JÖNKÖPING UNIVERSITY
School of Engineering

Potential for Automating Reference Designation within the CoClass Framework

Somayyeh Hazem

&

Siby Susan Varghese

MASTER THESIS 2024

Master in Built Environment with a specialization
SUSTAINABLE BUILDING INFORMATION MANAGEMENT

This thesis work has been carried out at the School of Engineering in Jönköping in the subject area Product Development - Sustainable Building Information Management. The work is a part of the Master of Science with a major Built Environment, specialization in Sustainable Building Information Management.

Examiner: Ibrahim Yitmen

Supervisor: Peter Johansson

Scope: 30 credits

Date:2024/06/20

ADDRESS: *School of Engineering, P.O Box 1026, SE-551 11 Jönköping, Sweden*
VISIT: *Gjuterigatan 5, Campus, Building E*
PHONE: *+46 (0)36 10 10 00*
WEB: *www.ju.se*

Potential for Automating Reference Designation within the CoClass Framework

Somayyeh Hazem and Siby Susan Varghese

Jönköping University, Jönköping 551 11, Sweden
haso21aw@student.ju.se
vasi22yl@student.ju.se

Abstract:
In construction projects, effective communication and categorization are vital. CoClass and the Reference Designation System (RDS) provide clear frameworks to facilitate this. CoClass is a classification system created to uniformly describe construction systems, aiming to avoid misunderstandings and ensure precise representation. RDS is an international naming convention for labelling systems and their elements. Reference Designation (RD), the outcome of RDS, is a unique identifier that is both human and machine readable. To access or reuse this data in the future, it can be published on the web. Despite the availability of modern classification systems for years, many companies stick to their old classification systems due to the significant time and cost required for upgrading. Therefore, this study aims to explore the automation of RD generation in construction projects utilizing CoClass and RDS. Additionally, it seeks to enhance data accessibility and integration by generating URIs for RDs using ontology. The objective is to demonstrate the potential for cost and time savings through automation. A case study investigating six building components within an office space, extracted from a BIM model, is carried out. Leveraging IfcOpenShell and Dynamo scripts, CoClass parameters are added to BIM model and used to automate RDs. The BIM data, structured as a knowledge graph, which then supported the development of ontology. The study results demonstrate successful partial automation of RDs and RD based URIs, showcasing the potential for efficient data representation and exploration in Semantic Web applications. The study concludes with recommendations for future research and the importance of Automating RDs within the CoClass Framework.

Keywords: CoClass, Reference Designation System (RDS), Knowledge Graphs, Semantic Web and Linked-Data, Automation

1 Introduction

In construction projects, good communication and effective categorization are essential [1]. Classification systems in the Architecture, Engineering, Construction and Operations (AECO) industry have been developed to standardize building data with consistent terminology and semantics in an object-oriented manner, thereby unifying data exchange methods throughout the asset lifecycle [2]. CoClass serves as a classification system designed to collectively describe construction systems, aiming to prevent misunderstandings and ensure accurate representation [1]. “This may be done simply by naming the systems, but as complexity grows there is a need for specific identifiers of systems and their elements. The identification ensures that the elements of system of interest can be recognized across various models and documents by both stakeholders and IT systems” [3]. It is stated that the Coclass system helps to create structures based on unique codes, properties, and activities, and can be used across different disciplines [2]. Implementing CoClass will be valuable throughout the entire project lifecycle. It is intended to offer information on cost estimation and product delivery. Managing information about changes may be the most crucial aspect of CoClass implementation [1]. The emphasis is on ensuring a seamless flow of data, improving efficiency in data exchange, minimizing rework, and enabling efficient data reuse. “When there is a method where every actor can understand the information, it will help to improve communication” [4].

Research has identified limitations in the previous BSAB 96 classification, highlighting a need for improvement [1]. As a response to this demand, CoClass is envisioned to serve as an enhanced classification system specifically tailored for the construction industry in Sweden. Furthermore, the Reference ID in CoClass gives a coding system considering function, product, location and type. This enhancement will greatly aid the facility management stage of the project as it empowers the operations manager to efficiently schedule maintenance tasks and replace components with greater flexibility [1].

According to SS-EN/IEC 81346, The Reference Designation System (RDS) is a naming convention helps for the identification of elements. A Reference Designation (RD), an ID is what results from RDS [3]. Even though many unique ID concepts like global unique identifier (GUID), global trade item number (GTIN), or in the form of IT ID values are available, all these are machine readable [5]. “The RD creates a link among various models of the system(s) which are designed and engineered by a wide range of actors, and thereby creates both a common language among humans and among various computer systems" [3]. Thus, it helps to generate both human and machine-readable coding systems to classify objects.

However, it is required to improve the way data is represented, shared and utilized among different disciplines for better collaboration. By representing the RD in Resource Description Framework (RDF) and Web Ontology Language (OWL), it facilitates the implementation of the Semantic Web (SW) which in turns enhances the interoperability between diverse data sources. Also, Linked Data (LD) principles advocate for publishing structured data on the web using unique URIs and establishing links between related resources and thus it facilitates SW. This approach enables data to be interconnected and navigable, allowing users and machines to discover and explore related information across different datasets [6]. This paper will discuss the possibilities of the automatic generation of RD and their unique identifiers using RDS and URIs.

1.1 Theoretical background

CoClass

ISO 12006-2 and IEC 81346-1 have been introduced to the AECO industry to establish a unified framework for construction classification, aiding information exchange among various parties throughout the asset lifecycle. This framework underpins national standards like CoClass, the Swedish digital classification system. CoClass is largely based on the international ISO 12006-2 and IEC 81346-2 standard systems and aligns with Industry Foundation Classes (IFC), the global standard for data exchange in the building industry [2].

CoClass is the result of BSAB 2.0, a collaborative industry project that involved a significant portion of the stakeholders in the Swedish construction industry [7]. Svensk Byggtjänst (2017b) [8] describes CoClass as follows: "CoClass is the new digital classification system for all built environments in Sweden with the potential to permanently change the construction and management sector. CoClass is estimated to contribute to billion-dollar savings by providing a platform for more efficient communication and asset management" [9].

The assessment of information management using CoClass can be approached both horizontally and vertically. Vertical evaluation involves examining information from a hierarchical, object-oriented perspective. Horizontal evaluation, on the other hand, tracks each building component throughout its life cycle. By using CoClass as a standardized language and data structure, different stakeholders can monitor changes and deviations in information related to a specific building component, identified by a unique CoClass designation code [2]. According to Klas Ekberg, CoClass provides a comprehensive system for identifying objects throughout their lifecycle by assigning Reference Designations (RDs) that specify their function, assembly, location, and type.

Reference Designation System (RDS)

"Reference Designation System (RDS)" is an international "naming convention" designed for systems and their elements and a practical technique to underpin a common understanding among different disciplines" [3]. A Reference-ID serves as a unique identifier specifically crafted to unmistakably distinguish systems and their elements. This is achieved through a combination of well-defined relations, including part-of and type-of relations, along with various aspects such as function, product, location, and type.

RDS is based on SS-EN/IEC 81346 (2009). To mitigate the risk of misinterpretations, the reference ID incorporates symbols to distinctly define each code. These symbols are positioned before the code, also serving as separators to facilitate clear differentiation between them.

These symbols are:

= which explains the function,

\- which specifies the product,

\+ which specifies the location and,

% which explains the type of the class. [10]

Within the construction industry, the object's context holds significance, particularly in facilitating location-based analysis. While the RDS can be approached from diverse perspectives, such as project-, country-, or client-based views, adopting a common schema proves valuable in constructing a unified ID system for various purposes [5].

For example, to represent a door (a construction component) that belongs to a furnishing system (a technical system) within a wall system (a functional system) in a specific building, where its main function is to create a way into a room by being in the wall to provide an opening, the ID needs to merge those codes together,

DD.BHA.B.RD.QQC whereas,

DD represents Construction_Entity, **BHA** represents Space, **B** represents Wall_System, **RD** represents Wall_Construction and **QQC** represents Door.

To be more specific,

Reference ID	
+DD01-B01.RD01.QQC01%QQC10 whereas, **DD01** represents 'University Building with ID/type number 01' (Construction entity), **B** represents 'The wall system' & 01 represents the ID number-(Functional systems), **RD** represents 'the furnishing system for access' and 01 represents the ID number -(Constructive systems), **QQC01** is the door with ID number 01, **QQC10** represents the 'Door type: Hinged door'-(Components)	Simple example in where prefix signs are introduced to break up the ID code based on location (+), product (-) and type (%) part. [5]
+DD1001-B1001.RD3201.QQC01%QQC10 whereas, **DD1001** represents 'University Building with ID and type' (Construction entity), **B** represents 'The wall system' & 10 represents the type 'The exterior wall system' and 01 represents the ID number-(Functional systems), **RD** represents 'the furnishing system for access', **32** represents the type 'Space Access System' and 01 represents the ID number-(Constructive systems), **QQC01** is the door number 01, **QQC10** represents the 'Door type:Hinged door'-(Components)	Simple example where type and ID numbering values are introduced along with each classification code. [5] "2+2 numbering schema is used in where first 2 numbers refer to type and 2 following numbers to an ID value of that type" (Eckerberg, 2019)
+BF2001.DD1001 **-B1001.RD3201.QQC01%QQC10** whereas, **DD1001** represents 'University Building with ID and type' (Construction entity), **B** represents 'The wall system' & 10 represents the type 'The exterior wall system' and 01 represents the ID number-(Functional systems), **RD** represents 'the furnishing system for access', **32** represents the type 'Space Access System' and 01 represents the ID number-(Constructive systems), **QQC01** is the door number 01, **QQC10** represents the 'Door type:Hinged door'-(Components)	Various parts are broken down into multiple lines from the significant prefix. [5]
-B1001.RD3201.QQC01%QQC10/++BHA1001 **B** represents 'The wall system' & 10 represents the type 'The exterior wall system' and 01 represents the ID number-(Functional systems), **RD** represents 'the furnishing system for access', **32** represents the type 'Space Access System' and 01 represents the ID number-(Constructive systems), **QQC01** is the door number 01, **QQC10** represents the 'Door type:Hinged door'-(Components), and **BHA1001** represents the 'goods handling space' which is the location of the component.	Referencing to a door in some specific room (built space). "In some cases, it makes sense to show the component's connection to a specific location locally" [5]

Table.1: Examples of Reference designation system for a door (construction component) in different ways

Knowledge Graphs and ontology

The term knowledge graph has recently been frequently referenced in both research and business, typically in relation to Semantic Web technologies, linked data, large-scale data analytics, and cloud computing. [11]. "A knowledge graph (i) mainly describes real world entities and their interrelations, organized in a graph, (ii) defines possible classes and relations of entities in a schema, (iii) allows for potentially interrelating arbitrary entities with each other and (iv) covers various topical domains.” [12]

An ontology serves as a structured representation of real-world entities, offering a clear depiction of the concepts within a specific domain through defined classes, properties associated with these concepts, and constraints based on the properties described. [13]. Ontologies are incorporated into the W3C standard stack of the Semantic Web, while knowledge graphs utilize a graph-based data structure to store information. This approach is particularly useful in situations requiring the integration, organization, and extraction of value from multiple data sources on a large scale. [14]. A knowledge graph is a semantic structure comprising vertices (or nodes) and edges. Vertices depict concepts or entities, representing main categories of objects like molecules and proteins. An entity denotes a tangible object in the real world, such as an illness like cancer. Meanwhile, edges delineate semantic relationships between concepts or entities. [15].

Semantic web and Linked-Data

Semantic web and linked data technologies are increasingly influential and significant in the Architecture, Engineering, Construction, and Facility Management (AEC/FM) sector [16].

The Semantic Web, an expansion of the existing Web, goes beyond merely posting data online. Its essence lies in establishing connections, enabling both humans and machines to navigate through a network of data. As per the World Wide Web Consortium (W3C), the Semantic Web serves as a platform for sharing and reusing data across various applications, businesses, and communities [17]. Linked data, a key concept, outlines the preferred approach for publishing and linking structured data in Resource Description Framework (RDF) format on the internet [18].

Linked data operates on four fundamental principles [19]:

1) Employ Uniform Resource Identifiers (URIs) as identifiers for entities.

2) Utilize Hypertext Transfer Protocol (HTTP) URIs to facilitate easy lookup of these identifiers by users.

3) Furnish valuable information in response to URI queries, employing established standards such as RDF and SPARQL Protocol and RDF Query Language (SPARQL).

4) Incorporate links to additional URIs to enable further exploration and discovery of related entities.

In the realm of the Semantic Web, ontologies play a key role in disseminating and linking structured data on the Web, a concept commonly referred to as linked data [18]. These ontologies encapsulate knowledge within specific domains and foster semantic interoperability by forging connections with external data sources. They offer the potential to overcome prior constraints associated with the sharing and utilization of unstructured construction information. Consequently, a diverse array of concepts and methodologies have been explored and put into practice, including the representation of construction domain knowledge, the extraction or transformation of semantic data from BIM, and the integration with external data sources [20].

To make a bridge between the BIM, Semantic Web and Linked data, Lee et al. (2016) propose a framework as follows:

- “Develop an ontology for publishing data in a linked data principle.
- Extract the information from the BIM model and generate or convert into a machine-readable format.
- Convert the extracted BIM data into an RDF graph.
- Use SPARQL query to retrieve or modify the output data” [18].

Resource Description Framework (RDF)

According to Decker et al., 2000, RDF serves as a data model for presenting information, particularly metadata, about web resources. Metadata provides details about other data. In RDF, the data model expresses statements about resources through subject, predicate, and object expressions, commonly referred to as "triples." To designate resources, RDF employs Uniform Resource Identifiers (URIs) and URI references (URIRefs) [21].

“The Triple patterns are identified by the following format:

- Subjects can be either URIs or Blank nodes

- Predicates are mostly URI

- Objects can be URIs, Blank nodes or literals.

These triple patterns from different data can be linked together and form an RDF graph (Hitzler, 2011)” [22].

By employing HTTP URIs as universally unique identifiers for both data elements and vocabulary terms, the RDF data model is inherently crafted for global usage, allowing anyone to reference any entity. Clients could retrieve supplementary information by looking up any URI within an RDF graph online. Consequently, every RDF triple contributes to the worldwide network of data, offering a foundation for exploring this expansive data realm. Additionally, linked data publishers may offer RDF dataset dumps for local data replication, along with SPARQL endpoints for direct querying of the data [20].

Automation

The push for digitalization in Sweden extends to various aspects of the planning and building process, including the automation of building permits. As part of this effort, there's a need to update the Swedish building specification to accommodate new requirements [23].

Utilizing a classification system for BIM provides advantages such as enabling a common definition of terms, specifying the elements to include in BIM, and facilitating automated conversion. [24]

Automating the code according to CoClass is something that should be pursued, at least to some extent. One problem that arose during the study is that implementing CoClass in a BIM model is time-consuming. This also means that the cost of such implementation is relatively high. If it were possible to automatically generate CoClass codes, it would save both time and money, thus simplifying the implementation of CoClass in the construction industry [25]. The theoretical investigation done in the paper “Analys och tillämpning av CoClass “demonstrates that certain types of code are impossible to generate. It also shows that certain parts of the code can theoretically be generated. If this aspect could be resolved, work with CoClass would be faster and smoother. It would also become easier and more attractive for companies in the industry to actively start using CoClass in their projects [25].

The importance of automation in urban planning processes, particularly in the context of digitalization, is underscored by Boverket. Municipalities are required to create new building maps, and significant cost savings are linked with digitalized procedures. With automation leading the way, these savings could potentially reach up to 75% by leveraging digital data streams for new building maps. According to Boverket, automation is crucial in Building Information Modeling (BIM), facilitating precise design tasks and accurate georeferencing. It also enhances life cycle analyses and streamlines material lists compilation from BIM models. Standardized classifications play a critical role in automating data utilization across processes, improving efficiency and data utilization for project design and regulatory oversight [26].

1.2 Problem formulation

Even though classification systems have been available for many years, the adoption rate remains slow. Transitioning to a new system demands a significant investment of time and effort, dissuading many companies from considering an upgrade so, numerous companies continue to rely on their old systems developed like BSAB 96 [1]. According to the research, another significant finding is that automating the generation of CoClass codes would streamline its implementation in projects, reducing costs and increasing efficiency. Additionally, this automation would boost CoClass's appeal to industry companies, thereby heightening their interest in integrating it into their daily workflows [9]. The RD within the CoClass framework gives a unique identifier, which establishes connections among different models of systems designed and engineered by various actors, fostering a common language for both humans and computer systems [3]. This linkage significantly enhances collaboration efforts.

1.3 Scope and objectives

This study aims to explore the possibilities of automating RD generation in construction projects utilizing CoClass and RDS. Additionally, it seeks to enhance data accessibility and integration by generating URIs for RDs using ontology. The objective is to demonstrate the potential for cost and time savings through automation, thereby encouraging companies to adopt new classification systems.

Research questions:

1) How can Reference Designations be automatically obtained?
2) How can unique identifiers be automatically generated for these Reference Designations using ontology and knowledge graphs?

Following parts explains the methodology, which includes the case study. Subsequently, the results obtained from the case study will be discussed, followed by an in-depth analysis and interpretation in the discussion section. The study concludes with recommendations for future research and the importance of Automating RDs within the CoClass Framework.

2 Method

In this research, a case study was conducted. The method of developing solution consists of four main steps. The steps will be discussed in further detail in Figure 1.

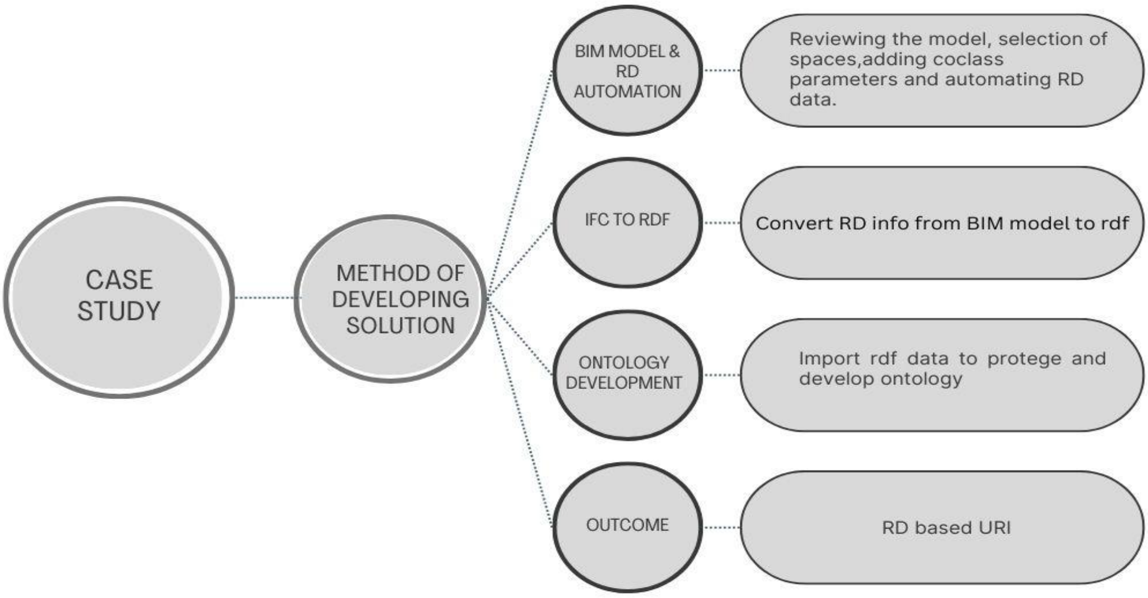


Fig.1: Methodology Framework

The study commenced with a comprehensive analysis of a three-story commercial building model which was provided by the School of Engineering, Jönköping University, focusing on an office room on the first floor. Within this space, six components, including a door, two windows, and three lighting fixtures, were selected for detailed examination.

Leveraging IfcOpenShell, a Python script was developed to extract CoClass classification parameters for each component, providing essential descriptors within the BIM framework. An automated Dynamo script was then developed to generate Reference Designations (RD) based on these parameters. To enrich the analysis further, the BIM data underwent transformation into RDF format using the IfcOpenShell Python library, facilitating the creation of a structured knowledge graph. This RDF data was then imported into Protégé, where an ontology was crafted which established a robust framework for data organization and querying.

Finally, a SPARQL query was formulated to probe the ontology, enabling validation and retrieval of specific component information such as attributes, relationships, and RD-based URIs. This holistic approach, integrating both ontology development and knowledge graph construction, enhanced the depth and efficiency of the study's analysis and data management processes.

3 Results

3.1 BIM Model Review and Component Selection:

A three-story BIM model was examined to locate a specific office room on the first floor. In this room, six key components were chosen for detailed analysis: one door, two windows, and three lighting fixtures, as illustrated in Figure 2.

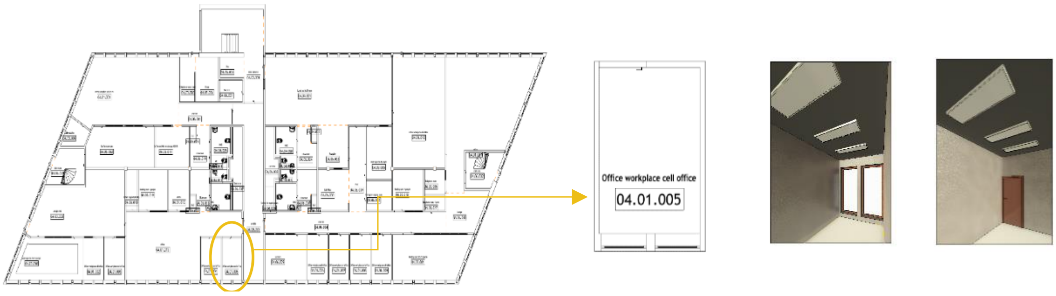


Fig.2: BIM Model and selected components

3.2 Automating RD

To add the CoClass classification parameters and their codes to the identified components, an Excel sheet was manually developed using the codes from CoClass Studio (https://byggtjanst.se/tjanster/coclass-studio). A Python script utilizing IfcOpenShell was then created to incorporate the CoClass classification parameters and their values into the IFC model for each of the six components. Additionally, as shown in Figure 3, a Dynamo script was developed to automate the generation of Reference Designations (RD) values based on the CoClass parameters. When the script is run, the RD values are automatically generated according to the provided CoClass parameters.

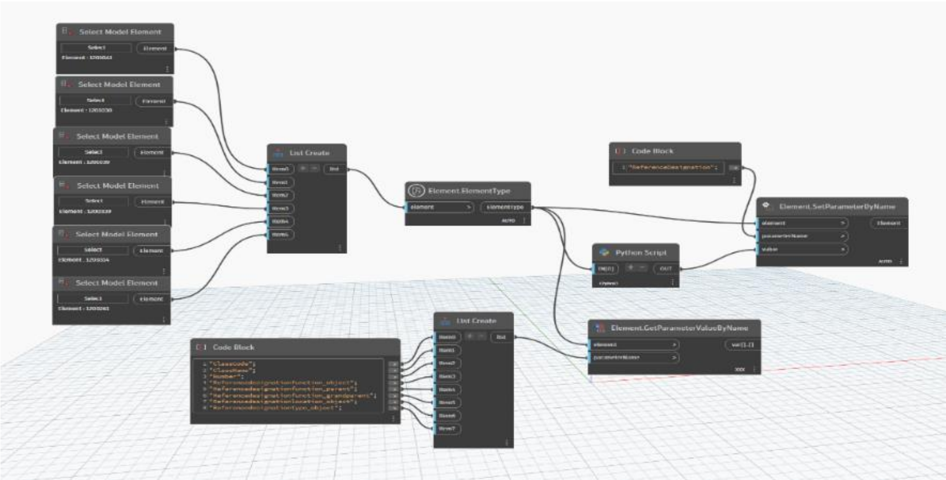
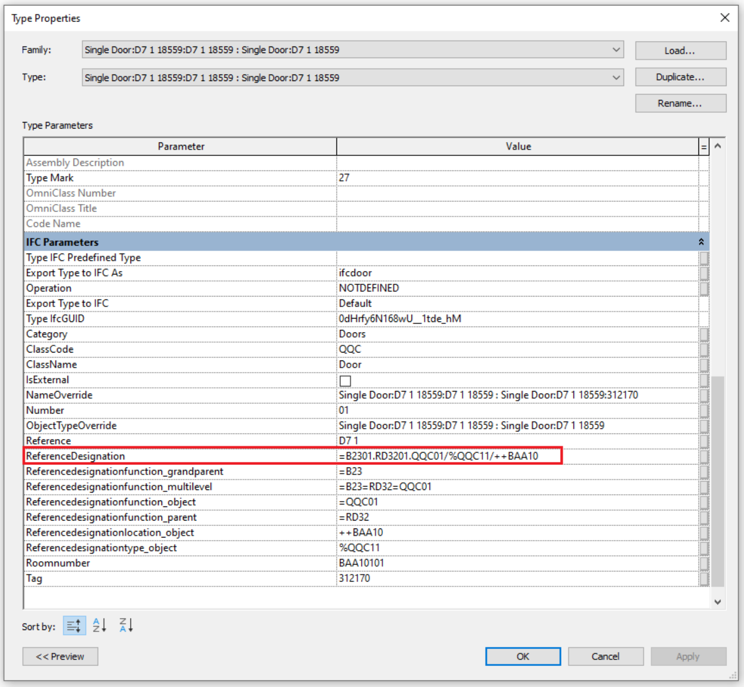


Fig.3: Dynamo script for automating RDs

Consequently, any changes to a CoClass parameter will automatically update the corresponding RD value (Figure 4).



Type Properties

Family: Single Door:D7 1 18559:D7 1 18559 : Single Door:D7 1 18559

Type: Single Door:D7 1 18559:D7 1 18559 : Single Door:D7 1 18559

Type Parameters

Parameter	Value
Assembly Description	
Type Mark	27
OmniClass Number	
OmniClass Title	
Code Name	
IFC Parameters	
Type IFC Predefined Type	
Export Type to IFC As	ifcdoor
Operation	NOTDEFINED
Export Type to IFC	Default
Type IfcGUID	0dHrfy6N168wU__1tde_hM
Category	Doors
ClassCode	QQC
ClassName	Door
IsExternal	☐
NameOverride	Single Door:D7 1 18559:D7 1 18559 : Single Door:D7 1 18559:312170
Number	01
ObjectTypeOverride	Single Door:D7 1 18559:D7 1 18559 : Single Door:D7 1 18559
Reference	D7 1
ReferenceDesignation	=B2301.RD3201.QQC01/%QQC11/++BAA10
Referencedesignationfunction_grandparent	=B23
Referencedesignationfunction_multilevel	=B23=RD32=QQC01
Referencedesignationfunction_object	=QQC01
Referencedesignationfunction_parent	=RD32
Referencedesignationlocation_object	++BAA10
Referencedesignationtype_object	%QQC11
Roomnumber	BAA10101
Tag	312170

3.3 Conversion to RDF Format

To facilitate data representation and exchange on the web, and to enable seamless integration with Semantic Web technologies, the Ifc data was converted into Resource Description Framework (RDF) format by using the IfcOpenShell Python library. The name space in the script was manually defined as http://CoClass_Classification/Ifc/ for the Coclass parameters in the model (Figure 5). This process resulted in an RDF format encompassing the entire IFC model data. Given the primary focus on CoClass parameters and Reference Designation (RD) values, a separate Python script was created to filter out only this specific data. The resulting RDF data contained the CoClass parameters, RD values, and their corresponding values, as illustrated in Figure 6.

```
import ifcopenshell
from rdflib import Graph, URIRef, Literal, Namespace
from rdflib.namespace import RDF, RDFS

ifc_file = ifcopenshell.open('Ifcwithcoclassproperties.ifc')

g = Graph()

ifc_ns = Namespace("https://CoClass_Classification/ifc/")
rdf_ns = Namespace("http://www.w3.org/1999/02/22-rdf-syntax-ns#")
rdfs_ns = Namespace("http://www.w3.org/2000/01/rdf-schema#")

for entity in ifc_file.by_type('IfcPropertySet'):

    property_set_name = entity.Name

    property_set_uri = URIRef(ifc_ns[property_set_name])

    g.add((property_set_uri, RDF.type, ifc_ns.PropertySet))
    g.add((property_set_uri, RDFS.label, Literal(property_set_name)))

    if hasattr(entity, 'HasProperties'):
        for property in entity.HasProperties:
            property_name = property.Name
            property_value = property.NominalValue

            property_uri = URIRef(ifc_ns[property_name])

            g.add((property_set_uri, ifc_ns.hasProperty, property_uri))
            g.add((property_uri, RDF.type, ifc_ns.Property))
            g.add((property_uri, RDFS.label, Literal(property_name)))
            g.add((property_uri, ifc_ns.hasValue, Literal(str(property_value))))

g.serialize(destination='Convertingto_RDF.rdf', format='xml')
```

```
<?xml version="1.0"?>
<rdf:RDF xmlns="http://www.w3.org/2002/07/owl#"
     xml:base="http://www.w3.org/2002/07/owl"
     xmlns:ns1="https://CoClass_Classification/ifc/"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:xml="http://www.w3.org/XML/1998/namespace"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
    <Ontology/>

 <!-- https://CoClass_Classification/ifc/Pset_CoClass_Classification -->

 <NamedIndividual rdf:about="https://CoClass_Classification/ifc/Pset_CoClass_Classification">
     <rdf:type rdf:resource="https://CoClass_Classification/ifc/PropertySet"/>
     <rdfs:label>Pset_CoClass_Classification</rdfs:label>
     <ns1:hasProperty rdf:resource="https://CoClass_Classification/ifc/ClassCode"/>
     <ns1:hasProperty rdf:resource="https://CoClass_Classification/ifc/ClassName"/>
     <ns1:hasProperty rdf:resource="https://CoClass_Classification/ifc/Number"/>
     <ns1:hasProperty rdf:resource="https://CoClass_Classification/ifc/ReferenceDesignation"/>
     <ns1:hasProperty rdf:resource="https://CoClass_Classification/ifc/Referencedesignationfunction_grandparent"/>
     <ns1:hasProperty rdf:resource="https://CoClass_Classification/ifc/Referencedesignationfunction_multilevel"/>
     <ns1:hasProperty rdf:resource="https://CoClass_Classification/ifc/Referencedesignationfunction_object"/>
     <ns1:hasProperty rdf:resource="https://CoClass_Classification/ifc/Referencedesignationfunction_parent"/>
     <ns1:hasProperty rdf:resource="https://CoClass_Classification/ifc/Referencedesignationlocation_object"/>
     <ns1:hasProperty rdf:resource="https://CoClass_Classification/ifc/Referencedesignationtype_object"/>
     <ns1:hasProperty rdf:resource="https://CoClass_Classification/ifc/Roomnumber"/>
 </NamedIndividual>

 <!-- https://CoClass_Classification/ifc/ReferenceDesignation -->

 <NamedIndividual rdf:about="https://CoClass_Classification/ifc/ReferenceDesignation">
     <rdf:type rdf:resource="https://CoClass_Classification/ifc/Property"/>
     <rdfs:label>ReferenceDesignation</rdfs:label>
     <ns1:hasValue>=B1001.RD3201.QQA01/%QQA60/++BAA10101</ns1:hasValue>
     <ns1:hasValue>=B1001.RD3201.QQA02/%QQA60/++BAA10101</ns1:hasValue>
     <ns1:hasValue>=B2301.RD3201.QQC01/%QQC11/++BAA10101</ns1:hasValue>
     <ns1:hasValue>=Q1101.HH1001.UAC01/%UAC12/++BAA10101</ns1:hasValue>
     <ns1:hasValue>=Q1101.HH1001.UAC02/%UAC12/++BAA10101</ns1:hasValue>
     <ns1:hasValue>=Q1101.HH1001.UAC03/%UAC12/++BAA10101</ns1:hasValue>
 </NamedIndividual>
</rdf:RDF>
```

Fig.5: Python script to convert Ifc data into RDF

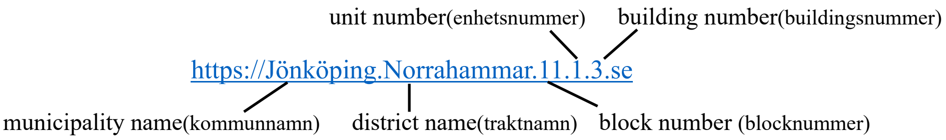
Fig.6: Ifc data in RDF format

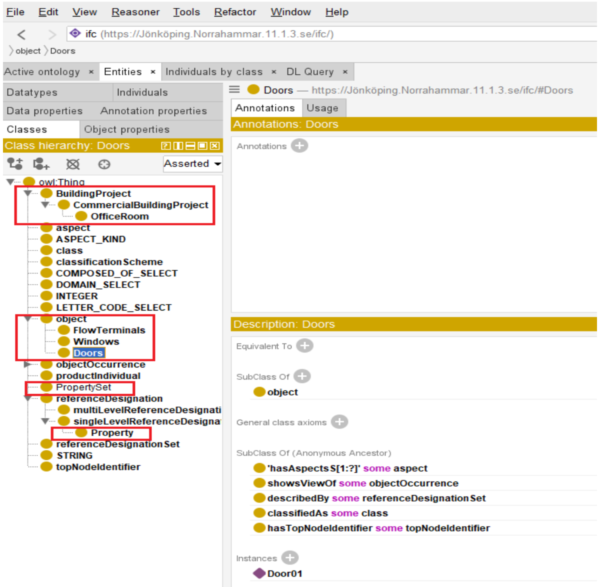
3.4 Ontology Development

The data developed as RDF constitutes a knowledge graph structure, which aids in extending the ontology for further analysis. By building upon this foundation, an ontology provided by Klas Eckerberg was developed and expanded to incorporate additional data and relationships in Protege as shown in Figure 7. The aim for developing the ontology was to automatically generate the unique identifiers regarding Reference Designation for each component.

As shown in Figure 8, when the RDF data was imported into Protégé, it resulted in an instance named "ReferenceDesignation" with `rdfs:label` and RD values in the annotations. To enable retrieval of RD-based URIs through SPARQL queries, the existing ontology needed to be further developed in alignment with the knowledge graph (RDF data) obtained from the annotations. During the ontology development, additional classes were incorporated, as highlighted in Figure 7. Domains and ranges were defined for both object properties and data properties, which were then assigned to the component instances, as illustrated in Figure 9.

To define a unique IRI, the property unit designation needed to be included, following the guidelines provided by Lantmäteriet (https://www.lantmateriet.se/en/maps/our-map-services/my-map/find-property-unit-designation-in-my-map/) to specify the location of the building to which the components belong to , as shown below.





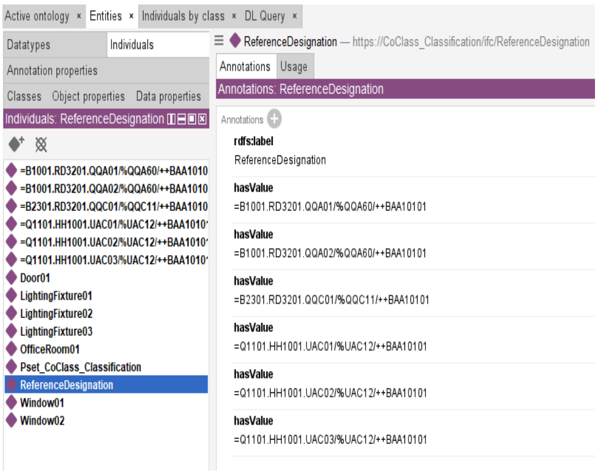
owl:Thing
- BuildingProject
  - CommercialBuildingProject
    - OfficeRoom
- aspect
- ASPECT_KIND
- class
- classificationScheme
- COMPOSED_OF_SELECT
- DOMAIN_SELECT
- INTEGER
- LETTER_CODE_SELECT
- object
  - FlowTerminals
  - Windows
  - Doors
- objectOccurrence
- productIndividual
- PropertySet
- referenceDesignation
  - multiLevelReferenceDesignati
  - singleLevelReferenceDesigna
    - Property
- referenceDesignationSet
- STRING
- topNodeIdentifier

SubClass Of: object

SubClass Of (Anonymous Ancestor):
'hasAspectsS[1:?]' some aspect
showsViewOf some objectOccurrence
describedBy some referenceDesignationSet
classifiedAs some class
hasTopNodeIdentifier some topNodeIdentifier

Instances: Door01

Fig.7: Class Hierarchy



=B1001.RD3201.QQA01/%QQA60/++BAA1010
=B1001.RD3201.QQA02/%QQA60/++BAA1010
=B2301.RD3201.QQC01/%QQC11/++BAA1010
=Q1101.HH1001.UAC01/%UAC12/++BAA1010
=Q1101.HH1001.UAC02/%UAC12/++BAA1010
=Q1101.HH1001.UAC03/%UAC12/++BAA1010
Door01
LightingFixture01
LightingFixture02
LightingFixture03
OfficeRoom01
Pset_CoClass_Classification
ReferenceDesignation
Window01
Window02

rdfs:label
ReferenceDesignation

hasValue
=B1001.RD3201.QQA01/%QQA60/++BAA10101

hasValue
=B1001.RD3201.QQA02/%QQA60/++BAA10101

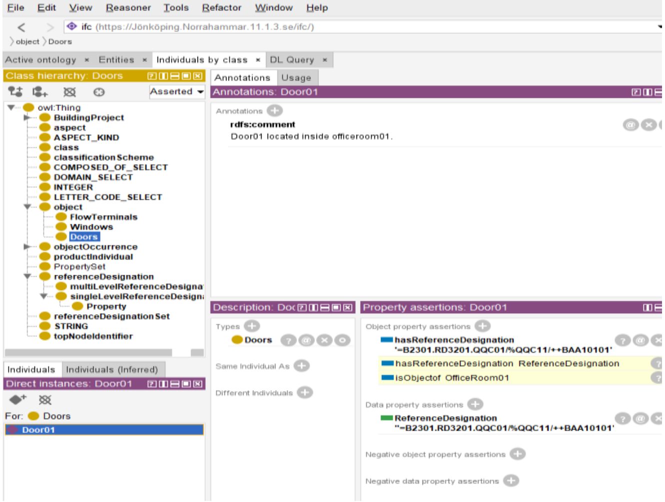
hasValue
=B2301.RD3201.QQC01/%QQC11/++BAA10101

hasValue
=Q1101.HH1001.UAC01/%UAC12/++BAA10101

hasValue
=Q1101.HH1001.UAC02/%UAC12/++BAA10101

hasValue
=Q1101.HH1001.UAC03/%UAC12/++BAA10101

Fig.8: The individual, 'ReferenceDesignation' with annotations.



owl:Thing
- BuildingProject
- aspect
- ASPECT_KIND
- class
- classificationScheme
- COMPOSED_OF_SELECT
- DOMAIN_SELECT
- INTEGER
- LETTER_CODE_SELECT
- object
  - FlowTerminals
  - Windows
  - Doors
- objectOccurrence
- productIndividual
- PropertySet
- referenceDesignation
  - multiLevelReferenceDesigna
  - singleLevelReferenceDesign
    - Property
- referenceDesignationSet
- STRING
- topNodeIdentifier

Door01

rdfs:comment
Door01 located inside officeroom01.

Types: Doors

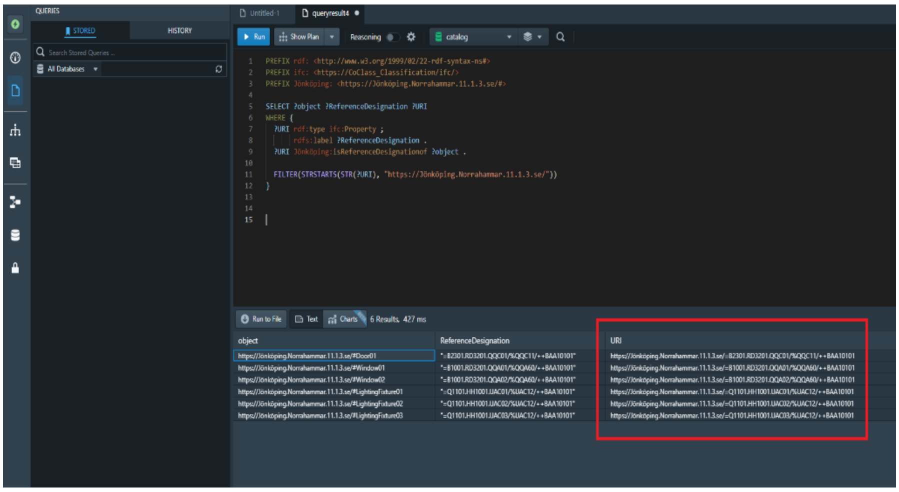
Object property assertions:
hasReferenceDesignation '=B2301.RD3201.QQC01/%QQC11/++BAA10101'
hasReferenceDesignation ReferenceDesignation
isObjectof OfficeRoom01

Data property assertions:
ReferenceDesignation "=B2301.RD3201.QQC01/%QQC11/++BAA10101'

Fig.9: Example of the representation of data properties and object property assertions for a door instance

3.5 Result: SPARQL Query Execution to access the RD based URI information

A SPARQL query was formulated to interrogate the ontology and validate the results. By querying the ontology, users can retrieve specific information about the components, such as their attributes, relationships, and RD-based URIs. The unique URIs resulted from the SPARQL query as highlighted in Figure 10, are a combination of the IRI defined in Section 3.4 and the RD values.



```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ifc: <https://CoClass_Classification/ifc/>
PREFIX Jönköping: <https://Jönköping.Norrahammar.11.1.3.se/#>

SELECT ?object ?ReferenceDesignation ?URI
WHERE {
  ?URI rdf:type ifc:Property ;
       rdfs:label ?ReferenceDesignation .
  ?URI Jönköping:isReferenceDesignationof ?object .

  FILTER(STRSTARTS(STR(?URI), "https://Jönköping.Norrahammar.11.1.3.se/"))
}
```

object	ReferenceDesignation	URI
https://Jönköping.Norrahammar.11.1.3.se/#Door01	"=B2301.RD3201.QQC01/%QQC11/++BAA10101"	https://Jönköping.Norrahammar.11.1.3.se/=B2301.RD3201.QQC01/%QQC11/++BAA10101
https://Jönköping.Norrahammar.11.1.3.se/#Window01	"=B1001.RD3201.QQA01/%QQA60/++BAA10101"	https://Jönköping.Norrahammar.11.1.3.se/=B1001.RD3201.QQA01/%QQA60/++BAA10101
https://Jönköping.Norrahammar.11.1.3.se/#Window02	"=B1001.RD3201.QQA02/%QQA60/++BAA10101"	https://Jönköping.Norrahammar.11.1.3.se/=B1001.RD3201.QQA02/%QQA60/++BAA10101
https://Jönköping.Norrahammar.11.1.3.se/#LightingFixture01	"=Q1101.HH1001.UAC01/%UAC12/++BAA10101"	https://Jönköping.Norrahammar.11.1.3.se/=Q1101.HH1001.UAC01/%UAC12/++BAA10101
https://Jönköping.Norrahammar.11.1.3.se/#LightingFixture02	"=Q1101.HH1001.UAC02/%UAC12/++BAA10101"	https://Jönköping.Norrahammar.11.1.3.se/=Q1101.HH1001.UAC02/%UAC12/++BAA10101
https://Jönköping.Norrahammar.11.1.3.se/#LightingFixture03	"=Q1101.HH1001.UAC03/%UAC12/++BAA10101"	https://Jönköping.Norrahammar.11.1.3.se/=Q1101.HH1001.UAC03/%UAC12/++BAA10101

Fig.10: Sparql query and the results

4 Discussion

In construction projects, effective communication and precise categorization are crucial. CoClass is a classification system designed to collectively describe construction systems, aiming to prevent misunderstandings and ensure accurate representation. The focus is on facilitating seamless data flow, enhancing efficiency in data exchange, minimizing rework, and enabling efficient data reuse.

This study investigates the potential for automating Reference Designation (RD) and creating unique identifiers using RDS and URIs, within CoClass framework. According to [25], the implementation of CoClass in AEC industry is time-consuming and costly. Therefore, companies are reluctant to adopt the CoClass classification system. However, automating Coclass codes, either partially or fully, could encourage them to use this classification system, improving interoperability and information exchange. Having a common language and the ability to communicate clearly and effectively can lead to significant cost savings and a better understanding of shared information. For instance, in the future, if a repair or maintenance is required for an element, RD-based URIs can provide detailed data regarding its location, constructive and functional systems, as well as its type. This would facilitate access to specific information for individuals, consequently it reduces time and cost.

In this study, the metrics were RDs and their URIs and the benchmarks were Coclass classification system, which is used to categorize the components and RDS, which can be used to label the components based on these classification systems. This is largely based on ISO 12006-2 and IEC 81346-2.

4.1 Challenges

Lack of Previous Studies: Absence of prior research made it hard to contextualize the benefits of automating CoClass within existing literature, hindering comparison with similar problems.

Time-consuming Implementation: Implementing CoClass parameters in a BIM model was difficult, especially since access to the CoClass API was restricted, requiring manual development in Excel. Automating this process could save time and enhance efficiency.

IFC to RDF Conversion Difficulty: Converting IFC data into RDF format posed challenges. Online converters were inaccessible or produced unsatisfactory results, necessitating the use of a Python script in Ifcopenshell. However, this process was time-intensive and involved trial and error.

Difficulty in Isolating Classification Parameters: The converted data included all Ifc data, making it challenging to isolate classification parameters, especially in large models. Filtering was necessary to focus only on CoClass properties, such as RD, and their corresponding values, to minimize errors.

4.2 Limitation

No access to the CoClass API and online RDF converters, as well as lack of previous literature for comparison, posed significant limitations to our study.

4.3 Future recommendations

For future research, alternative methods, such as Machine Learning (ML) or Artificial Intelligence AI for automation or integrating more advanced data processing techniques, would broaden the scope and appeal of the study. By leveraging ML and AI, the methodology could be more efficient and thereby reducing manual effort and human error and resulting in more accurate and useful data for the automation. This leads to better decision-making, improved project management, and enhanced overall productivity.

5 Conclusions

The study highlights the challenges and opportunities associated with implementing CoClass classification by automating Reference Designation (RD) and RD based unique URIs. By examining a three-story BIM model and selecting key components for detailed analysis, the research demonstrates the potential for improving efficiency and accuracy in data management through automation.

The manual development of an Excel sheet for CoClass codes and the creation of a Python script using IfcOpenShell to incorporate these codes into the IFC model were essential steps in the process. Furthermore, the development of a Dynamo script to automate RD generation based on CoClass parameters underscores the feasibility of automating data, which can significantly reduce time and cost.

Despite the theoretical potential for converting IFC data into RDF format through online converters, practical limitations necessitated the use of a Python script within IfcOpenShell. This process was labour-intensive and required extensive trial-and-error to achieve the desired outcome. The study also faced difficulties in isolating CoClass parameters from the extensive IFC model data, necessitating additional filtering steps.

The development of an ontology in Protégé and the execution of SPARQL queries to access RD-based URI information further illustrate the study's contributions to enhancing data interoperability and semantic integration. The creation of unique IRIs, guided by standards provided by Lantmäteriet, emphasizes the importance of standardized identifiers for efficient data retrieval and management.

Overall, the study underscores the potential benefits of automating RD and generation of RD based URIs, which can lead to significant improvements in data management within the construction industry. However, the challenges encountered, particularly the lack of access to online converters and the CoClass API, highlight the need for better resources and tools to support future research and implementation efforts.

By leveraging machine learning and artificial intelligence, the methodology could become more efficient, reducing manual effort and human error. This would result in more accurate and useful data for automation, leading to better decision-making, improved project management, and enhanced overall productivity.

Reference

[1] Hedro Zayton & Hamed Safdari, (2018). The Role of Interoperability in Construction Projects. Communication and the Implementation of CoClass. Master's thesis in Civil Engineering; Design and Construction Project Management, Chalmers University of Technology Göteborg, Sweden.

[2] Sadrinooshabadi, S., Taheri, A., Yitmen, I., & Jongeling, R. (2020). Requirement management in a life cycle perspective based on ISO 19650-1 and CoClass as the new classification system in Sweden. *Construction Engineering and Lighting Science, Jönköping University*, Jönköping, Sweden, and *PlanB BIM AB*, Stockholm, Sweden.

[3] Balslev, H. (2016). The Reference Designation System (RDS): A common naming convention for systems and their elements. In 26th Annual INCOSE International Symposium (IS 2016), Edinburgh, Scotland, UK.

[4] Svensk Byggtjänst. (2016b). Appendix A1: Referensbeteckningar och exempel. Hämtad from: https://static.byggtjanst.se/coclass/pdf/Appendix-A1-Exempel-referensbeteckningar-v1.1-20161122.pdf

[5] Raido Puust & Roode Liias. (2023), Structuring bim-related construction data through a standards-based classification system. Tallinn university of technology, Tallinn, Estonia.

[6] Sadeghineko, F., & Kumar, B. (2022). Application of semantic web ontologies for the improvement of information exchange in existing buildings. Construction Innovation: Information, Process, Management, 22(3), 444-464. https://static.byggtjanst.se/coclass/pdf/Appendix-A1-Exempel-referensbeteckningar-v1.1-20161122.pdf

[7] Svensk Byggtjänst. (2017a). BSAB. Retrieved 2017-05-20, from: https://byggtjanst.se/tjanster/bsab/

[8] Svensk Byggtjänst. (2017b). Om CoClassnya generationen BSAB. Retrieved 2017- 05-25, from: https://coclass.byggtjanst.se/sv/om#om-coclass

[9] Carlsson, O., 2017, Analys och tillämpning av CoClass, Chalmers University of Technology, Göteborg.

[10] SS-EN/IEC 81346 (2009) https://elstandard.se/documents/preview/316901

[11] Ehrlinger, L., & Wöß, W. (2016). Towards a Definition of Knowledge Graphs. Institute for Application Oriented Knowledge Processing, Johannes Kepler University, Linz, Austria.

[12] Paulheim, H. (2016). Knowledge Graph Refinement: A Survey of Approaches and Evaluation Methods. *Semantic Web Journal*, (Preprint), 1–20.

[13] Noy, N. F., McGuinness, D. L., et al. (2001). Ontology development 101: A guide to creating your first ontology: Stanford knowledge systems laboratory technical report KSL-01-05

[14] Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., Melo, G. D., Gutierrez, C., et al. (2021). Knowledge graphs. ACM Computing Surveys, 54(4), 1–37.

[15] Ji, S., Pan, S., Cambria, E., Marttinen, P., & Philip, S. Y. (2021). A survey on knowledgegraphs: Representation, acquisition, and applications. IEEE Transactions on Neural Networks and Learning Systems, 33(2), 494–514.

[16] Terkaj, W., Schneider, G. F., & Pauwels, P. (2017). Reusing Domain Ontologies in Linked Building Data: The Case of Building Automation and Control. *Institute of Industrial Technologies and Automation (ITIA-CNR), Milan, Italy; Fraunhofer Institute for Building Physics IBP, Nuremberg, Germany; Ghent University, Ghent, Belgium*.

[17] W3C, http://www.w3.org/2001/sw/ (accessed December 2015).

[18] Lee, D. Y., Chi, H. l., Wang, J., Wang, X., & Park, C. S. (2016). A linked data system framework for sharing construction defect information using ontologies and BIM environments. Automation in Construction, 102–113.

[19] Berners-Lee, T. (2006). Linked data-design issues. Available at: http://www.w3.org/DesignIssues/LinkedData.html (Accessed: December 2015).).

[20] Lee, D.-Y., Chi, H.-L., Wang, J., Wang, X., & Park, C.-S. (2016). A linked data system framework for sharing construction defect information using ontologies and BIM environments. *Automation in Construction*, 72, 12-21.

[21] Ayyadurai Charles, V.R.D. (2016), Automated rule checking for in-house BIM norms of building models

[22] Hitzler, P. (2011). Knowledge Representation for the Semantic Web. Dayton: Wright State University.

[23] Helen Eriksson, Tim Johansson, Per-Ola Olsson, Maria Andersson. Jakob Engvall, Isak Hast and Lars Harrie. (2020). Requirements, Development, and Evaluation of a National Building Standard—A Swedish Case Study.

[24] Ekholm, A & Häggström, L (2011). Building classification for BIM – reconsidering the framework. In Proceedings of the 28th W78 Conference, Sophia Antipolis, France.

[25] Carlsson, O. (2017). *Analys och tillämpning av CoClass*. Examensarbete i högskoleingenjörsprogrammet Byggingenjör, Institutionen för arkitektur och samhällsbyggnadsteknik, Avdelningen för Construction Management, Chalmers Tekniska Högskola, Göteborg

[26] Boverket, Byggnadsinformationsmodellering, BIM, Stöd för offentliga aktörer, 2024
https://www.boverket.se/globalassets/publikationer/dokument/2023/byggnadsinformationsmodellering-bim.pdf