# Ontology-based assembly knowledge representation and process file generation

*Dan* Song[1], *Xin* Ye[2], *Wenrong* Wu[3], *Zhijing* Zhang[1,*], *Qimuge* Saren[1], *Jiahui* Qian[1], and *Dongsheng* Zhu[1]

[1]School of Mechanical and Vehicle, Beijing Institute of Technology, Beijing, China
[2]National Natural Science Foundation of China, Beijing, China
[3]Laser Fusion Research Center, China Academy of Engineering Physic, Mianyang, China

**Abstract.** Aiming at the problems of low assembly knowledge shareability and reusability as well as long generation cycle of assembly process, this paper proposes an ontology-based assembly knowledge representation method, and generates assembly process file based on this method. The assembly ontology, modelling through protégé software, has three central classes: AssemblyObject, AssemblyElement, and AssemblyTool. The assembly ontology is described in OWL language and the assembly knowledge concepts including classes and individuals are linked through properties. In addition, the assembly ontology in OWL language is parsed through Python's RDFLib library, and it is called and displayed in LabVIEW. Finally, the assembly process file containing assembly sequence and assembly process parameters is generated. This method realizes the formal description of assembly process knowledge at the semantic level and improves the shareability and reusability of assembly knowledge. Besides, the corresponding assembly process knowledge can be quickly queried and obtained through this method, improving the efficiency of assembly process planning, and providing intelligent assembly basic knowledge.

**Keywords:** Ontology, Knowledge representation, Assembly ontology, Assembly process.

## 1 Introduction

With the vigorous development of Industry 4.0 in Germany, Made in China 2025, and the Internet industry in the United States, intelligent manufacturing and intelligent assembly have attracted more and more attention. The National Institute of Standards and Technology (NIST) proposes a strategic planning called Smart Assembly, which aims to quickly plan the assembly design of new products and achieve flexible assembly of variable batch products. Now, there are many domestic and foreign universities and research institutes exploring the application of artificial intelligence in the field of assembly. Based

* Corresponding author: zhzhj@bit.edu.cn

on the production rules of disassembly and replacement of tools and fixtures, Y.F. Huang [1] et al. search out the assembly process through pictures intelligently, and realize the automatic generation of assembly process information. The research on modelling assembly knowledge using semantic ontology has attracted more and more attention. For example, K young-Yun Kim [2-3] of Wayne State University in the United States proposed an ontology-based assembly design framework, which includes variable conceptual entities such as products, assemblies, assembly components, parts, sub-assemblies, assembly features, connection structures, connection characteristics and so on. The assembly ontology is used to realize assembly design, information sharing and virtual assembly simulation. Lihong Qiao of Beihang University [4] proposes a geometry-enhanced assembly process ontology model, using semantic ontology technology to establish the description method, data structure and relationship description of product geometric information in the assembly process; moreover, an assembly process decision-making framework based on geometry-enhanced assembly process ontology is established. RAPHAEL establishes an ontology program that expresses the geometry of the assembly, and maps the geometric neutral file STEP AP203 with the ontology model to realize the information conversion between the geometric information of the 3D model and the semantic model [5].

## 2 Ontology-based assembly knowledge representation

### 2.1 Ontology technology

The definition of ontology currently generally accepted is: ontology is a clear formal specification of shared conceptual model. This definition includes four layers of meaning: conceptual model, formalization, clarity, and sharing. That is, the ontology is a model obtained by abstracting the related concepts of some phenomena in the objective world, and these concepts and the constraints of used to these concepts have clear definitions. The ontology embodies the approved knowledge of relevant fields, and can be processed by computer.

Web Ontology Language (OWL) is a web ontology representation language released by the World Wide Web Consortium (W3C). It is based on description logic (DL) and can better meet the requirements of ontology knowledge representation and reasoning complexity in terms of formal definition. This paper adopts OWL-DL as the modelling language of assembly ontology model. OWL-DL ontology mainly consists of three elements: Class, Property and Individual, corresponding to the entity, attribute and instance in the description logic.

### 2.2 Assembly ontology modelling

In this paper, ontology application software Protege is used to model assembly ontology of assembly process which has three central classes: (a)AssemblyObject, (b)AssemblyElement and (c)AssemblyTool. The specific class hierarchy is shown in figure 1. Classes in this paper usually start with a capital letter and there are no spaces in the class name.

The class “AssemblyObject” represents all products and action execution objects involved in the assembly process. Action execution objects include all kinds of assembly process positions and feed boxes. The product is generally composed of parts and components, the latter of which contain standard parts and non-standard parts.

Subclasses of “AssemblyElement” are mainly “AssemblyAction” and “AssemblyFeature”. The “AssemblyAction” class represents various assembly actions, and

it has subclasses such as MoveTo, Clamp, Adsorb, etc. While the “AssemblyFeature” class represents different assembly features of parts, for example shaft_hole connection.

The class “AssemblyTool” presents specific tools executing assembly actions of both the automatic assembly process and manual assembly process. It has subclasses of automatic assembly tools, namely, loading and unloading module, assembly execution module, visual alignment module, assembly platform module and auxiliary mechanism module, and manual assembly tools, such as hammer, wrench and screwdriver.

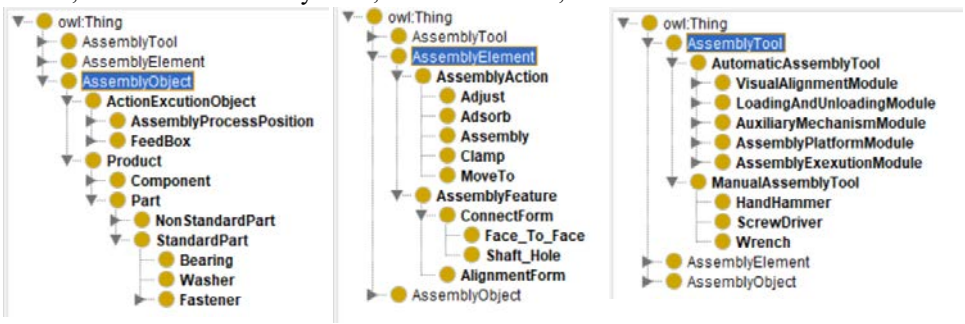


**Fig. 1.** Class hierarchy of assembly ontology.

Taking the linkage mechanism of a product as an example, the corresponding individuals are created in the classes established above, and every individual represents an independent instance. For example, add Nut01, Nut02, Nut03 and Nut04 individuals in Nut class. The individuals in this paper usually start with a capital letter and are marked with Arabic numerals at the end.

OWL Properties represent relationships. There are two main types of properties, Object properties and data properties. Object property is the relationship between two individuals, in other words, object property links individual to individual. While the data property links individual to the XML schema data type value or RDF text, that is, data property describes the relationship between the individual and the data value. The properties in this paper start with a lowercase letter, and most properties begin with "has" or "is". Define object properties and data properties as shown in figure 2.

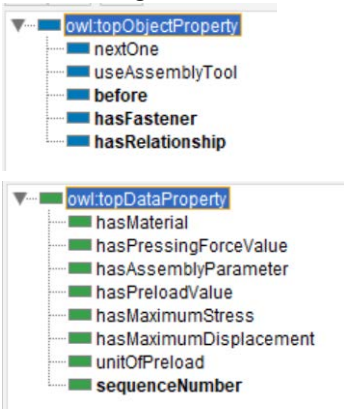


**Fig. 2.** Object properties and data properties of assembly ontology.

Next, the object property “before” is taken as an example to detail the property association.

“Before” is a transitive object property. Add “before” object property to corresponding part individuals. If individual part A has to be assembled before individual part B, for individual part A, add the property "before individual part B ". And for it is transitive, when

there is " individual A before individual B" and " individual B before individual C", " individual A before individual C" can be pushed out. In other words, if individual part A is assembled before individual part B, and individual part B is assembled before individual part C, then individual part A has to be assembled before individual part C.

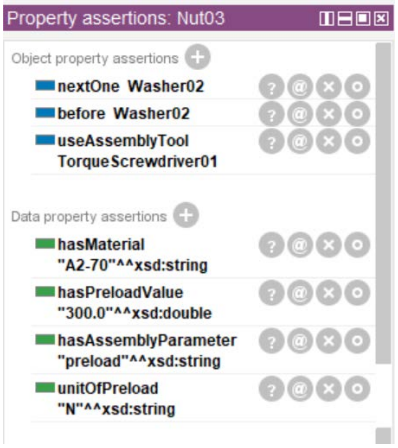


**Fig. 3.** Properties of individual Nut03.

According to the existing manual assembly experience of the linkage mechanism, the assembly sequence of this product is:

TensionBase01->Bearing02->Bearing04->Hook01->ShaftComponent01->Nut01->Hook02-> ShaftComponent02->Nut03->Washer02->Nut02->Washer04- >Nut04

Add various object properties and data properties to each individual. As shown in figure 3 the various properties were assigned to the individual Nut03. The Nut03 has the object property "nextOne Washer02", the object property "before Washer02", the object property "useAssemblyTool TorqueScrewdriver01", the data property "hasMaterial A2-70", and data property "hasPreloadValuel 300N".

In the OntoGraf plug-in of Protégé, the assembly sequence of each part can be seen as shown in figure 4. The assembly sequence is presented in the form of dashed connecting lines with arrows from the priority assembled part to the next assembled part, and the dashed connecting lines with arrows represent the object property “before”. The dashed connecting line and its end-to-end individuals form a triple "Nut03—before—Washer02" as shown in the figure. The figure clearly shows the assembly sequence of the linkage mechanism.

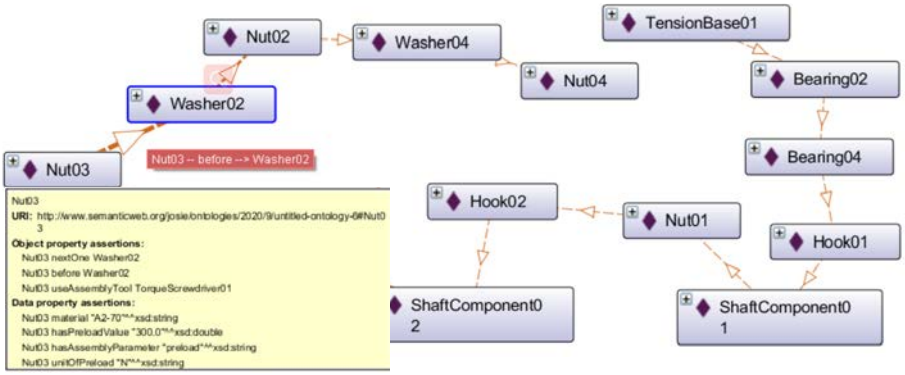


**Fig. 4.** Assembly sequence of the linkage mechanism shown in OntoGraf.

# 3 Ontology-based assembly process file generation

The assembly knowledge is stored in the assembly ontology, forming a huge entity relationship network, forming a knowledge graph, as shown in figure 4. However, this form of knowledge graph is not friendly to experts in the assembly field. For this reason, the authors parse the assembly ontology and a general assembly process file is generated finally.

## 3.1 OWL language parsing

The basic thinking of OWL is to express a simple resource statement by a triple composed of subject, predicate and object. The subject of the triple usually corresponds to the individual in the ontology, while the predicate corresponding to the property, and the object corresponding to the value of the property. For example, the triple "Nut03—before—Washer02" has “Nut03” as subject, “before” as predicate, and “Washer02” as subject. Based on this, OWL language can be parsed, and the assembly knowledge required by knowledge experts and domain experts can be extracted.

RDFLib is a open source library of Python for processing RDF. It contains a parser and serializer for RDF/XML. The primary interface that RDFLib exposes for working with RDF is a Graph, the set of the above triple of subject, predicate and object. RDFLib can perform several kinds of operations to the triples in RDF as well as OWL since OWL is developed on the base of RDF. For example, by querying the before property, all triples that have “before” as subject are obtained. And store the "subject", "predicate" and "object" of the extracted n triples, as shown in Table 1.

**Table 1.** Triples with the predicate as “before”.

| Subject | Predicate | Object |
|---|---|---|
| TensionBase01 | before | Bearing02 |
| Bearing02 | before | Bearing04 |
| Bearing04 | before | Hook01 |
| Hook01 | before | ShaftComponent01 |
| ShaftComponent01 | before | Nut01 |
| Nut01 | before | Hook02 |
| Hook02 | before | ShaftComponent02 |
| ShaftComponent02 | before | Nut03 |
| Nut03 | before | Washer02 |
| Washer02 | before | Nut02 |
| Nut02 | before | Washer04 |
| Washer04 | before | Nut04 |

According to the transitivity of the “before” property in the assembly ontology and the actual assembly experience of single thread, it can be known that in the triple of the "subject before object", except for the initial part, all other individuals of the subject exist in another triple as object. Therefore, the individual that exists only as the subject but not as the object in above triples is the initial assembly part.

After finding the initial assembly part, sort all the assembly parts. Define an empty array “partorder”, the first element of which is the initial assembly found above, and starting from the second element, the current assembly part is the object individual part in the triple where the previous part individual is the subject. Access all the triples in Table 1 to obtain

the “partorder” array with n+1 elements, and then find the complete assembly sequence of n+1 assembly parts.

Next, the specific assembly process of each assembly part is parsed, including assembly tools, assembly parameters and parameter values. It is similar to the parsing method of the previous assembly sequence triple, that is, the RDFLib toolkit is called to parse the triples whose properties are "useAssemblyTool", "hasAssemblyParameter", "hasPreloadValue" and "hasPressingForceValue", and the corresponding subjects and objects of which are stored.

### 3.2 Assembly process file generation

After finding all the assembly knowledge required through the RDFLib toolkit, the knowledge needs to be presented in a suitable form. LabVIEW is a graphical programming language, suitable platform for developing graphical application, and it has a friendly human-computer interaction interface. Therefore, LabVIEW is chosen as the development software for assembly process generation.

LabVIEW2018 and later versions come with a python interface, which can call python programs. Figure 5 shows the program diagram of LabVIEW2019 calling Python3.6. A python session is open first and a session is created to provide input for a series of subsequent operations, followed by python functions encapsulated and python scripts called. The python session is closed at the end of the program to prevent memory leaks.

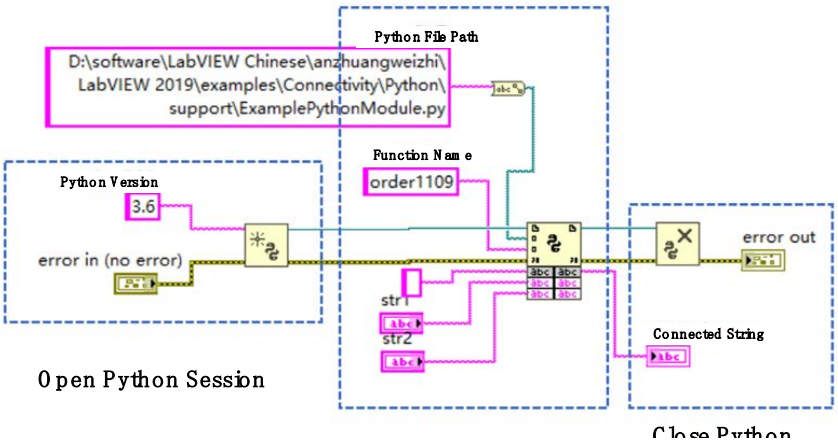


**Fig. 5.** Program diagram of Labview2019 calling python3.6.

Through the program of LabVIEW calling python parsing program of section 3.1, assembly ontology is parsed and assembly knowledge is visualized. Figure 6 shows the assembly process display interface after calling python program to visualize the assembly ontology. In the display interface, select the path of the assembly ontology (the form of .owl file) and assembly ontology parsing program (the form of .py file) in the top left corner of the interface, and type the version of the python running program in the upper right corner of the interface. In this example, the Python version is 3.8. Click the “showOWL” button in the top right of the interface to display the assembly ontology of OWL language in the right side of the interface, and click the “Parsing” button in the top right of the interface to display the assembly process with assembly sequence and assembly parameters in tabular form on the left side of the interface. In addition, the program

generates an independent assembly process file and saves it in the computer in excel format, as shown in figure 7, and displays the storage path of the assembly process file in the lower left corner of the display interface in figure 6.

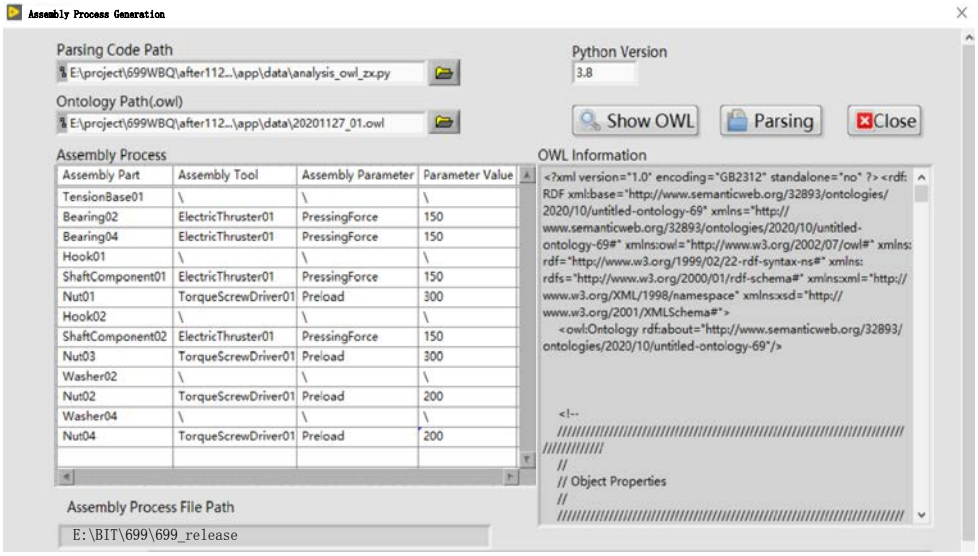


**Fig. 6.** Program diagram of Labview2019 calling python3.6.

## 4 Conclusion

Aiming at the problems of low shareability and reusability of assembly knowledge and long assembly process generation cycle, this paper proposes an ontology-based assembly knowledge representation method, and generates assembly process file based on this method. The assembly ontology is described in OWL language and modelled by Protégé software, and has three central classes: AssemblyObjects, AssemblyElement, and AssemblyTool. Corresponding individuals and properties are established in the assembly ontology. The latter includes object properties and data properties. Object property connect individual and individual, and data property connect individual and data value. Parse the assembly ontology of OWL language through the graph function in Python's RDFLib library, extracting all triples in OWL with “before” as the predicate, sorting the assembly parts obtaining from the triples to get the assembly sequence. And the triples whose properties are "useAssemblyTool", "hasAssemblyParameter", "hasPreloadValue" and "hasPressingForceValue" are extracted in the same way, to find the assembly-related process knowledge. Finally, the python parsing program is called in LabVIEW to display the assembly-related process knowledge in the interface and to generate the assembly process file containing the assembly sequence and process parameters. This method realizes the formal description of assembly process knowledge at the semantic level, improving the shareability and reusability of assembly knowledge, and realizes rapid query and acquisition of corresponding assembly process knowledge, improving the efficiency of assembly process planning, which lays a strong knowledge base for intelligent assembly.

The work was supported by the National Key Research and Development Program of China (No.2019YFB1310900) and Open Fund of State Key Laboratory of Intelligent Manufacturing System Technology (CN).

## References

1. A. Delchambre. Computer -aided Assembly Planning[M]. CHAPMAN & HALL, 1989: 12-1.
2. Kim K Y, Chin S, Kwon O, et al. Ontology-based modeling and integration of morphological characteristics of assembly joints for network-based collaborative assembly design[J]. Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 2009, 23(01):71.
3. Kim K Y, Chin S, Kwon O, et al. Ontology-based modeling and integration of morphological characteristics of assembly joints for network-based collaborative assembly design[J]. AI EDAM, 2009, 23(1): 71-88.
4. Lihong Qiao, Yixin Zhu, ANWER Nabil. Geometrically enhanced assembly process ontology modelling[J]. Chinese Journal of Mechanical Engineering, 2015, 51(22).
5. RAPHAEL B, KRIMA S, RACHURI S, et al. Onto STEP: Enriching product model data using ontologies[J]. Computer-Aided Design, 2012, 44(6) 575-590.