standard, there are three components that are required to facilitate its adoption that are very tightly coupled to the abstract data model itself: 1) A schema, 2) human-readable documentation, 3) source code for processing representations.

**JSON Schema**

A schema allows algorithmic verification of the correct assembly of a representation. We chose JSON Schema (Wright and Andrews, 2018) because it can be used with nearly any file format that is based on the same underlying data model (including CBOR).

JSON Schema supports almost all the validation needs for Standard 205P representation specifications, with a few minor exceptions. While additional properties, such as “units”, can be added to a JSON Schema, the validation will not do anything with that information. The snippet below illustrates the JSON schema for Motor (RS0005) performance.

```json
{
  "Performance": {
    "type": "object",
    "properties": {
      "maximum_power": {
        "description": "Maximum operational input power",
        "type": "number",
        "minimum": 0.0,
        "units": "W",
        "notes": "Operational limit set to prevent overheating"
      },
      "standby_power": {
        "description": "Power draw when not operating",
        "type": "number",
        "minimum": 0.0,
        "units": "W"
      },
      "number_of_poles": {
        "description": "Number of poles",
        "type": "integer",
        "exclusiveMinimum": 0,
        "multipleOf": 2
      },
      "drive_representation": {
        "description": "Corresponding drive representation",
        "$ref": "RS0006.schema.json#definitions/DriveRepresentation"
      },
      "performance_map": {
        "description": "Motor performance when operating",
        "$ref": "RS0005.schema.json#definitions/PerformanceMap",
        "notes": ["If no performance map is defined, ...", "***Informative note:*** This field may be ..."]
      }
    }
  }
}
```

**Source Schema**

The challenge with Standard 205P is that the data model content must be captured twice: once in the document itself, and once in the schema that is the basis of demonstrable compliance with the standard. However, maintaining the consistency of data model content is inefficient and error prone if performed manually.

While the JSON schema could itself be the common source, the syntax itself can be difficult to author for several reasons: - The keywords used in JSON Schema do not always align with the terms used in the underlying data model - Without special formatters, it can be difficult to track matching brackets - Elements are denoted as required separately from their definitions, which makes it hard to keep them updated

Instead, we established a common source for the data model content that is used to generate both the JSON Schema and the documentation. This source schema is described in YAML. YAML is a more human-readable/editable superset of JSON, and equally machine readable (with YAML parsers established for every major programming language). Like CBOR, YAML is based on the same underlying data model as JSON. YAML is a text-based file format, meaning that files can be version-controlled (i.e., changes can be algorithmically identified and displayed, and changes from multiple authors can be automatically merged).

The snippet below illustrates the source YAML schema for Motor (RS0005) performance. We rely on the reader’s judgement to decide whether the YAML format would easier to read and edit than the JSON format.

```
Performance:
  Object Type: "Data Group"
  Data Elements:
    maximum_power:
      Description: "Maximum operational input power"
      Data Type: "Numeric"
      Units: "W"
      Constraints: ">=0.0"
      Notes: "Operational limit set to prevent overheating"
      Required: True
    standby_power:
      Description: "Power draw when not operating"
      Data Type: "Numeric"
      Units: "W"
      Constraints: ">=0.0"
      Required: True
    number_of_poles:
      Description: "Number of poles"
      Data Type: "Integer"
      Required: True
```
Similar to CBOR, YAML can be validated against a JSON Schema. To this end, we have developed a JSON Schema meta-schema that ensures the YAML data models developed for Standard 205P are internally consistent and adhere to the normative naming and relational rules dictated in the standard. With these checks in place, not only will the documentation and the schema be consistent with each other, they will also be correctly formed according to the standard based on automated meta-validation – eliminating some of the manual effort of reviewing proposed data models for new representation specifications.

Documentation

Most document processing software lacks support for programmatic document creation. Microsoft Word is likely the most prominent word processing software in use today, but updating the data model tables in a Word document based on a schema is not a commonly supported function. Furthermore, some of the more sophisticated features of Word used for proper styling and cross-referencing are fragile and improper implementations are easily overlooked, especially for documents the size of an ASHRAE standard.

ASHRAE 205P embraces automatic Portable Document Format (PDF) generation through the LaTeX typesetting software (LaTeX Team, 2021). The basis of the document content is Markdown (Gruber and Swartz, 2021). The appeal of Markdown is that it is text-based (i.e., version-controllable), and effectively separates the document content from its styling. We use a python workflow to perform the following tasks (as illustrated in Figure 1):

1. Convert the data model content from the YAML source schema into Markdown tables
2. Insert these tables into templated Markdown files containing the primary text content of the Standard’s main body and each of its representation specifications
3. Convert the generated Markdown content into LaTeX source using Pandoc (MacFarlane, 2021)
4. Use LaTeX’s typesetting engine to compile the LaTeX source into a PDF document.

Table 2 illustrates the documented for Motor (RS0005) performance generated from the source schema.

Figure 1: Automated process used to generate the ASHRAE Standard 205P document PDF from YAML schema and templated Markdown

Programmatic Interface

There is yet a third instance of the data model that is required for application of Standard 205P: the internal source code of software that reads, stores, and processes compliant representations. Once again, it can be inefficient and error prone to maintain parity between the schema and corresponding source code. Every data element in a representations also needs to be read, stored, and processed in the application software.

The challenge with software source code is that every programming language has a different syntax, and it is not practical to generate source code for every programming language used in building performance simulation. Instead, we are targeting a lower level approach of generating source code in C++ with a C language library interface. Libraries are often written in C because C compilers generate efficient object code; programmers then create interfaces to the library so that the routines can be used from other computational languages like Fortran and higher-level languages like Python, Ruby, and Java.

The library compiled from the source code from the schema will provide consistent access to 1) the data elements in a representation, and 2) common functions for further verifying the representation (e.g., performance data contains the right amount of data according to its grid dimensions).

Bringing it all together

The data modeling framework described above is heavily automated to maximize efficiency and enforce consistency. The source code for much of this process is publicly hosted on GitHub (https://github.com/open205/schema-205) and configured with continuous integration (CI) to automatically 1) validate the generated source YAML schema against the meta-schema, 2) generate the JSON schema, Markdown documentation, and C library source code from
### Table 2: Motor (RS0005) performance data model documentation

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Data Type</th>
<th>Units</th>
<th>Constraints</th>
<th>Req</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>maximum_power</td>
<td>Maximum operational input power to the motor</td>
<td>Numeric</td>
<td>W</td>
<td>≥0.0</td>
<td>✓</td>
<td>Operational limit set to prevent overheating or overcurrent, not the rated name plate power</td>
</tr>
<tr>
<td>standby_power</td>
<td>Power draw when motor is not operating</td>
<td>Numeric</td>
<td>W</td>
<td>≥0.0</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>number_of_poles</td>
<td>Number of poles</td>
<td>Integer</td>
<td>↓0, %2</td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>drive_representation</td>
<td>The corresponding Standard 205 drive representation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>performance_map</td>
<td>Data group describing motor performance when operating</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- If no performance map is defined, the motor shall be assumed to transfer all electric power directly to mechanical shaft power
- **Informative note:** This field may be omitted for motor-driven equipment where motor efficiencies are incorporated into their performance data

---

the source YAML schema, 3) validate established example files against the generated JSON schema, 4) compile the standard PDF (this step happens in a private repository due to ASHRAE copyright restrictions), 5) compile and test the C/C++ library. This process happens every time a change is made to any of the files in the GitHub repository. For example, if someone were to modify the source YAML schema the CI would automatically execute each task to test that change across different platforms (i.e., Windows, Mac OS X, and Linux) and different environments (e.g., different versions of Python).

We illustrate the automation workflow employed by the data modeling framework in Figure 2.

![Figure 2: Data modeling framework automation workflow used for a) generating ASHRAE Standard 205P content and b) validating schemas and representations](image)

This data modeling framework is not specific to HVAC representations, in fact it is currently being evaluated as the basis for other data modeling efforts within the field of building performance simulation, including ASHRAE 229P, “Protocols for Evaluating Rule-set Implementation in Building Performance Modeling Software”, and IBPSA-USA’s Building Data Exchange Committee. Pandoc can also be used to convert the Markdown into HTML for more convenient web-based documentation.

**The Standard 205 Toolkit**

The data modeling framework we described is also leveraged in the development of a software toolkit. The purpose of the toolkit is to help facilitate adoption of Standard 205P. The initial functionality of the toolkit specifically focuses on supporting manufacturer adoption to ease their investment in developing their own internal infrastructure to comply with the standard. In the meantime, manufacturers need an effective way to create and share prototype representations. As a binary format, CBOR is challenging to work with in the absence of a software infrastructure to support it. The toolkit can generate templates to create representation data in spreadsheet format (XLSX). Spreadsheets are broadly used in engineering fields and is a logical way to share information among engineers. Although spreadsheet formats do not lend themselves to workflow automation, the toolkit provides a two-way translation between XLSX representations and CBOR. This translation also implicitly allows for the direct validation of XLSX representations against the normative JSON schema.

While CBOR is an efficient exchange format (both in terms of processing time and disk storage), working with other formats (in addition to XLSX) will be valuable for other early prototyping efforts. Because many file formats are based on the same underlying data model (e.g., CBOR, JSON, and YAML), translating between these formats is trivial to support. In fact, the official example files for the standard are all maintained in JSON format because it is easier to maintain in version control.
Beyond this initial functionality, several future developments are planned, including 1) translating Standard 205P representation data into software inputs (e.g., generating curve-fits from performance maps), so analysts can utilize representations while software developers implement official support, and 2) visualizing representation data, especially performance maps. Currently the toolkit is developed as a Python command line tool, with the idea that it can be further developed into a desktop application or a hosted web application with an API.

Discussion

Throughout the development of ASHRAE Standard 205P the SPC has focussed on honing the design and architecture of the standard and developing a supporting software infrastructure to facilitate adoption by users immediately after the standard is published. However, until there is an end-to-end demonstration of the standard in practice (i.e., a manufacturer generated representation that is used to determine equipment performance in building simulation software), much of the design results from anticipations of challenges that have not been encountered yet. For example, when the SPC inevitably needs to modify the schema, how will software tools adapt to the potential of representations conforming to different versions of the standard? To the extent possible, we want to sort out as many of these issues with the data model now to prevent the need for many compatibility breaking changes in the future.

In some ways, the largest challenges to publishing a standard like 205P is establishing effective communication between equipment manufacturers and simulation software developers. Historically, manufacturers’ perspective on equipment performance has been dominated by 1) certified ratings requirements referenced by energy codes and incentive programs, and 2) comfort design for extreme conditions. Standard 205P requires manufacturers to expand the amount of information about their equipment, from what they have historically divulged, to now cover all conditions their equipment may encounter. While manufacturers have valid reasons to withhold potentially proprietary data, they are coming to understand that sharing more data may be necessary in order for their equipment to be accurately represented in simulation models. This is increasingly true as more performance-based energy codes and incentive programs include pathways for simulation aided building design. Manufacturers will begin adopting Standard 205P as these codes and programs begin requiring Standard 205P representations for compliance.

A related challenge is simply in the difference in conceptual models between building engineers and software engineers. Where a building engineer might see a roof top unit, a software engineer sees a collection of an air-to-air direct expansion refrigerant coil system packaged in a fan assembly, a mechanical drive, a motor, and an electronic motor drive; all components with data models that can be reused to describe other types of packaged equipment (e.g., ERVs, mini-splits, and pumps). These different approaches to abstraction can make it difficult to communicate meaningful distinctions.

At the end of the day, both groups leave the discussion with greater understanding of each other’s domain and, ultimately, the result is better data models representing HVAC equipment. One notable case of this is the realization from manufacturers on the committee that chillers are often modeled as lossless, idealized refrigerant cycles. In reality, building engineers actually design equipment to cool mechanical rooms due to the heat gains from efficiency losses between the plug and the compressor. This example illustrates an impact that is not typically captured in building performance simulation practice. We expect that adoption of Standard 205P will reveal other shortcomings of the current equipment simulation models and enable greater distinction between competing market equipment.

Conclusion

ASHRAE Standard 205P enables the exchange of standardized HVAC&R equipment performance data from manufacturers to building performance simulation software applications. This exchange is supported through a carefully designed data modeling framework and facilitated through the capabilities of a software toolkit. Equipment representation specifications can be authored through an common source schema format that is both human-editable and machine-readable. The common source schema is the basis for automatic generation of a JSON schema for representation file validation, PDF data model documentation, and a C library interface for processing compliant representations and simulating the corresponding equipment in simulation software.

With ASHRAE Standard 205P, manufactures and practicing building performance modelers will have greater confidence that the equipment proposed for a building project is better represented in the models used to inform building design and operation.

References


buildingSMART Data Dictionary (bSDD).
https://www.buildingsmart.org/users/services/
buildingsmart-data-dictionary/.

https://google.github.io/flatbuffers/.

https://www.gbxml.org/.

https://daringfireball.net/projects/markdown/.


JSON.org (2018). Introducing JSON.
http://www.json.org/.


LaTeX Team (2021). The LaTeX Project.
https://www.latex-project.org/.

