2. Background

BHoM is an open source collaborative computational framework that integrates many concepts across existing AEC languages and platforms [8] [14]. BHoM consists of (1) object models (oM), (2) engines that operate on data, (3) adapters that map and translate data and (4) user interfaces to manipulate data. Currently, BHoM has over 1.200 object models with an extendable data dictionary and adapters to over 30 different software. It also allows using customized object models. While OOP classes implement or inherit implementations of methods, RDF does not have methods. But BHoM's interpretation of OOP is closer to an "ontological" approach than the "classical" OOP paradigm [15]. In the following sections, we first discuss BHoM's approach to OOP and afterwards compare BHoM with the technological standards set by the W3C and the LBD-CG.

2.1. BHoM 's Interpretation of OOP

At its core, BHoM's object model is a set of C# a programming language types (classes, interfaces, enums, etc.) and methods. But BHoM takes an approach that often departs from conventional OOP principles by: (1) Separating functionality from types and (2) focusing on ontologically meaningful interfaces.

First and foremost, BHoM imposes that all functionality is separated from the types. In other words, BHoM types have attributes but no methods. All functionality applicable to the oM types is isolated, and it is primarily grouped in C# projects called *Engines*. Similarly to oM projects, these groups target a specific domain, use specific namespaces and are suffixed with "*Engine*", e.g. methods for Structural Engineering are placed in the *Structure_Engine* under namespaces starting with *BH.Engine.Structure*. This separation makes BHoM similar to a virtual object database like those used to convert objects to relational databases[3].

Further, the development of BHoM types strictly follows the composition over inheritance principle. There are numerous interfaces and interface implementations, with each interface holding not only a contractual agreement with implementing types but also an ontological meaning.

2.2. Comparison of BHoM and Semantic Web standards

Whereas BHoM does not apply the standards set by the W3C, LBD community group, and developed through the Linked Data for Architecture and Construction (LDAC) workshops, conversion is still possible since they differ mainly in their technical implementations. The following subsections compare BHoM to Semantic Web standards regarding their identifiers, data models, data schemas, data exchange formats, and querying methods (see Table 1).

Identifiers. A noticeable difference between the two approaches is the way they identify things. Whereas in the Semantic Web, all nodes and edges (except for literals, e.g. strings, dates, etc.) are identified using a dereferenceable Uniform Resource Identifier (URI) [12], BHoM uses Globally Unique IDentifier (GUID) to identify the object models. Linked Data of the Semantic Web relies on HTTP URLs as identifiers of things [16]. But BHoM conventions also establish that every oM type and every piece of functionality is separated into different files. Because each file is committed to a public Git repo, each oM type is associated with a Unique Resource Identifier (URI) in the form of an HTTP address.

Database Model. While the Semantic Web uses a graph-based data model [17], BHoM relies on an object-oriented data model. RDF is a standard graph-based data model for data interchange on the internet [10] and consists only of 'triples': subject, predicate and object [12], whereas BHoM does not include explicit classes that describe the relationships between objects.

[3] Database Object-Relational Mappers (ORMs)