
THE INNOVATION HUB
HUB
FOR AFFORDABLE HEATING AND COOLING
The Innovation Hub
for Affordable Heating and Cooling

API for DCH

API for DCH

V1.0 (Oct 2021)

Contents

Purpose of This Document 3
Senaps Database API 3
DCH Building Model API 3
SPARQL 3
ModelReference class fields 4
Query class fields 4
BRIQL 5
QueryInvocation class fields 5
Query class fields 5
NodeReference class fields 5
ModelReference class fields 6
QueryVar class fields 6
NodeVar class fields (subclass of QueryVar) 6
PropertyVar class fields (subclass of QueryVar) 7
VarFields 7
MatchType 7
Block class fields 7
PropertyPath class fields 8
Property class fields 8
Query response 9
QueryResponse class fields 9
ResponseValue 9
NullValue 9
NodeValue class fields 9
PropertyValue class fields 9
TypeValue class fields 9
PointInfo class fields 10

Purpose of This Document

This document aims at detailing how app developers and users can acquire building model and timeseries data from DCH.

There are two separate APIs offered by DCH to manage the data and the building models hosted on the platform. While the data hosted on Senaps can be managed using the Senaps API, the DCH API offers building model queries to perform tasks on the semantic building model.

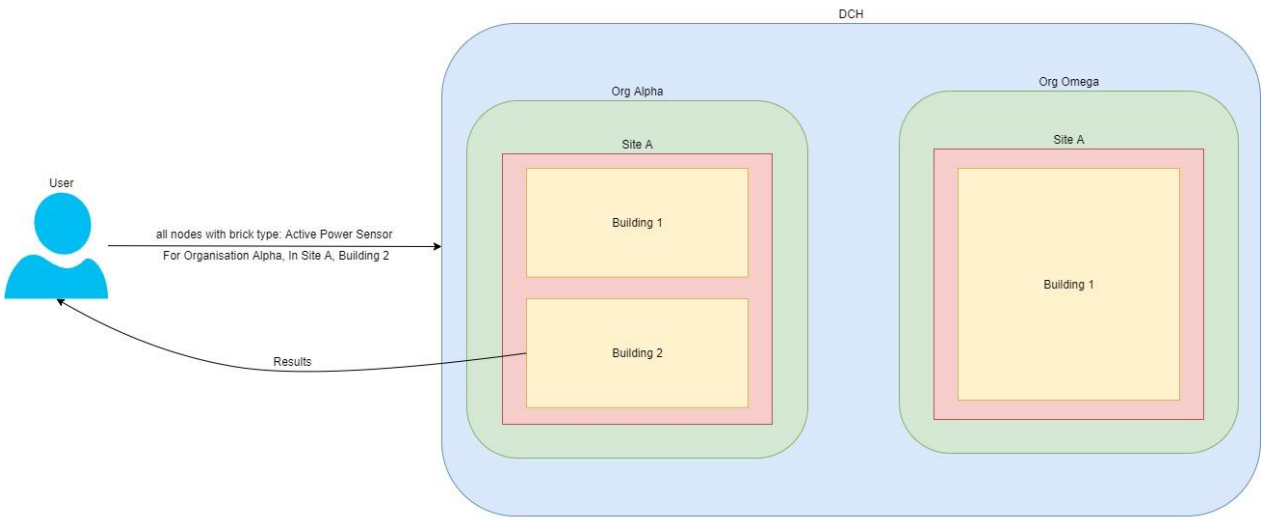
Senaps Database API

The datastreams of nodes in building models are hosted on Senaps which offers an extensive API interface for users. API examples can be found here: https://senaps.io/api-docs/#/.

DCH Building Model API

DCH supports two query languages to retrieve information from the building models, SPARQL and BRIQL. BRIQL is the recommended query language for DCH.

The contents of the queries are written in a JSON object which is the body of a RestAPI request. The RDF semantic models hosted on DCH use Brick Ontology, so the API can query building nodes based on their Brick classification. At this stage, DCH can provide a Postman environment with examples to enable users to use the DCH API.



SPARQL

SPARQL is a well-known query language used to obtain information from RDF databases. Using SPARQL you can query building nodes and the relationships between these nodes.

The API endpoint POST https://staging.dataclearinghouse/dch/v1/sparql/select accepts a JSON body with the following fields:

ModelReference class fields

Field	Type	Meaning
org_id	string	ID of the organisation
site_id	string	ID of the site
building_id	string (nullable)	ID of the building (null for sites)
model_id	String (nullable)	URI of the model (set only in responses)

Query class fields

Field	Type	Meaning
where_verbatim	query	SPARQL expression
distinct	boolean	If true, exclude duplicate results
input_variables	List of string	List of inputs
output_variables	List string	List of output variables as defined by user in query

Response mime:

When response_mime is "application/sparql-results+json", the response body is formatted according to the standard JSON schema for SPARQL results (see https://www.w3.org/TR/rdf-sparql-json-res/).

Alternatively, when response_mime is "application/sparql-results-dch+json", the response body is a terse custom JSON format unique to DCH. **TODO describe here.**

SPARQL example returning all Hot Water Pumps feeding the Air Handler Units:

```
{
  "models": [
    {
      "org_id": "Alpha",
      "site_id": "A",
      "building_id": "Building 1"
    }
  ],
  "query_specification": {
    "where_verbatim": "  ?hwp a brick:Hot_Water_Pump . ?hwp brick:feeds/rdf:type brick:Air_Handler_Unit . ",
    "distinct": true,
    "input_variables": [],
    "output_variables": [
      {"name":"hwp"}
    ],
    "bindings": {}
  },
  "response_mime": "application/sparql-results-dch+json"
}
```

BRIQL

BRIQL is a domain-specific language and protocol used for querying information on the building model nodes and relationships. Like SPARQL variables, BRIQL variables represent RDF nodes. Unlike SPARQL variables, labels and brick Points do not need to be requested as variables. Rather, a node is returned as an object representation that includes its label and any related brick Points. This means that BRIQL provides a more object-oriented API, with less overhead.

The BRIQL API endpoint POST https://staging.dataclearinghouse/dch/v1/query accepts a QueryInvocation object.

QueryInvocation class fields

Field	Type	Meaning
query_ref	string (nullable)	If set, a reference to a stored query (mutually exclusive with query_ref and describe).
query_def	**Query** (nullable)	If set, a literal Query to be run (mutually exclusive with query_ref and describe).
describe	**NodeReference** (nullable)	If set, query will exhaustively describe this node (mutually exclusive with query_ref and query_def).
models	List of **ModelReference**	List of ModelReferences against which the query is to be run.
bindings	List of **QueryVar**	List of variables to be pre-bound in queries.
limit_node_refs	Map of string to list of **NodeReference** (nullable)	Map of QueryVar name to a list of NodeReferences. This will use the SPARQL query construct VALUES (?varname) { (uri1) (uri2) etc}, to force the query to solve only for the provided nodes.

Query class fields

Field	Type	Meaning
ref	string (nullable)	Optional string to treat as a reference to this query
comment	string (nullable)	Optional string to describe query's purpose
variables	List of **QueryVar**	In select mode: list of variables used in the query Block.
query	**Block**	In select mode: the outermost block of the SPARQL SELECT query to be matched.
limit_node_refs	Map of string to list of **NodeReference** (nullable)	Optional map of QueryVar name to a list of NodeReferences. This will use the SPARQL query construct VALUES (?varname) { (uri1) (uri2) etc}, to force the query to solve only for the provided nodes.

NodeReference class fields

Field	Type	Meaning
model_ref	**ModelReference**	Reference to a model (site or building)
node_id	String (nullable)	If null: the NodeReference represents the model itself. Otherwise, the NodeReference represents the identified node within the model.

ModelReference class fields

Field	Type	Meaning
org_id	string	ID of the organisation
site_id	string	ID of the site
building_id	string (nullable)	ID of the building (null for sites)
model_id	String (nullable)	URI of the model (set only in responses)

QueryVar class fields

This is an abstract class; subclasses are NodeVar, TypeVar and PropertyVar, used for matching instance nodes, brick types, and brick properties respectively. Fields in this class are present in all subclasses.

Field	Type	Meaning
name	string	Name of the variable. Do not prefix with $ or ?.
comment	string	Comment about variable's purpose
input	boolean (nullable)	If true, this variable is a bindable input
output	boolean (nullable)	If true, this variable will be included in results
nullable	boolean (nullable)	If true, this variable may be null in results
constraints	**Block** (nullable)	If set, apply this Block's triples as constraints on this QueryVar. This is especially important for nullable QueryVars, where all triples should be bundled in a single OPTIONAL Block (graph pattern), where constraints can not be defined in the outermost query Block. Any other Implied triples will be merged into this Block.

NodeVar class fields (subclass of QueryVar)

Field	Type	Meaning
bind	List of **VarFields** (nullable)	List of fields to bind in query invocations (if null/empty, default is [id]). Required if input field is true.
fetch	List of **VarFields** (nullable)	list of fields to fetch and return in results (if null/empty, default is [id]). Required if output is true. If 'pointinfo' is included, fetch_points must have at least one element.
org_id	string (nullable)	If these fields are set in a query or invocation, compose a node URI to be bound to the variable.
site_id	string	
building_id	string (nullable)	
node_id	string (nullable)	
brick_types	List of **MatchType** (nullable)	If set in a query or invocation, constrain this node variable to match at least one of the listed MatchTypes
fetch_points	List of **MatchType** (nullable)	If set, only return associated points that match at least one of the listed MatchTypes. This field is required to be set if the fetch field includes pointinfo
filter_on	list of **VarFields** (nullable)	If set and filter_string is set, filter string is applied to these fields.

filter_string	string (nullable)	If set and filter_on is set, this string (expected to be a a SPARQL filter) is applied to fields listed in filter_on.

PropertyVar class fields (subclass of QueryVar)

Field	Type	Meaning
property	string (nullable)	This value, if bound in a query, will constrain the brick property of a variable. In an output variable, this will be bound to a Brick property for each solution. Possible values are: feeds, isFedBy, hasPart, isPartOf, hasLocation, isLocationOf, hasPoint, isPointOf.

VarFields

VarFields is an enumeration, represented in JSON with a string equal to one of the following

String value	Meaning
model_id	The ID of the model to which the node belongs
id	The ID of the node
type	The Brick type of the node
hypernym	The hypernym of the node's type (I.e. Location, Equipment, Zone or Point)
pointinfo	The IDs and streams IDs of points attached to the node
streams	Streams IDs of points attached to the node
label	The label of the node

MatchType

Either tags must be set, or both match and type must be set.

Field	Type	Meaning
match	string (nullable)	Specify the kind of match to perform on type field: • isa (match this Brick type or its child types) • equals (match this Brick type exactly) • parent (match only child types of this type) • hypernym (match this hypernym)
type	string (nullable)	Match this Brick type. Requires match is 'equals', 'hypernym', 'isa', or 'parent'. Mutually exclusive with tags. This field expects only the fragment component of the type's URI without prefix or delimeters (eg "Room", not "brick:Room").
tags	List of string (nullable)	If set and match is "tags", match any Brick type which has all of these tags

Block class fields

At run time, queries attempt to find solutions that match the nested structure of query elements. The fundamental unit is the Block (equivalent to a graph pattern in SPARQL terminology). Each query contains at least one block. Each block contains zero or more paths, and zero or more nested blocks. Internally, these are converted to valid SPARQL syntax at query run time. Paths are converted to triples patterns first (in specified order), followed by conversion of blocks (again, in specified order).

Field	Type	Meaning
comment	string	Comment about block's purpose.
paths	List of **PropertyPath** (nullable)	If set, these paths must all be matched by this block.
nested	List of **Block**(nullable)	If set, nest these Blocks inside this Block.
subquery	List of **Query** (nullable)	If set, add these Queries as SPARQL subquery SELECT blocks. NOT IMPLEMENTED YET
logic	string (nullable)	Either "and" (assumed default) or "or", indicating the matching logic to be applied to the elements enclosed by this block (all match, or any match respectively).
optional	boolean (nullable)	If true, this block's constraints are matched optionally. Defaults to false.
model	**ModelReference** (nullable)	If set, apply this block's constraints to the given model (using SPARQL's GRAPH {...} syntax).

PropertyPath class fields

A property path expresses a chain of edges between one model node and another. A property path matches one subject node, one or more predicates (properties), and an object node.

Field	Type	Meaning
from_ref	string	The SPARQL subject node (must be the name of a variable in the query definition)
properties	Non-empty list of **Property**	One or more Property, representing the chain of predicates between subject and object nodes.
to_ref	string	The SPARQL object node. Must be the name of a variable in the query definition.

Property class fields

Field	Type	Meaning
property	string (nullable)	If set, this Property is the named Brick property (mutually exclusive with the variable and or field)
variable	string (nullable)	If set, this Property is the named variable (mutually exclusive with the property and or field)
or	List of **Property** (nullable)	If set, this Property's position in the matched property path may take any one of the named Brick Properties (mutually exclusive with the property and variable field)
min	null, 0 or 1	Min and max fields, if set, apply property path quantifiers to specify how many chained occurrences of this property are to be matched by the query. min==0 → z**ero or more** occurrences min==1 and max==null → **at least one** occurrence min==0 and max ==1 → **either zero or one** occurrence Other combinations are not supported.
max	null, or 1	

Query response

For a select query, the response is a QueryResponse object. This

QueryResponse class fields

Field	Type	Meaning
models	List of ModelReference	The models which were included in the query
variable_names	list of string	Names of output variables.
solution_nodes	map of string to {map of string to NodeValue}	Outer key: variable's name Inner key: node's full URI Inner value: a NodeValue describing the node
solution_table	List of {map of string to **ResponseValue**}	The outer list is a list of solutions (rows). Each solution is a map of variable name to a ResponseValue

ResponseValue

There are four subclasses of ResponseValue class: NullValue, NodeValue, PropertyValue, TypeValue.

NullValue

(no fields, represents an unbound output variable)

NodeValue class fields

This represents node values bound to variables in solutions.

Field	Type	Meaning
model_index	integer	The model to which the node belongs (an index into the list in the models field of the QueryResponse)
id	string	The ID (local to the model) of the node, if requested
type	string	The brick type of the node, if requested
hypernym	string	The hypernym of the node's type (if requested)
pointinfo	List of **PointInfo**	List of PointInfo instances (if requested and any exist)
streams	List of string	List of Seanaps stream IDs (if 'streams' was requested in fetch field). This is only useful if the requested node is a Point.

PropertyValue class fields

This represents property values bound to variables in solutions.

Field	Type	Meaning
property	string	The property bound to the variable

TypeValue class fields

This represents type values bound to variables in solutions.

Field	Type	Meaning
type	string	A Brick type
hypernym	string	Hypernym of the type

PointInfo class fields

Field	Type	Meaning
type	string	The Brick type of the Point
point	string	The Point's node ID
streams	List of string	The Senaps stream ID(s), if any, belonging to the Point

BRIQL example returning all Hot Water Pumps feeding Air Handler Units:


```
{
 "models": [
  {
   "org_id": "Alpha",
   "site_id": "A",
   "building_id": "Building 1"
  }
 ],
 "query_def": {
  "comment": "Find which HWPs are feedings which AHUs.",
  "mode": "select",
  "variables": [
   {
    "var_type": "node",
    "name": "hwp",
    "output": true,
    "fetch": [
     "id"
    ],
    "brick_types": [
      {
       "match":"isa", "type":"Hot_Water_Pump"
      }
    ]
   },
   {
    "var_type": "node",
    "name": "ahu",
    "output": true,
    "fetch": [
     "id"
    ],
    "brick_types": [
      {
       "match":"isa", "type":"Air_Handler_Unit"
```

```
        }
      ]
    }
  ],
  "query":{
    "paths":[
      {
        "from_ref":"hwp",
        "to_ref":"ahu",
        "properties":[{"property":"feeds"}]
      }
    ]
  }
 }
}
```