Hindawi
Advances in Civil Engineering
Volume 2021, Article ID 8861625, 18 pages
https://doi.org/10.1155/2021/8861625

Research Article

Semiautomatic Generation of Code Ontology Using ifcOWL in Compliance Checking

Yuchao Li,[1] **Qin Zhao**,[1] **Yunhe Liu**,[1] **Xinhong Hei**,[2] **and Zongjian Li**[3]

[1]*School of Civil Engineering and Architecture, Xi'an University of Technology, Xi'an 710048, China*
[2]*School of Computer Science and Engineering, Xi'an University of Technology, Xi'an 710048, China*
[3]*State Key Laboratory of Rail Transit Engineering Information, China Railway First Survey and Design Institute Group Co.,Ltd, Xi'an 710048, China*

Correspondence should be addressed to Qin Zhao; zhaoqin6688@xaut.edu.cn

Received 5 February 2021; Revised 24 June 2021; Accepted 2 July 2021; Published 10 July 2021

Academic Editor: Hui Yao

Code compliance checking is a very important step in engineering construction, but most of code compliance checking relies on manual review at present. With the development of semantic web technology, ontology can be used to represent code information and check the code automatically. However, code ontology is established manually by researchers who have sufficient domain knowledge, in which it is easy to cause poor hierarchical structure of classes. It is also possible for code ontology not being suitable for compliance check. This paper proposes a semiautomatic construction method of railway code ontology based on ifcOWL. The railway code ontology is developed by converting ifcOWL which extends semantic information of railway code. This method can ensure the completeness of the hierarchical relationship of the classes in code ontology with good scalability, which makes use of taxonomy in ifcOWL. The establishment of ontology is divided into two processes with low coupling, namely, extension and conversion, which reduces the domain knowledge requirements of the researchers. Finally, a practical specification is selected to generate a code ontology that achieves some clauses checking.

1. Introduction

Compliance checking is a very important content in the process of engineering design. The traditional method relies on manually consulting the design model and matching the results with the code, which is not only inefficient but also prone to errors [1]. Over the recent years, the usage of semantic web technologies has notably increased in the domains of architecture, engineering, and construction. Specification rules are represented by semantic web for automatic compliance checking [2, 3]. Semantic web technology is an ideal approach to merge heterogeneous information from various sources and to explicit information by formal standardized knowledge representations. As a critical element of the semantic web, ontology plays a significant role in its application, which provides a framework that can be employed to model knowledge and translate the knowledge into a form that can be interpreted by both computers and humans [4]. Ontology can be applicated on the checking of fire code compliance [5], building evacuation code [6], and so on. On the one hand, code ontology is often used as the representation model of specification semantics in compliance checking. The common way of checking is to establish SPARQL (SPARQL Protocol and RDF Query Language) statements or SWRL (Semantic Web Rule Language) rules on the code ontology to check the model information. On the other hand, the information integrity of the model should be considered before compliance checking; that is, we need to check whether these models have the attributes that need to be checked [7]. It is easy to cause ambiguity of results when there is a lack of required checking information in the actual compliance checking work. The semantic integrity of building information model is a necessary condition to ensure the application in a specific field [8]. Code ontology is composed of the concepts in the code and relationship between them, which can provide complete

semantic information for integrity checking. Whether it is compliance checking or integrity checking, code ontology is an important prerequisite.

However, the establishment of ontology is a difficult work, which needs a lot of investigations from experts in the field [9]. Since ontology is a concept that was put forward very early, there are many methods for ontology modeling [10, 11]. There is no unified method for all ontologies, and different methods are needed in different situations. [12] In the process of ontology establishment, ontology reuse is a very important method [13]. Other available ontologies can provide basic content to save a lot of work for ontology establishment. IfcOWL is one of the few ontologies in the field of building information model which is transformed from IFC (Industry Foundation Classes) and is also the only ontology recommended by BuildingSMART [14]. ifcOWL and IFC have the same limits of information representation which can represent the semantics of many concepts in the field of architecture. However, ifcOWL is mainly aimed at the architecture engineering field, and there is still a lack of necessary concept description for other fields [15]. In addition, a part of content is missing in the process of ifcOWL conversion from IFC.

Therefore, this paper proposes a semiautomatic generation method for rail engineering code ontology based on ifcOWL. The concept semantic in rail engineering code is supplemented by extending ifcOWL and then extended ifcOWL is converted to code ontology. This method can reduce the work of concept hierarchy in ontology modeling and reduce the requirement of domain knowledge level of ontology creator. The code ontology has strong extensibility because the concepts are converted from ifcOWL.

2. Related Work

2.1. Ontology Application in Code Compliance Checking. The code checking in construction engineering can be automatically executed by computer, whose premise is that the specification should be capable of being correctly "understood" and interpreted by the machine. Ontology can represent concepts and relationships, and the content to be checked in the specification can be represented by rules supported by semantic web for semantic reasoning directly instead of syntax matching. There are many researches that improve the problem of code compliance checking through ontology. Zhong et al. [16] proposed a CQIEOntology to check the compliance of construction quality, in which SWRL rules are established to infer the inspection results. CQIEOntology is constructed according to part of Java inspection framework, which includes inspection objects and inspection tasks. Xu and Cai [17] checked the spatial compliance of underground utilities through four related ontologies: utility product ontology (UPO), transportation object ontology (TOO), geometry Ontology (GEO), and utility spatial rule ontology (USRO). Four different ontologies involve different contents, UPO and TOO provide the conceptualization of urban infrastructure products, in which the main concepts/classes in the domain of interest were identified based on a review of relevant open standards and textual documents. The compliance checking results are obtained through SPARQL. Lu et al. [18] created CSC ontology to check the construction safety specifications. Domain experts understand the knowledge from the construction solution database and define the concept classification of structural safety inspection. The establishment of ontology in these studies is more or less dependent on the existing classification of some concepts.

2.2. Ontology Building Method. Since the concept of ontology was put forward in 1993 [19], there have been many different models for ontology construction; ENTERPRISE ontology [20] and TOVE (Toronto Virtual Enterprise) [21] are considered to be the earliest ontology building method guidelines. Then, many ontology building methods based on the characteristics of different fields are proposed, such as KACTUS [22] in manufacturing and engineering domains, IDEF5 (Integrated definition for Ontology Description Capture Method) [23] in Software engineering domain, METHONTOLOGY [24] in chemistry domain, On-To-Knowledge method [25] in knowledge management system domain, and Ontology Development 101 method [26]. In practical engineering applications, Ren [27] established a complete BrM ontology for bridge maintenance knowledge management through the Ontology Development 101 method from concept definition to rule establishment, and the established rules can evaluate the technical condition of the bridge. Abanda [28] established the cost estimation ontology through METHONTOLOGY method and the construction of the ontology is decomposed to different small tasks. Concepts, attributes, and relationships were obtained from UKNRM1 ontology, and then concept classification was established by top-down method.

Building specification is a highly professional field, which contains many and complex semantic knowledge [29]. The complexity of specification makes the establishment of specification ontology more challenging. Both Ontology Development 101 and METHONTOLOGY methods mentioned that a very important step is to consider ontology reuse. If existing ontologies can be reused or extended, it is easier to define taxonomy and attribute relationship [30].

2.3. IFC and ifcOWL. IFC, an important information conceptual model in the field of AEC (architecture, engineering, and construction), has been developed as a data model to describe the data of construction industry and registered as ISO 16739 : 2013. IFC defines an entity relationship model, which is composed of hundreds of entities organized into an object-based inheritance hierarchy. In order to support the semantic web application of data interoperability, flexible data exchange, distributed data management, and BuildingSMART subsequently released ifcOWL which is represented by an agreed Web Ontology Language (OWL) ontology for IFC. Therefore, IFC or ifcOWL can be used to improve their domain-specific ontology. Liu et al. [31] constructed an ontology that can be used for engineering quantity calculation. The ontology not only includes the

description of basic components in IFC but also creates new concepts such as studs and plates for engineering quantity calculation. Zhong et al. [32] simplified classes and object properties in IFCOWL, and a building information ontology suitable for building environment compliance checking is established. Borrmann et al. [33] mentioned that the shield tunnel model ontology is established according to the spatial concept in IFC. A part of the ifcOWL is reused in these studies. However, ifcOWL does not contain the many features of IFC, such as some attribute mapping [34, 35]. Moreover, the IFC standard has too few concepts to describe specific-domain components [36].

Therefore, this paper extends ifcOWL to represent the concept in the code through understanding the rail engineering code. A conversion method is proposed which can semiautomatically convert the ifcOWL into a part of the rail engineering code ontology. This method can save some labor resources, and code ontology has strong extensibility because it is converted by ifcOWL.

3. ifcOWL Extension Method

Code ontology should contain a complete description of the concepts in the code. The concepts which are not contained in ifcOWL need to be extended. IfcOWL is extended with three different ways, namely, attribute value extension, entity extension, and property extension, respectively.

3.1. Attribute Value Extension. Attribute value extension mainly aims at more specific description of generalized entities, such as fire door or mechanical ventilation system. These concepts are equivalent to generalized concepts in IFC added a restriction, such as concept: Door + feature: Fire Protection = Fire Door. The type of entity is represented through its own attributes, such as PredefinedType. However, all types cannot be included in IFC. In the above example, it can be understood that Fire Protection is a characteristic of Door. But in other concepts, its restriction is not the same as semantic relationship between Door and Fire Protection, such as Concrete is a kind of Material. Different types are required to represent different entities. Generally, the attribute which define entity types is selected to extend more types. ObjectType attribute is selected for IfcObject entity and the Category attribute is selected for IfcMaterial. The extension method is shown in Figure 1. A class is added as a subclass of the attribute value type which needs to be extended, of which the instances are extended value of attribute; it is shown in the red part of Figure 1. The restriction of corresponding class is modified in ifcOWL. The approach is similar to extending the enumeration value of PredefinedType attribute. Different entities will require different extension classes, even if they are extended by same attribute. When extending new content in the future, it is easy to find extension class. It is also convenient to query extended content.

3.2. Entity Extension. Although the content of IFC is continually updated, the concepts of track domain, road, and bridge domain have been added in the version of

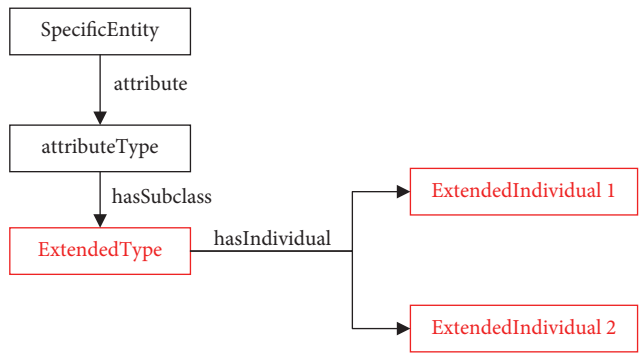


Figure 1: Attribute value extension method.

IFC4x3_RC1. More professional concept in the code is often not expressed in IFC. Entity extension is aimed at some concepts that are not represented in IFC. ifcOWL is generated by EXPRESS format conversion of IFC. The method of entity extension is to extend the new concept with EXPRESS format to ifcOWL.

The concept description in the code can refer to IfcRail bSI SPEC standard [37], which is submitted by China Railway BIM alliance and certified by BuildingSMART. The shared spatial structure of IfcRail standard is shown in Figure 2. The spatial structure of the IFC4x3 standard is shown in Figure 3. It can be seen from the figure that IfcRail standard can be regarded as an extension of IFC standard. IfcRail standard is submitted in 2016, some content that is the same as IFC in old version has changed. For example, IfcFacility that is superclass of IfcBuilding is the latest addition. Therefore, the EXPRESS structure of extended concepts is corresponding to the latest IFC and then extended to ifcOWL.

3.3. Property Extension. Most entities of IFC description document have a property set templates which contain the properties that entity should have and property value type. For instance, the property set of IfcTransportElement is shown in Figure 4, when PredefinedType is ELEVATOR. Obviously, these attributes are not complete and not mapped in ifcOWL. The properties of concepts are indispensable for integrity checking.

The property association mode in IFC is that entity is connected to property set (IfcPropertySet) through the relationship (IfcRelDefinesByProperties) and property set is associated with multiple properties through attribute HasProperties. Properties can be described by six types of entities, namely, IfcPropertySingleValue, IfcPropertyTableValue, IfcPropertyEnumeratedValue, IfcPropertyList Value, IfcPropertyReferenceValue, and IfcPropertyBounded Value, which, respectively, represent different type of property. In order to fully express the semantics and reduce the complexity of extension, IfcPropertySingleValue is selected as the type of extended property. As can be seen from Figure 4, each property defines not only the type but also the value type. For example, the ClearWidth property defines the property value type as IfcPositiveLengthMeasure inherited from IfcMeasureValue. There are more basic measure types in the

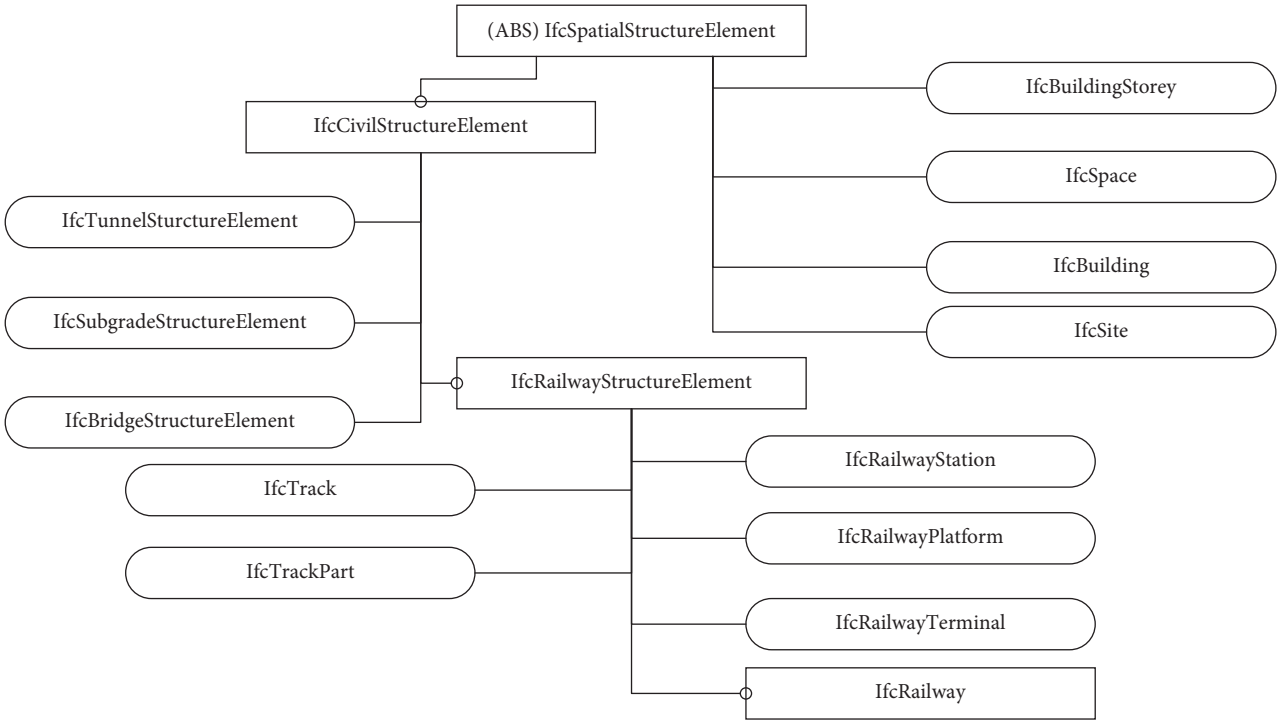


Figure 2: The shared spatial structure of IfcRail standard.

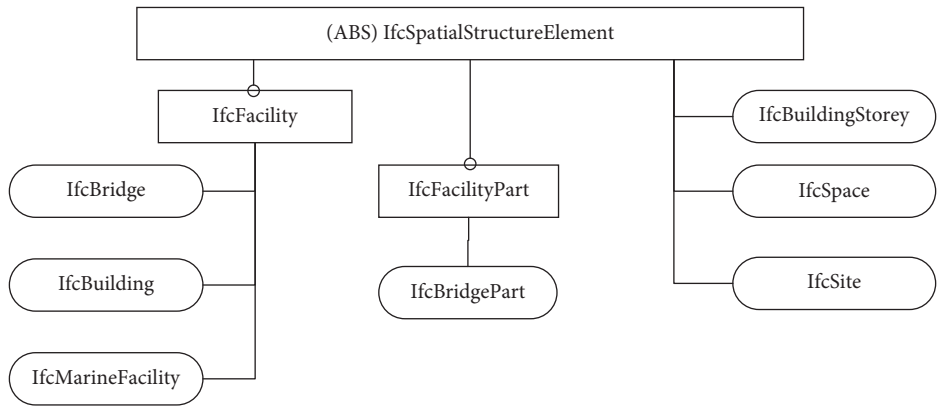


Figure 3: The spatial structure of the IFC4x3 standard.

Pset_TransportElementElevator
FireFightingLift (P_SINGLEVALUE/IfcBoolean): indication whether the elevator is designed to serve as a fire fighting lift the case of fire (TRUE) or not (FALSE). A fire fighting lift is used by fire fighters to access the location of fire and to evacuate people.
ClearWidth (P_SINGLEVALUE/IfcPositiveLengthMeasure): clear width of the object (elevator). It indicates the distance from the inner surfaces of the elevator car left and right from the elevator door. The shape information is provided in addition to the shape representation and the geometric parameters used within. In cases of inconsistency between the geometric parameters and the shape properties, provided in the attached property, the geometric parameters take precedence.
ClearDepth (P_SINGLEVALUE/IfcPositiveLengthMeasure): clear depth of the object (elevator). It indicates the distance from the inner surface of the elevator door to the opposite surface of the elevator car. The shape information is provided in addition to the shape representation and the geometric parameters used within. In cases of inconsistency between the geometric parameters and the shape properties, provided in the attached property, the geometric parameters take precedence.
ClearHeight (P_SINGLEVALUE/IfcPositiveLengthMeasure): clear height of the object (elevator). The shape information is provided in addition to the shape representation and the geometric parameters used within. In cases of inconsistency between the geometric parameters and the shape properties, provided in the attached property, the geometric parameters take precedence.

Figure 4: The properties of elevator in IFC document.

subclass of IfcMeasureValue, which come from the definition in ISO 10303-41. The unit of value is specified indirectly by basic measure types (international unit by default). Unit and value type of the property are defined by the type of the NominalValue attribute. The method is shown in Figure 5. The first step of property extension is to extend the property definition and define value type of NominalValue. The second step is to extend the property set definition. New relationship entity is extended to relate entity finally. As shown in Figure 5, the extended content is red. A new attribute IsDefinedBy is extended which should have been defined in ifcOWL, but IsDefinedBy is not defined because the range of RelatedObjects attribute of IfcRelDefinesByProperties is different from domain of IsDefinedBy.

4. ifcOWL Conversion Method

The concepts in the specification information are represented by extending ifcOWL, and the hierarchical structure between concepts can be obtained by converting extended ifcOWL. The conversion method is proposed in this section, which can convert the extended content to code ontology. The method is divided into three categories, entity conversion, attribute value conversion, and entity property conversion, which are corresponding to different extension methods.

4.1. Entity Conversion. Entity conversion is to convert the corresponding concept to code ontology according to the structural form in ifcOWL. The prefix of IFC entity is omitted in code ontology; for example, IfcDoor is converted to Door. Entity conversion does not focus on the attributes of the entity and the class hierarchy in ifcOWL is only retained, whose purpose is to improve the concept classification in the code with the help of the hierarchical structure in ifcOWL.

Two criteria are defined in entity conversion:

(1) If the converted entity is a subclass of IfcObject, all the parent classes of the entity up to IfcObject need to be converted

(2) Only the entity involved in the specification information is converted

The first criterion is to preserve the relationship between the concepts. The second criterion can avoid entity conversion that does not appear in the code. The conversion process is shown in Figure 6. Entities with $*$ are entities that need to be converted and entity conversion retains the hierarchical relationship between IfcObject and its subclasses.

4.2. Attribute Value Conversion. The purpose of attribute value extension is to describe generalized concepts with certain characteristics. The generalized concepts can be regarded as the parent class of concepts with certain characteristics in code ontology, such that doors are the parent classes of fire doors.

The conversion method is shown in Figure 7. Entities are converted into classes in the code ontology according to the entity conversion method. Characteristic class is characteristic description of concept in the code ontology. The attributes of the extended attribute value are converted to the subclasses of Characteristic. There is a hasCharacteristic relationship between the entity class and the characteristic class, which means that the entity class has certain characteristics. The extended types in ifcOWL are converted to ExtendedType. The extended attribute values are, respectively, converted into the subclass of the characteristic class and the subclass of the converted entity and there is a hasCharacteristic relationship between them.

The understandable description of attribute value conversion is to take the extended attribute value as a subclass of the entity and consider attribute values as a certain characteristic of the entity, which is a representation mode of the characteristic class and entity class in code ontology. The subclasses of Characteristic class are used to distinguish the attributes of different extended attribute, which can also be understood as distinguishing different characteristic types.

4.3. Property Conversion. A new representation mode of property is defined in code ontology. Property is regarded as a new class in code ontology. The relationship hasProperty is set between the entity class and the property class, which indicates that the entity has properties. Units class is defined in code ontology, and the relationship Unit between Property class and Units means the unit of the property. Data property Value is defined to represent value type of the property. Property class in code ontology is converted from extended properties in ifcOWL. The conversion method is shown in Figure 8.

Entities are converted into classes in code ontology according to the entity conversion method. The extended properties are converted into subclasses of Property class in code ontology. According to the value type of NominalValue of the extended proper, Units class in code ontology is mapped and Unit relationship is set between properties and units. The mapping relationship between data type in OWL, the value type of NominalValue, and unit is shown in Table 1. The value type of Value is defined according to mapping relationship.

5. Experimental Results

The code for design of Metro (GB50157-2013) [38] is one of the important codes in the metro construction domain in China, which covers at least 14 specialties. The code for design of Metro contains not only the provisions of the elements in the metro station, but also the specifications for trains, subgrades, lines, and other nonconstruction fields. The purpose of this paper is to create a code ontology that can be used for compliance checking in the construction engineering field. Therefore, the following types of specifications have been removed during the experiment:

(1) The first specification is abstraction limitation of information. For example, in clause 9.4.6, the room

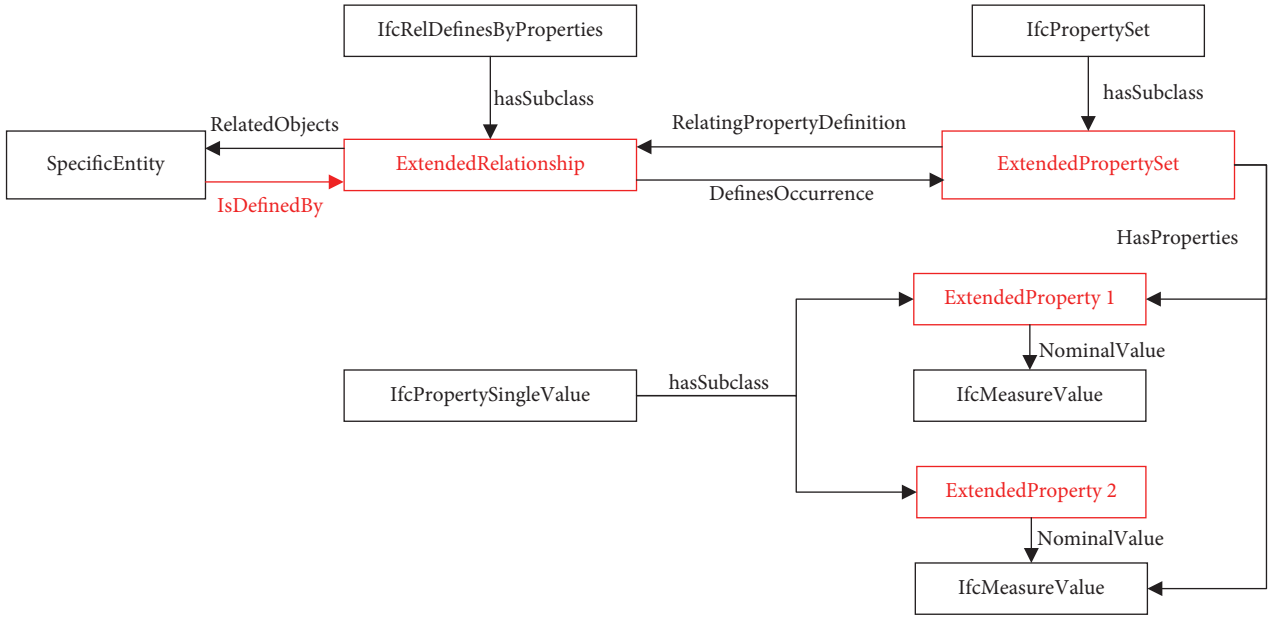


Figure 5: The property extension method.

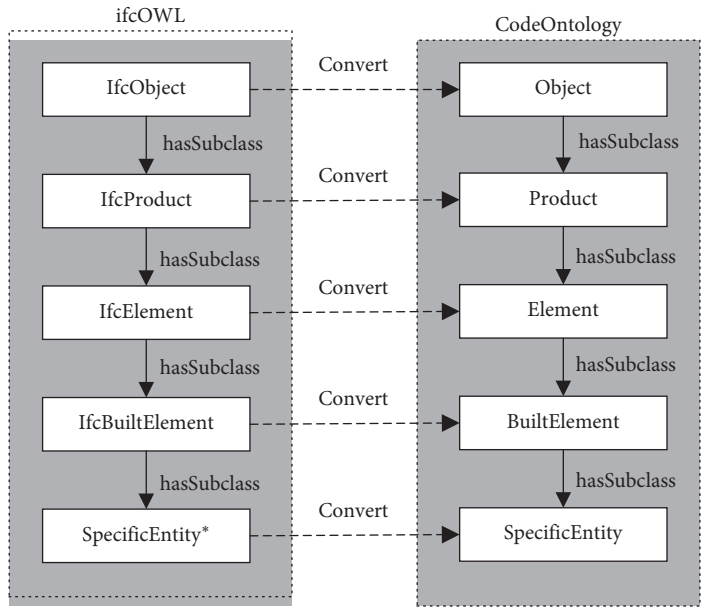


Figure 6: The entity conversion method.

with noise source should take sound insulation and absorption measures. These measures are abstract and need to be described more specific by experts.

(2) The second is nonconstruction engineering content. Like vehicle field, this part limits the requirements for trains in metro engineering.

(3) Third, we have regulations of the construction process. For example, the construction methods of underground station structure under specific conditions are specified in clause 11.4. Nowadays, most of the compliance checking is to check the rationality of the design results, and the representation of the design process is unclear.

The mandatory clauses of different field are selected under the above premise. 10 clauses are selected to correspond to 10 specialties, which convert to code ontology used different methods. The code clauses are shown in Table 2, in which the concepts are similar to other clauses.

A concept mapping table is generated manually to map the corresponding concepts and extension contents. The table is divided into three mapping tables corresponding to three extension methods, which are shown in Tables 3–5,

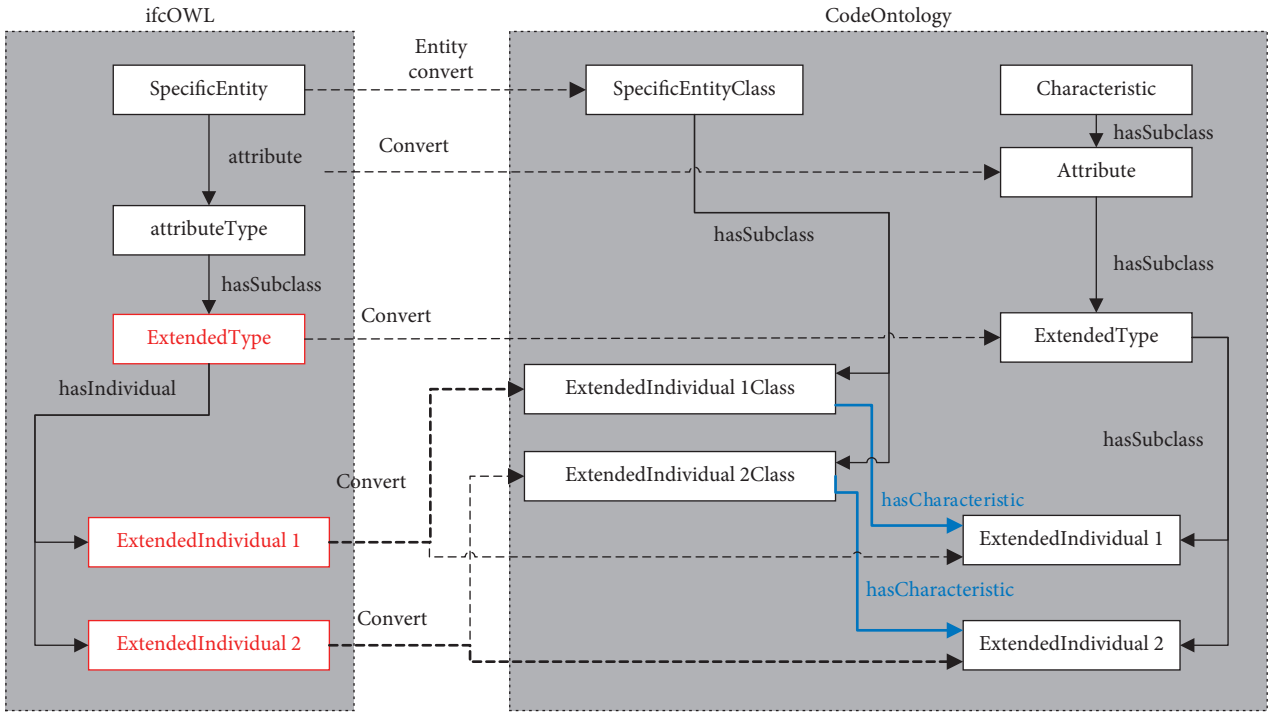


Figure 7: The attribute value conversion method.

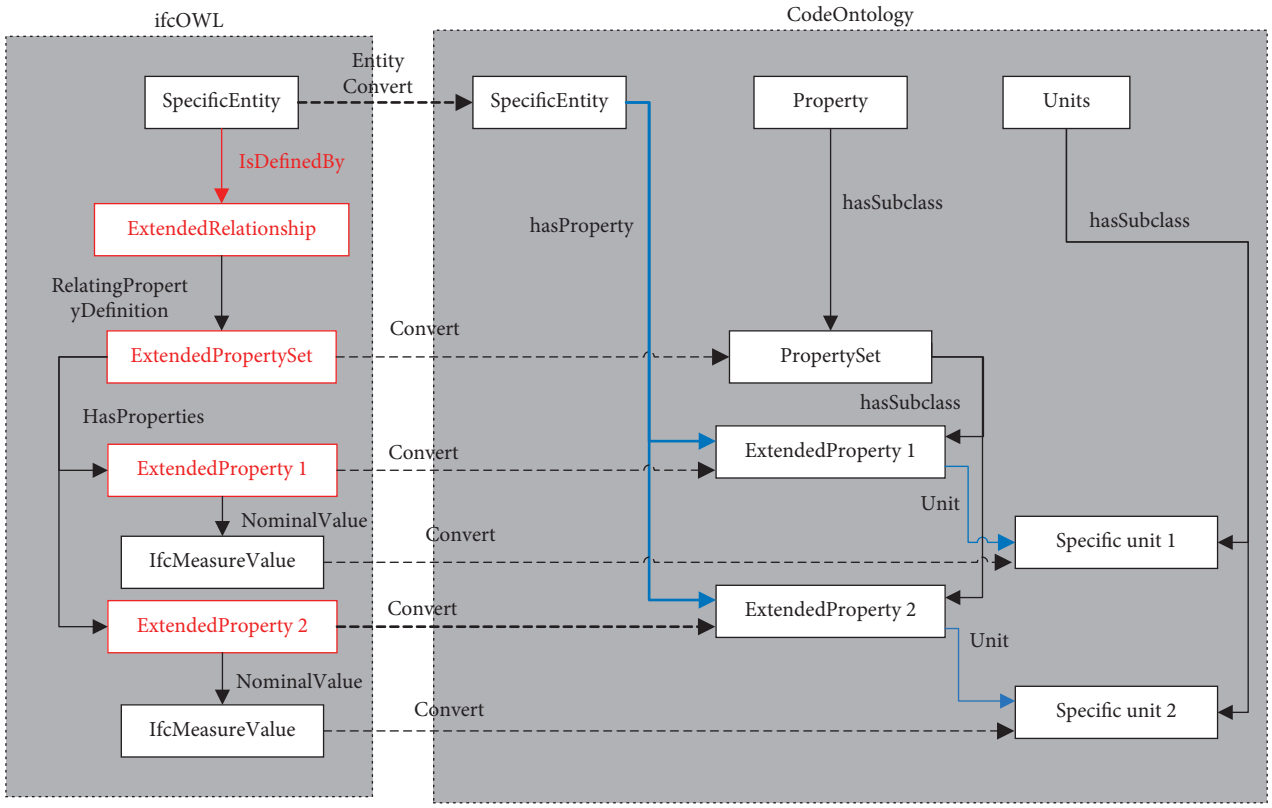


Figure 8: The property conversion method.

Table 1: The mapping relationship between data type in ifcOWL, the value type of NominalValue, and unit.

Value type in IFC	Value type in OWL	Unit
IfcVolumeMeasure	xsd:double	Cubic meter
IfcTimeMeasure	owl:real	Second
IfcThermodynamicTemperatureMeasure	owl:real	Kelvin
IfcSolidAngleMeasure	owl:real	Steradian
IfcRatioMeasure	owl:real	—
IfcPlaneAngleMeasure	owl:real	Radian
IfcMassMeasure	owl:real	Kilograms
IfcLengthMeasure	owl:real	Meters
IfcElectricCurrentMeasure	owl:real	Ampere
IfcParameterValue	owl:real	—
IfcDescriptiveMeasure	xsd:string	—
IfcCountMeasure	xsd:PostivieInteger	—
IfcContextDependentMeasure	owl:real	—
IfcAreaMeasure	owl:real	Square meter
IfcAmountOfSubstanceMeasure	owl:real	Mole
IfcLuminousIntensityMeasure	owl:real	Candela

Table 2: Code clauses in experiment.

Clauses	Content	Domain	Processing method
7.1.3	The design service life of the main structure of the ballastless track and the concrete track sleepers should not be less than 100 years.	Track	Attribute value and property extension
8.2.5	The subgrade bed shall be divided into surface layer and bottom layer. The thickness of surface layer shall not be less than 0.5 m, and that of bottom layer shall not be less than 1.5 m.	Subgrade	Entity extension
9.4.4	Signs for guidance, accident evacuation, and passenger service should be set up inside the station.	Station architecture	Entity extension
13.2.31	The room with gas fire extinguishing should be equipped with mechanical ventilation system, and the exhausted gas must be directly discharged to the ground	Ventilation and air conditioning	Attribute value and property extension
14.2.5.5	The water supply pipe shall not pass through the substation, communication signal room, control room, distribution room and other electrical rooms.	Water supply and drainage	Attribute value extension
19.4.5	Fire detectors should be installed in the public area of station hall floor, public area of platform floor, public area of transfer, various equipment rooms, warehouses, duty rooms, offices, entrance and exit, power distribution rooms in underground stations.	Automatic fire alarm	Attribute value extension
22.6.3	The data line shall be halogen-free, low smoke and flame-retardant shielded cable.	Passenger information system	Attribute value and property extension
24.8.1	The control center should be equipped with automatic fire alarm, building automation system, fire accident broadcast, automatic fire extinguishing, water firefighting, smoke control systems.	Operation control center	Attribute value extension
25.1.15	When the rated speed of elevator is 0.5 m/s and the lifting height is not more than 6 m, the number of upper and lower horizontal steps shall not be less than 2; when the rated speed is 0.5 m/s and the lifting height is greater than 6 m, the number of upper and lower horizontal steps shall not be less than 3; when the rated speed is greater than 0.65 m/s, the number of upper and lower horizontal steps shall not be less than 3.	Passenger transport equipment	Property extension
28.2.5	Two fire compartments should be separated by a firewall with a fire resistance rating of not less than 3 h and a class A fire door. When the firewall is equipped with an observation window, a class a fire window should be used. The slab in the fire compartment should have a fire resistance rating of not less than 1.5 h.	Disaster prevention	Attribute value and property extension

Table 3: Entity extension content mapping relationship.

Clauses	Concepts	Entity in IfcRail
9.4.4	Station	IfcRailwayStation
9.4.4	Denoter	IfcRailwayDenoterDevice
8.2.5	Subgrade	IfcSubgrade
8.2.5	Subgrade bed	IfcSubgradeFillingWorks

Table 4: Attribute value extension content mapping relationship.

Clauses	Concepts	Entity in IFC	Attribute	Extended value
19.4.5	Public area of station hall floor	IfcSpace	ObjectType	ConcourseLayerPublicArea
19.4.5	Public area of platform floor	IfcSpace	ObjectType	PlatformLayerPublicArea
19.4.5	Public area of transfer	IfcSpace	ObjectType	TransferPublicArea
19.4.5	Equipment room	IfcSpace	ObjectType	EquipmentRoom
19.4.5	Warehouse	IfcSpace	ObjectType	StoreRoom
19.4.5	Duty room	IfcSpace	ObjectType	DutyRoom
19.4.5	Office	IfcSpace	ObjectType	Office
19.4.5	Power distribution room	IfcSpace	ObjectType	ElectricityDistributionRoom
19.4.5	Entrance and exit	IfcSpace	ObjectType	AccessRoad
22.6.3	Data line	IfcCableSegment	ObjectType	DataCable
24.8.1	Control center	IfcBuilding	ObjectType	ControlCenter
24.8.1	Automatic fire alarm system	IfcDistributionSystem	ObjectType	FireAlarmSystem
24.8.1	Building automation system	IfcDistributionSystem	ObjectType	BuildingAutomationSystem
24.8.1	Fire accident broadcast system	IfcDistributionSystem	ObjectType	FireBroadcastingSystem
24.8.1	Automatic fire extinguishing system	IfcDistributionSystem	ObjectType	FireExtinguishingSystem
24.8.1	Water firefighting system	IfcDistributionSystem	ObjectType	WaterFireFightingSystem
24.8.1	Smoke control system	IfcDistributionSystem	ObjectType	SmokeProtectingSystem
7.1.3	Concrete	IfcMaterial	Category	Concrete
28.2.5	Fire compartment	IfcSpace	ObjectType	FireCompartment
28.2.5	Firewall	IfcWall	ObjectType	FireWall
28.2.5	Fire window	IfcWindow	ObjectType	FireWindow
28.2.5	Fire door	IfcDoor	ObjectType	FireDoor
13.2.31	Mechanical ventilation system	IfcDistributionSystem	ObjectType	MechanicalVentilation
14.2.5.5	Substation room	IfcSpace	ObjectType	SubstationRoom
14.2.5.5	Communication and signal room	IfcSpace	ObjectType	CommunicationSignalRoom
14.2.5.5	Control room	IfcSpace	ObjectType	ControlRoom

Table 5: Property extension content mapping relationship.

Clauses	Concepts	Entity in IFC	Restriction	Extended property	Value type
7.1.3	Track	IfcTrackElement		StructureType	IfcDescriptiveMeasure
7.1.3	Track	IfcTrackElement		DesignedLifetime	IfcTimeMeasure
22.6.3	Cable	IfcCableSegment		HalogenProof	IfcBoolean
22.6.3	Cable	IfcCableSegment		IsFlameRetardant	IfcBoolean
22.6.3	Cable	IfcCableSegment		IsInsulation	IfcBoolean
25.1.15	Escalator	IfcTransportElement		Height	IfcLengthMeasure
25.1.15	Escalator	IfcTransportElement	PredefinedType ESCALATOR	RatedVelocity	IfcLinearVelocityMeasure
25.1.15	Escalator	IfcTransportElement		StepNumber	IfcCountMeasure
28.2.5	Fire door	IfcDoor	ObjectType FireDoor	Rating	IfcDescriptiveMeasure
28.2.5	Fire window	IfcWindow	ObjectType FireWindow	Rating	IfcDescriptiveMeasure
28.2.5	Slab	IfcSlab		FireEndurance	IfcTimeMeasure
13.2.31	Mechanical ventilation system	IfcDistributionSystem	ObjectType MechanicalVentilation	ExhaustMode	IfcDescriptiveMeasure

respectively. The restriction is added in Table 5 because the properties is related to the concepts that the entity is restricted. Jena package in the Java environment is used to deal with OWL in experiment.

5.1. Entity Extension and Conversion. In entity extension, Table 3 is imported to obtain the EXPRESS structure of entities corresponding to entities in IfcRail. The extended entity is added in child hierarchy of its parent class corresponding class in ifcOWL. The extended entities are shown in Figure 9, which is displayed by Protege's OntoGraf.

The concepts in ifcOWL are converted according to entity conversion method. The prefix "Ifc" is omitted and the root concept of structure is IfcObject. The converted class structure is shown in Figure 10.

5.2. Attribute Value Extension and Conversion. In attribute value extension, Table 4 is imported. A new class is created for each entity with extended attribute value and each attribute to be extended, which is a subclass of the extended attribute value type. The attribute value is extended as an instance of the class. The extended class structure is shown in Figure 11.

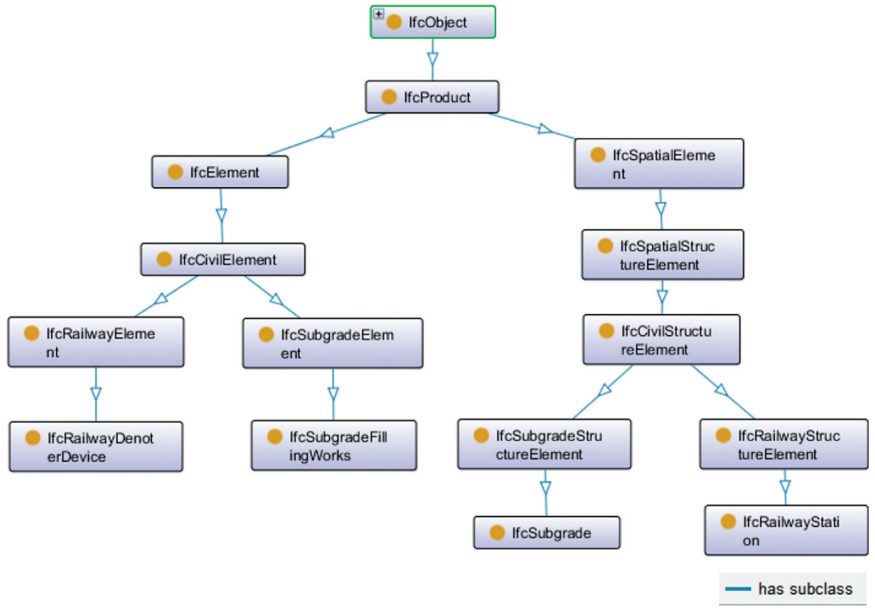


Figure 9: The extended entities in ifcOWL.

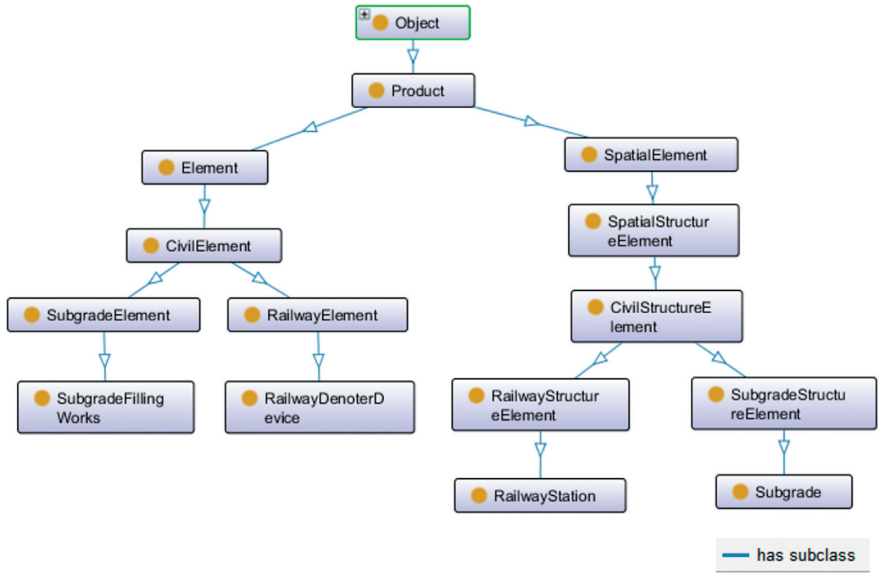


Figure 10: The extended entities in code ontology.

There are four steps to complete conversion from ifcOWL to code ontology according to the conversion method:

(1) Create a new class Characteristic and object property hasCharacteristic in code ontology.

(2) The entity with the extended attribute value is converted into classes in code ontology through entity conversion and the extended value is converted to a subclass of corresponding entity class.

(3) The extended attribute is converted into a subclass of Characteristic class, which is named after entity name + attribute name. The extended attribute value is a subclass of the class, whose name is prefixed with Char.

(4) The object property hasCharacteristic associates entity class and characteristic class.

The class structure of code ontology after conversion is shown in Figure 12.

5.3. Property Extension and Conversion. In property extension, Table 5 is imported. New property set class, named Pset + entity, is created. New relationship entity class is named after IfcRel + entity name. New property class is

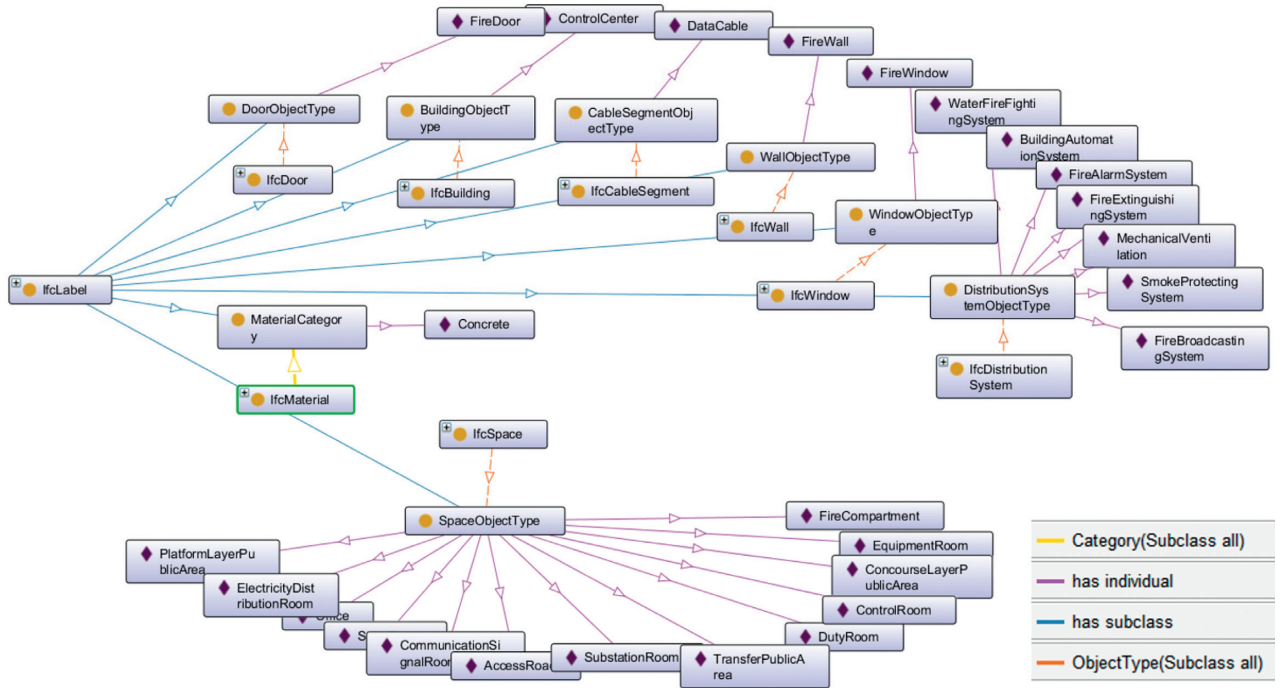


Figure 11: The extended attribute value in ifcOWL.

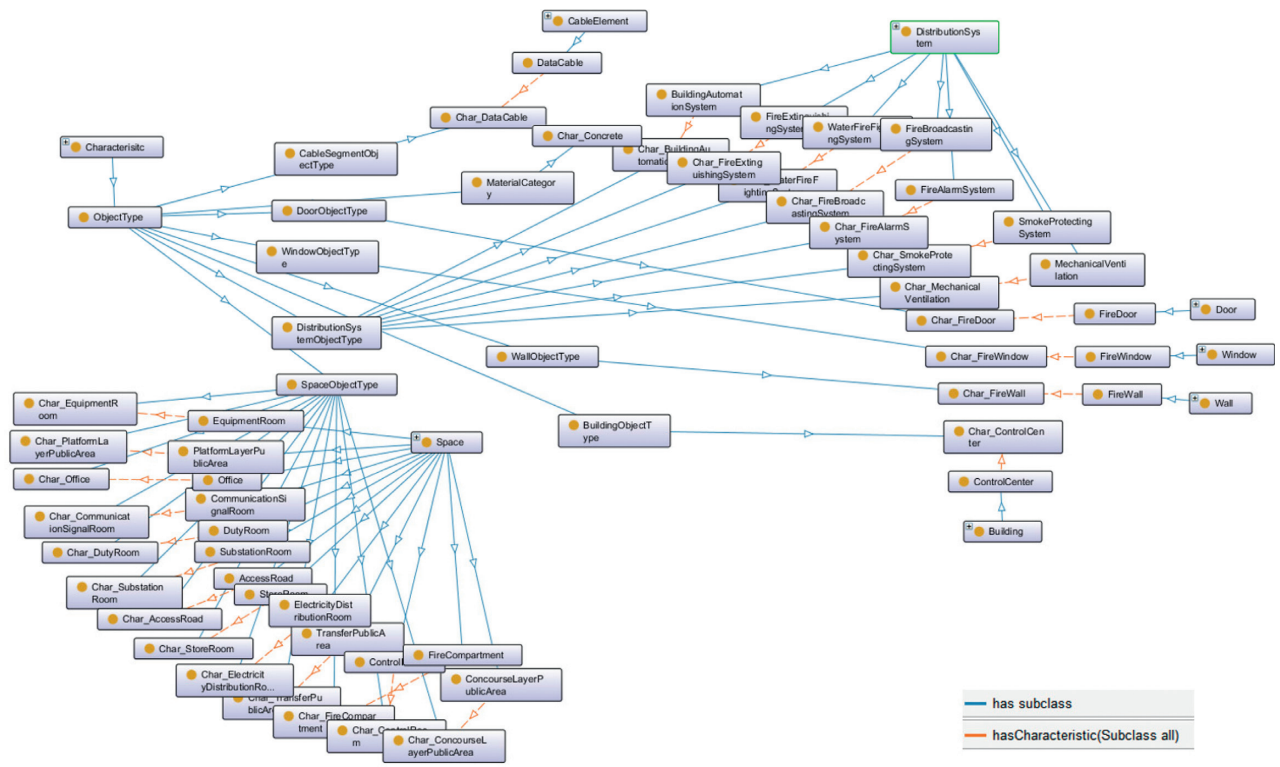


Figure 12: The extended attribute value in code ontology.

created whose name is P + property name. The property value type is restricted by property value type column in Table 5. Finally, the restriction of class is added to relate the content of extension.

A restriction column is added in Table 5 to indicate that the extended property is related to the restricted entity. When the value of PredefinedType attribute of IfcTransportElement is ESCALATOR, it means that it is an

```
<rdfs:subClassOf>
    <owl:Restriction>
        <owl:onProperty rdf:resource="RelatedObjects"/>
        <owl:allValuesFrom>
            <owl:Class>
                <owl:intersectionOf rdf:parseType="Collection">
                    <rdf:Description rdf:about="#IfcTransportElement"/>
                    <owl:Restriction>
                        <owl:onProperty rdf:resource="#PredefinedType"/>
                        <owl:hasValue rdf:resource="#ESCALATOR"/>
                    </owl:Restriction>
                </owl:intersectionOf>
            </owl:Class>
        </owl:allValuesFrom>
    </owl:Restriction>
</rdfs:subClassOf>
```

Figure 13: The restriction of RelatedObject in IfcRelEscalator.

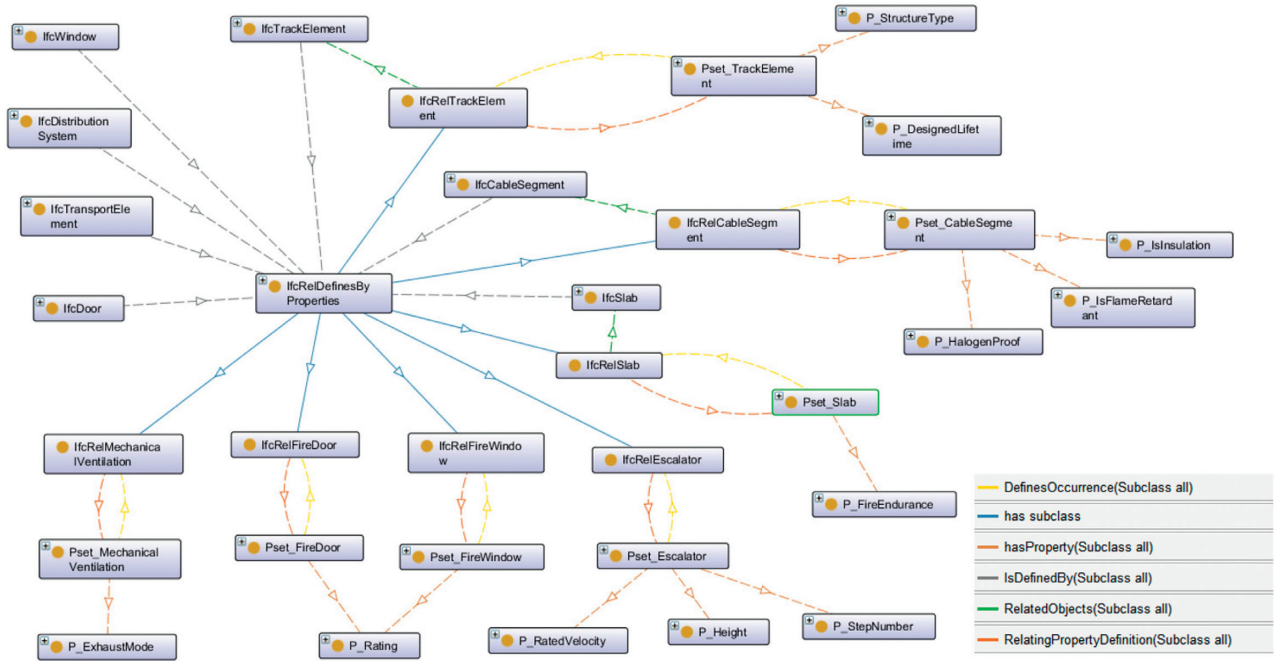


Figure 14: The property extension in ifcOWL.

escalator, and the extended property Height is semantically related to the escalator. Therefore, if there is restriction, it need to be added in class. The restriction of RelatedObject in the extended relationship entity IfcRelEscalator of IfcTransportElement is shown in Figure 13, namely,∀ RelatedObject (IfcTransportElement and (PredefinedType {ESCALATOR})), which means that RelatedObject must relate IfcTransportElement whose value of PredefinedType is ESCALATOR.

The extended class structure in ifcOWL is shown in Figure 14.

There are four steps to convert ifcOWL into code ontology:

(1) Create a new class Property, unit class Units, object property hasProperty and object property Unit. The range of hasProperty is Property. The domain of Unit is Property, and the range of Unit is Units.

(2) Entities with extended properties are converted into class in code ontology through entity conversion method.

(3) The extended property set converted into subclass of Property, which is named after entity name + PropertySet. The extended properties are converted into subclasses of property set class in code ontology. The class restriction is created to relate entity class and property class through hasProperty.

(4) The value type of NominalValue of the extended property is converted into Units class by mapping Table 1. The class restriction is created to relate property class and unit class through Unit.

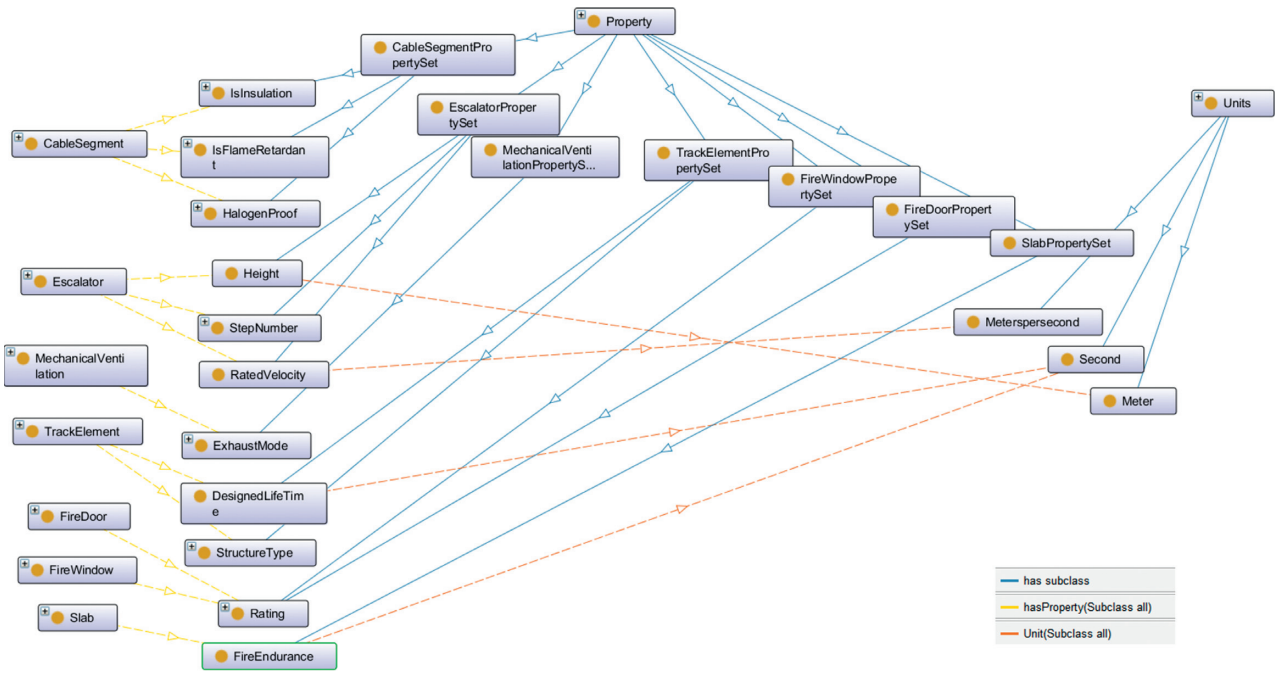


Figure 15: The property extension in code ontology.

The code ontology class structure is shown in Figure 15.

The conversion of restricted entities is similar to attribute value conversion. Property class in code ontology is related to class which are converted from restricted entities. If restricted entities are new concept for code ontology, an additional step of attribute value extension is executed in Step 2.

Code ontology is generated as OWL and imported to Protege 4.0. Hermit1.4.3 inference engine included in protege is used to check the consistency of code ontology, and there is no problem.

5.4. Code Checking Experiment

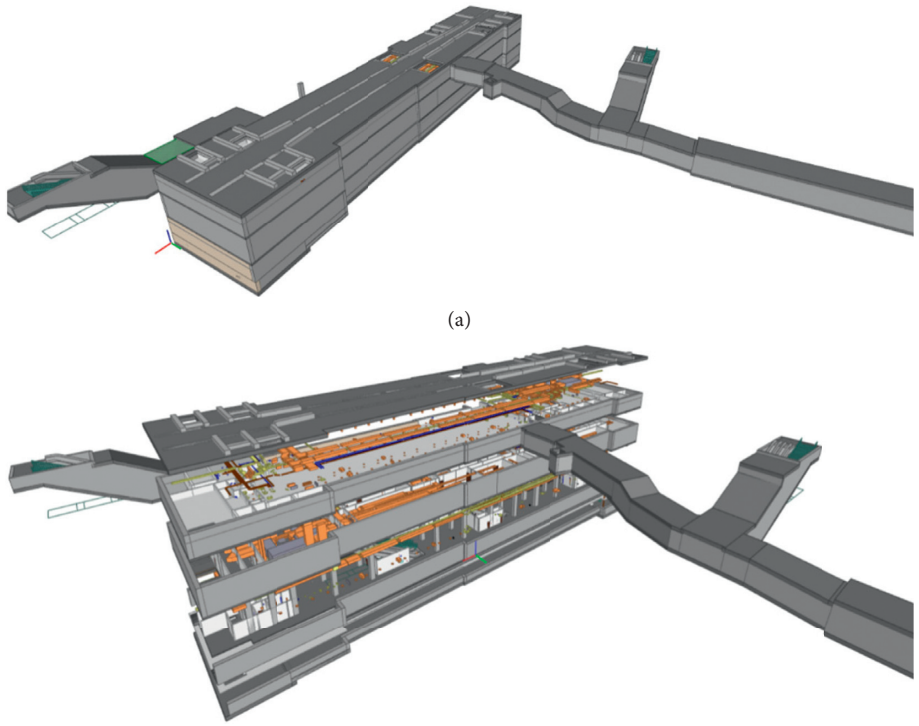
5.4.1. Integrity Checking. Integrity checking is to check whether the model has sufficient attributes to meet the requirements of specification compliance checking. The code ontology is a knowledge model of various relationships between concepts in the specification, which includes the definition of various attributes. Therefore, some relationships between the attributes and components in the ontology are converted to SPARQL statements, which are used to query the model information to check the missing information.

A real metro station model is selected for testing, whose data format is IFC. The model is shown in Figure 16 and display tool is BIM Vision [39]. In previous experiment, "Escalator" should have a "hasProperty" relationship with properties: Height, RatedVelocity, and StepNumber in code ontology. These definitions are converted to SPARQL statements.

The model information is mapped to the ontology instance. Escalators *A* and *B* in Figure 17 correspond to instances *A* and *B*, respectively. Integrity checking is executed by SPARQL query module on Protege and the results are shown in Figure 18, where CPEscalator indicates that the information is complete. The properties of escalator *A* is complete in Figure 17 and the "StepNumber" of escalator *B* is missing, which is the same as the checking result.

5.4.2. Compliance Checking. Compliance checking is to check whether the information of the model meets the specification requirements. The provisions of the specification can be converted into SWRL rules which are used to reason checking results in the code ontology. The experimental model is the same as previous model in Figure 16. The result class is defined in the code ontology to represent the checking result. The selection of experiment clause is 7.1.3. The design service life of the main structure of the ballastless track and the concrete track sleepers should not be less than 100 years.

The content of track is converted to SWRL rule: TrackElement(?a)^hasProperty(a,d)^DesignedLifeTime(d)^hasValue(?d,b)^swrlb:greaterThanOrEqual(b, 100)^hasProperty(a, s) ^ StructureType(s)^hasValue (?s,?BallastlessTrack) - > Compliance(?a). There are new result classes in ontology to represent the checking results. Compliance class indicates the inspection object is in compliance with the specification. The model information is mapped to the ontology instance and SWRL rules are reasoned through SWRL Tab in Protege; it is found that Track *B* is an instance of Compliance class as shown in Figure 19, which indicates that Track *B* conforms to the specification. The results correspond to IFC model in Figure 20, and the property of track *A* is displayed as 50, while the DesignedLifetime of track B is 100.



(a)

(b)

Figure 16: The metro model.

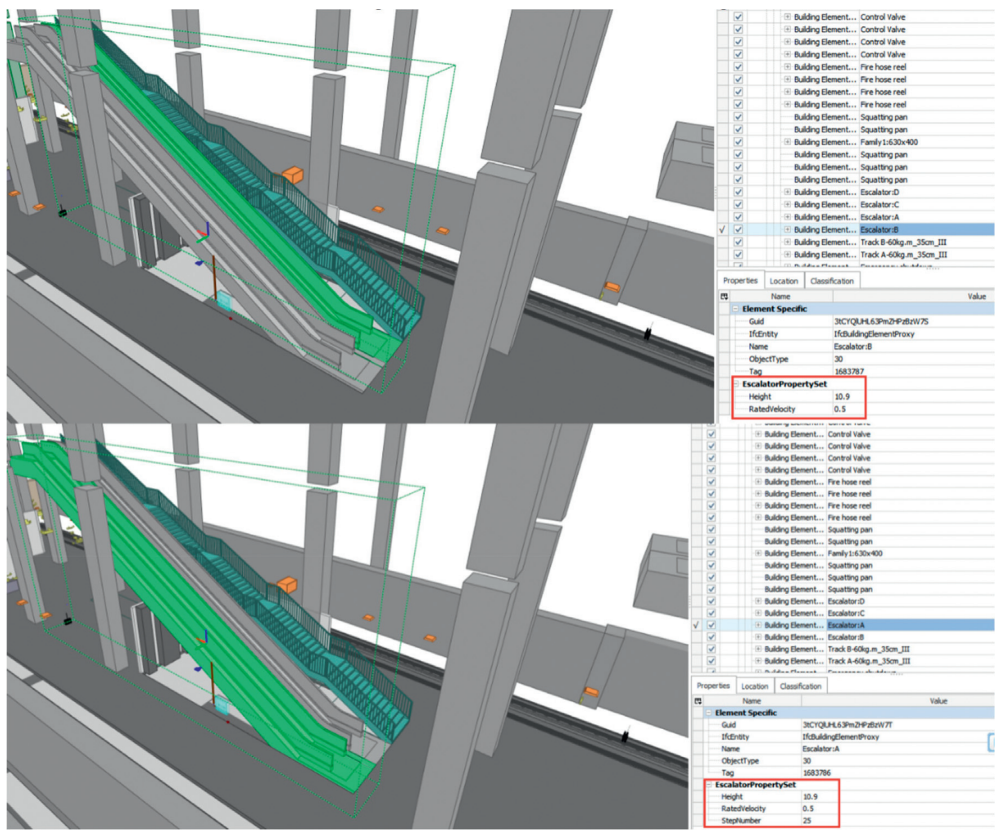


Figure 17: The property of Escalators *A* and *B*.

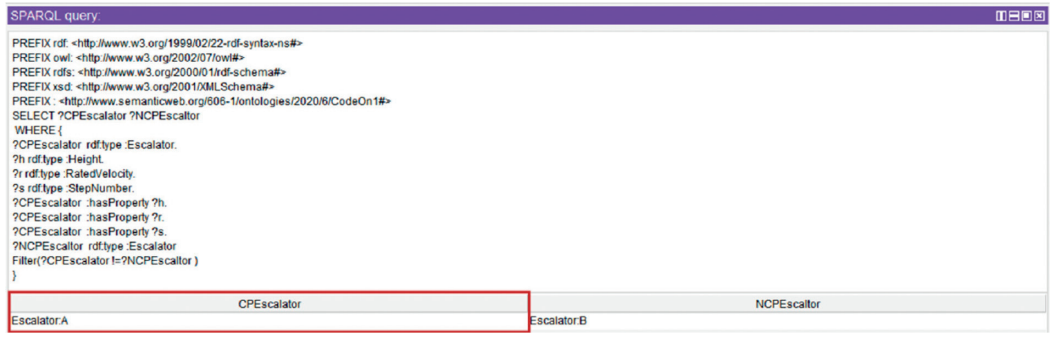


Figure 18: The integrity check results in Protege.

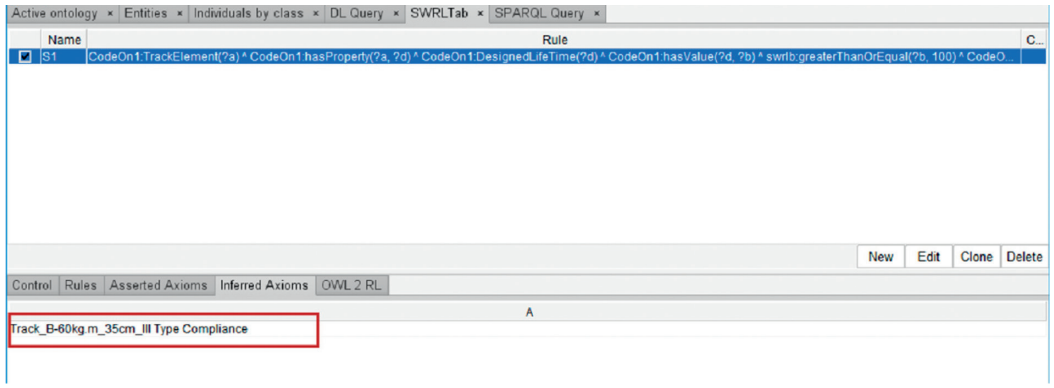
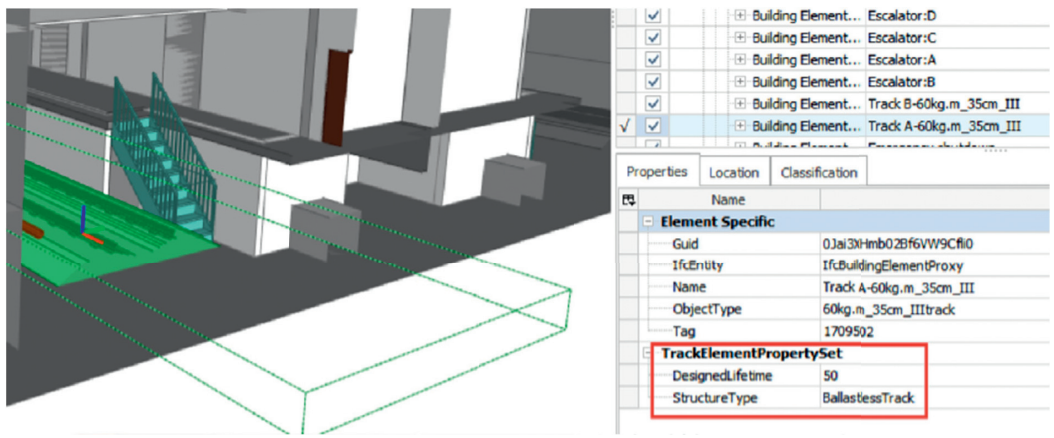
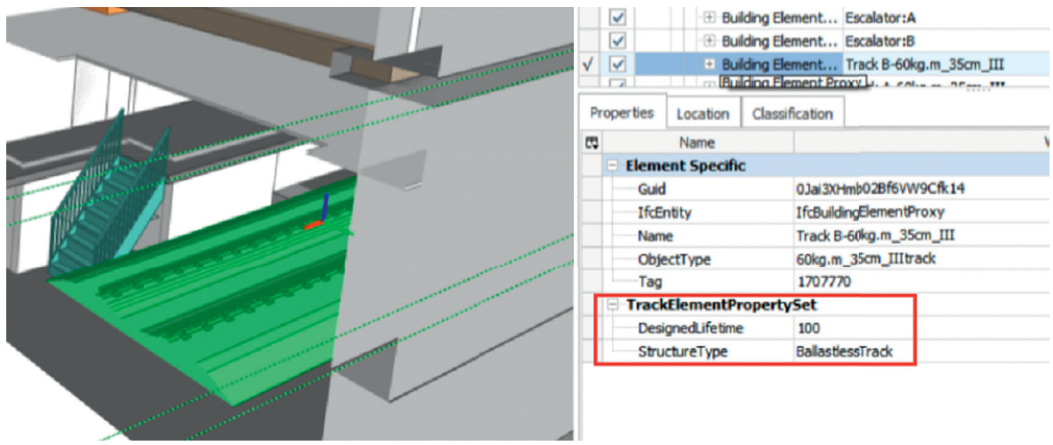


Figure 19: The compliance check results in Protege.



(a)

Figure 20: Continued.



(b)

Figure 20: The property of Tracks *A* and *B*.

ifcOWL is extended and converted to code ontology in the experiment, and then the code checking is implemented successfully by the code ontology. The feasibility of obtaining code ontology is proved by experiment. However, it is still necessary to manually mark the information in code as object in IFC in the preliminary work. At present, ontology construction cannot avoid the process of manual processing. It is necessary to map the model information to the code ontology in code checking, which can be relatively easy because the code ontology is converted by ifcOWL.

6. Conclusion and Discussion

6.1. Discussion. Compliance check is one of the necessary steps in the process of engineering construction. More and more researchers focus on new technologies such as semantic web to achieve automatic compliance check. A common method is to use the rules in the knowledge base to infer the results of compliance. Developing a code ontology that contains enough knowledge is the premise of compliance checking and a more important role is that the specification ontology can be used to check the integrity of the model. The establishment of domain ontology always is a difficult work, because it needs a lot of experts with rich knowledge to discuss and the consistency of the ontology needs to be guaranteed. IfcOWL, as one of the few ontologies in the field of building information, can provide the hierarchical structure of building information concepts and the definition of the relationship between concepts, which can provide a reference for code ontology. However, ifcOWL is actually aimed at general architectural concepts; the concepts in the code need to be extended in ifcOWL.

This paper proposes a code ontology generation method based on ifcOWL. ifcOWL is used to represent the code information in the process of extending ifcOWL and the concepts that cannot be represented by ifcOWL are extended. This part of work can reduce the domain knowledge requirements of researchers; because the code concepts have complex hierarchical classification relationships, researchers only need to consider the representation of concepts and the hierarchical classification relationship is implied in ifcOWL. Although various relationships between architectural concepts can be expressed in ifcOWL, the representation of some complex relationship may become more complex through ifcOWL, which increases the complexity of checking algorithms and affects the efficiency in the later code checking. In the process of ifcOWL conversion, the mapping patterns of attributes and classes are proposed. The domain experts' knowledge is not needed in this process and the two parts of the work of extending and converting are not related to each other. Ontology can be generated automatically after mapping is established. The method is also implemented in the metro design specification. However, it is still necessary to manually mark the information in code as object in IFC. Researchers can express the concepts in code with the help of IFC in this way instead of having a comprehensive knowledge of code information, which reduce the requirements of understanding of the code knowledge in the ontology construction. The extension of ifcOWL and the conversion of ifcOWL are separated. The extension is to express the semantics of code information through ifcOWL and the conversion is to generate code ontology with help of ifcOWL hierarchy. The separation of two parts can better reduce the coupling of work of ontology construction. Finally, the integrity and compliance of some components of the metro model are checked through the established metro design specification ontology. The integrity is checked through SPARQL querying and the compliance is checked through SWRL reasoning, which are common ontology applications. Although ifcOWL is highly extensible, if code ontology is only established by extended ifcOWL, it may cause ontology redundancy in the future and it is necessary to disambiguate in the process of ontology extension. These issues should be taken seriously in future work.

6.2. Conclusion and Future Work. For the absence of code ontology and the difficulty of code establishment, the ontology building method proposed in this paper which is

divided into two parts. One is to extend ifcOWL to represent code information; the concepts in ifcOWL are extended with the help of other domain IFC standard for more domain concepts definition and property extension improves the semantic relationship between property and entity in the specification. The other is to convert ifcOWL into code ontology. The extended ifcOWL is converted into code ontology, where extended entity and attribute value are converted into classes. The semantic representation of Property and Unit is defined in code ontology. Finally, an experts-dependency specification that is the code for design of Metro is selected as experimental content in this paper, where several clauses in different fields are selected to prove the feasible of the method. The consistency of established ontology is verified by HermiT inference engine. The compliance and integrity of the components of a metro model are checked with the help of established ontology, which proves the effectiveness of the ontology. The coupling degree of extension and conversion is very low, which can reduce the requirements of ontology builders for code domain knowledge and also improve the efficiency of specification ontology establishment. Code ontology is easy to extend in future because it is based on ifcOWL structure.

Although this paper achieves the generation of a part of code ontology which can be used to check integrity of information based on hierarchical classification of concepts in ifcOWL. However, there are many other relationships that are not represented in code ontology, such as spatial containment relationship and setting relationship. These relationships represent the logical relationships between different concepts in specification information and are main checkpoints in compliance check. Some of the relationships can be expressed in ifcOWL and we can add them to code ontology through the conversion of ifcOWL. Some of them cannot be expressed by ifcOWL, and we need to redefine it to conform to the established ontology. Therefore, this part of the ontology needs to be further extended and improved in future work, which can improve the effect of automatic compliance checking but also play a role in other semantic application scenarios.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (no. 51878556), the Key Scientific Research Projects of Shaanxi Provincial Department of Education (20JY049), and the Project of State Key Laboratory of Rail Transit Engineering Information (SKLKZ21-03).

References

[1] X. Tan, A. Hammad, and P. Fazio, "Automated code compliance checking for building envelope design," *Journal of Computing in Civil Engineering*, vol. 24, no. 2, pp. 203–211, 2010.

[2] P. Pauwels, D. Van Deursen, R. Verstraeten et al., "A semantic rule checking environment for building performance checking," *Automation in Construction*, vol. 20, no. 5, pp. 506–518, 2011.

[3] P. Pauwels, S. Zhang, and Y. C. Lee, "Semantic web technologies in AEC industry: a literature overview," *Automation in Construction*, vol. 73, pp. 145–165, 2017.

[4] S. H. Hong, S. K. Lee, and J. H. Yu, "Automated management of green building material information using web crawling and ontology," *Automation in Construction*, vol. 102, pp. 230–244, 2019.

[5] O. Balaban, E. Sezen, Y. Kilimci, and G. Cagdas, "Automated code compliance checking model for fire egress codes," *Digital Applications in Construction - ECAADe*, vol. 2, pp. 1–10, 2012.

[6] J. Choi, J. Choi, and I. Kim, "Development of BIM-based evacuation regulation checking system for high-rise and complex buildings," *Automation in Construction*, vol. 46, pp. 38–49, 2014.

[7] J. Gu, H. Zhang, and M. Gu, "Automatic integrity checking of IFC models relative to building regulations," *Proceedings of the International Conference on Internet Multimedia Computing and Service*, pp. 52–56, Xi'an, China, August 2016.

[8] B. Koo, S. La, N. W. Cho, and Y. Yu, "Using support vector machines to classify building elements for checking the semantic integrity of building information models," *Automation in Construction*, vol. 98, pp. 183–194, 2019.

[9] M. Shen, D. R. Liu, and Y. S. Huang, "Extracting semantic relations to enrich domain ontologies," *Journal of Intelligent Information Systems*, vol. 39, no. 3, pp. 749–761, 2012.

[10] A. G. Pérez, M. F. López, and O. Corcho, *Ontological Engineering with Examples from the Areas of Knowledge Management, E-Commerce and the Semantic Web*, Springer-Verlag, London, UK, 2011.

[11] R. Iqbal, M. A. A. Murad, A. Mustapha, and N. M. Sharef, "An analysis of ontology engineering methodologies: a literature review," *Research Journal of Applied Sciences, Engineering and Technology*, vol. 6, no. 16, pp. 2993–3000, 2013.

[12] M. Uschold, "Building Ontologies, towards a Unified Methodology," in *Proceedings of the 16th Annual Conference of the British Computer Society Specialist Group on Expert Systems*, pp. 16–18, Cambridge, UK, December 1996.

[13] Z. Ma and Z. Liu, "Ontology- and freeware-based platform for rapid development of BIM applications with reasoning support," *Automation in Construction*, vol. 90, pp. 1–8, 2018.

[14] P. Pauwels, T. Krijnen, W. Terkaj, and J. Beetz, "Enhancing the ifcOWL ontology with an alternative representation for geometric data," *Automation in Construction*, vol. 80, pp. 77–94, 2017.

[15] Y. Zhou, Y. Wang, L. Ding, and P. E. D. Love, "Utilizing IFC for shield segment assembly in underground tunneling," *Automation in Construction*, vol. 93, pp. 178–191, 2018.

[16] B. T. Zhong, L. Y. Ding, H. B. Luo, Y. Zhou, Y. Z. Hu, and H. M. Hu, "Ontology-based semantic modeling of regulation constraint for automated construction quality compliance checking," *Automation in Construction*, vol. 28, pp. 58–70, 2012.

[17] X. Xu and H. Cai, "Semantic approach to compliance checking of underground utilities," *Automation in Construction*, vol. 109, Article ID 103006, 2020.
[18] Y. Lu, Q. Li, Z. Zhou, and Y. Deng, "Ontology-based knowledge modeling for automated construction safety checking," *Safety Science*, vol. 79, pp. 11–18, 2015.
[19] T. R. Gruber, "A translation approach to portable ontology specifications," *Knowledge Acquisition*, vol. 5, no. 2, pp. 199–220, 1993.
[20] M. Uschold, "Towards a methodology for building ontologies," in *Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing IJCAI*, Montreal, Canada, April 1995.
[21] M. Grüninger and M. Fox, "Methodology for the design and evaluation of ontologies," in *Proceedings of the IJCAI'95, workshop on basic ontological issues in knowledge sharing*, Toronto, Canada, April 1995.
[22] S. Guus, W. Bob, and J. Wouter, "The KACTUS view on the 'O' word," in *Proceedings of the International Joint Conference on Artificial Intelligence*, Montreal, Canada, August 1995.
[23] P. C. Benjamin, C. P. Menzel, R. J. Mayer et al., *IDEF5 Method Report*, Knowledge Based Systems, Inc, TX, USA, 1994.
[24] M. F. Lopez, A. Gomez-Perez, J. P. Sierra, and A. P. Sierra, "Building a chemical ontology using METHONTOLOGY and the ontology design environment," *IEEE Intelligent Systems*, vol. 14, no. 1, pp. 37–46, 1999.
[25] S. Staab, R. Studer, H.-P. Schnurr, and Y. Sure, "Knowledge processes and ontologies," *IEEE Intelligent Systems*, vol. 16, no. 1, pp. 26–34, 2001.
[26] N. Noy and D. Mcguinness, "Ontology development 101: a guide to creating your first ontology," *Knowledge Systems Laboratory*, vol. 32, 2001.
[27] G. Ren, R. Ding, and H. Li, "Building an ontological knowledgebase for bridge maintenance," *Advances in Engineering Software*, vol. 130, pp. 24–40, 2019.
[28] F. H. Abanda, B. K. Foguem, and J. H. M. Tah, "Bim - new rules of measurement ontology for construction cost estimation," *Engineering Science and Technology, an International Journal*, vol. 20, no. 2, pp. 443–459, 2017.
[29] T. H. Beach, Y. Rezgui, H. Li, and T. Kasim, "A rule-based semantic approach for automated regulatory compliance in the construction sector," *Expert Systems with Applications*, vol. 42, no. 12, pp. 5219–5231, 2015.
[30] J. Niu and R. R. A. Issa, "Developing taxonomy for the domain ontology of construction contractual semantics: a case study on the AIA A201 document," *Advanced Engineering Informatics*, vol. 29, no. 3, pp. 472–482, 2015.
[31] H. Liu, M. Lu, and M. Al-Hussein, "Ontology-based semantic approach for construction-oriented quantity take-off from BIM models in the light-frame building industry," *Advanced Engineering Informatics*, vol. 30, no. 2, pp. 190–207, 2016.
[32] B. Zhong, C. Gan, H. Luo, and X. Xing, "Ontology-based framework for building environmental monitoring and compliance checking under BIM environment," *Building and Environment*, vol. 141, pp. 127–142, 2018.
[33] A. Borrmann, T. H. Kolbe, A. Donaubauer, H. Steuer, J. R. Jubierre, and M. Flurl, "Multi-scale geometric-semantic modeling of shield tunnels for GIS and BIM applications," *Computer-Aided Civil and Infrastructure Engineering*, vol. 30, no. 4, pp. 263–281, 2015.
[34] W. Terkaj and A. Šojić, "Ontology-based representation of IFC EXPRESS rules: an enhancement of the ifcOWL ontology," *Automation in Construction*, vol. 57, pp. 188–201, 2015.
[35] G. Costa and L. Madrazo, "Connecting building component catalogues with BIM models using semantic technologies: an application for precast concrete components," *Automation in Construction*, vol. 57, pp. 239–248, 2015.
[36] P. Pauwels and W. Terkaj, "EXPRESS to OWL for construction industry: towards a recommendable and usable ifcOWL ontology," *Automation in Construction*, vol. 63, pp. 100–133, 2016.
[37] Railway BIM data standard, *Railway BIM data standard*, 2015, https://www.buildingsmart.org/wp-content/uploads/2017/09/bSI-SPEC-Rail.pdf.
[38] P. R. C. Ministry of Housing and Urban-Rural Development, *Code for Design of Metro: GB50157-2013*, China Architecture & Building Press, Beijing, China, 2013.
[39] BIMVision, *BIMVision*, 2021, https://bimvision.eu/.