Artificial Intelligence for Engineering Design, Analysis and Manufacturing

cambridge.org/aie

A semantic similarity-based method to support the conversion from EXPRESS to OWL

Yan Liu[1], Qingquan Jian[2] and Claudia M. Eckert[3]

[1]Department of Computer Science, College of Engineering, Shantou University, Shantou 515063, China; [2]School of Computer Science and Technology, Changchun University of Science and Technology, Changchun 130022, Jilin, China and [3]School of Engineering and Innovation, The Open University, Walton Hall, Milton Keynes MK7 6AA, UK

Research Article

Cite this article: Liu Y, Jian Q, Eckert CM (2023). A semantic similarity-based method to support the conversion from EXPRESS to OWL. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **37**, e24, 1–21. https://doi.org/10.1017/S0890060423000185

Received: 31 August 2022
Revised: 4 April 2023
Accepted: 6 July 2023

Keywords:
EXPRESS; natural language processing; product data; semantic similarity; Web Ontology Language (OWL)

Corresponding author:
Yan Liu; Email: yanliu@stu.edu.cn

Abstract

Product data sharing is fundamental for collaborative product design and development. Although the STandard for Exchange of Product model data (STEP) enables this by providing a unified data definition and description, it lacks the ability to provide a more semantically enriched product data model. Many researchers suggest converting STEP models to ontology models and propose rules for mapping EXPRESS, the descriptive language of STEP, to Web Ontology Language (OWL). In most research, this mapping is a manual process which is time-consuming and prone to misunderstandings. To support this conversion, this research proposes an automatic method based on natural language processing techniques (NLP). The similarities of language elements in the reference manuals of EXPRESS and OWL have been analyzed in terms of three aspects: heading semantics, text semantics, and heading hierarchy. The paper focusses on translating between language elements, but the same approach has also been applied to the definition of the data models. Two forms of the semantic analysis with NLP are proposed: a Combination of Random Walks (RW) and Global Vectors for Word Representation (GloVe) for heading semantic similarity; and a Decoding-enhanced BERT with disentangled attention (DeBERTa) ensemble model for text semantic similarity. The evaluation shows the feasibility of the proposed method. The results not only cover most language elements mapped by current research, but also identify the mappings of the elements that have not been included. It also indicates the potential to identify the OWL segments for the EXPRESS declarations.

Introduction

STandard for Exchange of Product model data (STEP) has long provided a reliable format for the exchange of data in product development processes. As its descriptive language EXPRESS lacks the richness to describe formal semantics for more complex knowledge representation with both geometry (such as size and shape) and non-geometry (such as function and behavior) information (Barbau et al., 2012; Qin et al., 2017; Kwon et al., 2020), many researchers suggest to convert STEP models to Web Ontology Language (referred as OWL) models to explicitly express, represent, and exchange semantic information. To assist the conversion process, this paper proposes a method that automatically locates the corresponding language elements of EXPRESS and OWL based on semantic analysis of the official reference manuals. It could also be applied to the identification of OWL segments for the EXPRESS declarations on the data model level.

Product data sharing and exchange supports collaborative product design and development (Eslami et al., 2018; Andres et al., 2021). To enable this exchange, the International Organization for Standardization (ISO) has developed STEP, an international standard referenced as ISO 10303, to provide a mechanism capable of describing products, independent of any particular product modeling environment. The product data models in STEP are described by EXPRESS, a formal information requirement specification language which consists of language elements allowing unambiguous data definitions and their constraints. EXPRESS has been implemented in many computer-aided systems (e.g., CAD, CAE, and CAM) and product data/lifecycle management systems.

To include more semantic information, researchers have suggested using models described in OWL. OWL proposed by the World Wide Web Consortium (W3C) is an ontology language for the Semantic Web with a formally defined meaning. It is designed to represent rich and complex knowledge about objects and their relations by using classes, properties, individuals, and data values. Product data models based on OWL have been applied to enhance the semantic interoperability in the manufacturing industry (Ramos, 2015; Alkahtani et al., 2019; Fraga et al., 2020).

To translate EXPRESS-based models to OWL ontologies could enhance product data sharing and exchange with rich semantic information, but this process requires the mapping

© The Author(s), 2023. Published by Cambridge University Press. This is an Open Access article, distributed under the terms of the Creative Commons Attribution licence (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted re-use, distribution and reproduction, provided the original article is properly cited.

between the two languages. Some researchers have suggested solutions, for example, Krima et al. (2009), Barbau et al. (2012), and Pauwels and Terkaj (2016). However, most research is based on the experts' understanding of the two languages and finds the mapping relations manually. The experts have to look up the language elements among hundreds of items, which is an effort intensive and error prone mapping process.

The mapping process identifies the pairs of language elements with similar usage from the two modeling languages, respectively, the definitions of which are all described in the language manuals. In this sense, the mapping is a semantic matching problem of the texts in the two language manuals. The semantic matching also applies to the matching between the data models built upon the language elements. This can be handled by natural language processing (NLP) techniques. This paper proposes an EXPRESS-to-OWL 2 framework, which analyzes the similarities in terms of three aspects, that is heading semantics, text semantics, and heading hierarchy. A method combining Random Walks (RW) and Global Vectors for Word Representation (GloVe) models was designed to calculate the heading similarities. A model based on Decoding-enhanced BERT with disentangled attention (DeBERTa) is suggested for text similarity computation. The similarities from heading, text semantics, and heading hierarchy are then aggregated for an overall similarity of a pair of language elements by an Analytic Hierarchy Process–Simple Additive Weighting (AHP-SAW) method. The top four potential mapping elements of OWL 2 are provided to support the conversion from EXPRESS to OWL, which narrows down the look-up scope. The application shows that the results cover most already identified mappings in literature along with also providing new mappings. The proposed model is also applied to mapping on the data model level. Data model consists of EXPRESS declarations or OWL expressions, which provides the template and governs the format of product data in a physical file. An example of a linear bearing in stamping outer ring is used to illustrate the identification of OWL expressions for EXPRESS declarations.

The rest of the paper is organized as follows. Section "Related works" discusses the literature on the mapping between EXPRESS and OWL 2. Section "Fundamentals for method development" introduces the research approach and the fundamental elements of the proposed method that is the language manuals and RW, GloVe, and DeBERTa models. The proposed framework is presented in Section "Multiple similarity-based method for mapping", followed by its evaluation in Section "Evaluation and mapping results". Section "Conclusion" concludes the paper.

Related works

The conversion of EXPRESS-based models to OWL ontologies enhances the exchange of semantic information. Researchers have proposed mapping rules mainly based on a manual process to link the languages elements. The earliest work that we have found is the research of Schevers and Drogemuller (2005), which takes Industry Foundation Classes (IFC), a particular EXPRESS schema, as an example and converts several EXPRESS elements to OWL, elements such *Entity* to *Class* and *List* to *List*. Later more structured mapping approaches are proposed by Agostinho et al. (2007), Beetz et al. (2008), and Krima et al. (2009), where EXPRESS elements are grouped into *Schema*, *Entity*, *Attribute*, simple, constructed, and aggregated data types and the corresponding OWL elements are presented for the EXPRESS elements under each group. Following Beetz's work in transforming IFC to an "ifcOWL" ontology, Terkaj and Šojić (2015) and Pauwels et al. (2017) provide a more detailed conversion procedure. These efforts on developing ifcOWL keep the resulting OWL ontology as close as possible to the original EXPRESS schema of IFC (Pauwels et al., 2017) and have been used to build knowledge-based models for storing geometric information (Farias et al., 2018; González et al., 2021; Wagner et al., 2022). Different from ifcOWL, "OntoSTEP" from Krima et al. (2009) serves a broader scope, based on which Barbau et al. (2012) developed a plug-in for ontology editor Protégé. OntoSTEP integrates the concepts and semantic relationships into the geometry-enhanced ontology model and has been used to build a semantic-rich product model for a range of applications, such as program-generation for robotic manufacturing systems (Zheng et al., 2022), knowledge graph establishment for product quality assurance (Kwon et al., 2020), and feature extraction for assembly sequence planning (Gong et al., 2021).

Different researchers have applied slightly different interpretations to the mapping. As illustrated in Table 1, the first column lists the main language elements of EXPRESS and the remaining columns show the corresponding OWL elements proposed by different researchers. "–" denotes an unaddressed element. It can be seen that some conversions have converged such as *Entity* to *Class, while others are* handled differently. For example, Agostinho et al. (2007) consider direct conversion for simple data types, as they have equivalents in OWL like *string* to *string*. However, this is not always the case. *Number* and *real* are represented in slightly different ways in OWL and Krima et al. (2009) translated them into *decimal* and *double*. Instead of a one-to-one mapping, Beetz et al. (2008) and Terkaj and Šojić (2015) create new classes for these types by setting properties. The difference of the mapping approaches lies in the different understanding and choices of the researchers. In addition, it seems that some elements are not covered. For example, there are also logical operators (*NOT*, *AND*, *OR*, and *XOR*) in EXPRESS, which are not mentioned in the reviewed research.

The conversion requires the researchers to learn the whole reference manuals, keep the concepts in mind, and be able to recall where the corresponding elements are located in the reference manuals. This process not only requires effort but is also prone to misunderstandings. A computerized method could address this by automatically locating the corresponding language element pairs of EXPRESS and OWL 2. However, only limited research exists on automatic mapping. This paper aims to fill this gap by proposing a method based on NLP techniques to locate the language elements and further to match the expressions based upon the languages.

Fundamentals for method development

NLP techniques enable building an intelligent method to support the conversion from EXPRESS to OWL and reduce the effort of manually mapping. This section first describes the steps of the research, and how the two language reference manuals were analyzed to show the suitability of applying NLP techniques. At the end, the NLP models used in this research are introduced.

Overview of the research approach

As illustrated in Figure 1, this research started with identifying the research topic through examining the literature, and then built the translation method based on NLP techniques followed by

https://doi.org/10.1017/S0890060423000185 Published online by Cambridge University Press

Table 1. The mapping results of the existing research

EXPRESS elements		OWL elements				
		Schevers and Drogemuller (2005)	Agostinho et al. (2007)	Beetz et al. (2008)	Krima et al. (2009) and Barbau et al. (2012)	Terkaj and Šojić (2015) and Pauwels et al. (2017)
Schema		–	Ontology	Ontology	Ontology	Ontology
Entity		Class	Class	Class	Class	Class
Instance		–	Individual	Individual	Individual	Individual
Attribute		Slots	ObjectProperty DataProperty	Object Property	ObjectProperty DataProperty	Functional object property with specified domain and range
Simple data type	Number	–	–		Decimal	
	Real	Class with a property			Double	
	Integer	–	Direct	Class has a single OWL	Integer	Class with a restriction
	Logical	–	Conversion	DatatypeProperty	–	On an OWL: DatatypeProperty
	Boolean	–			Boolean	
	String	–			String	
	Binary	–			–	
Constructed data type	Enumeration	–	OneOf	OneOf	ObjectOneOf EquivalentClasses	Equivalence to a one of collection of OWL: NamedIndividual items
	Select	Entities or other types	UnionOf	The union domain range of OWL: ObjectPropertys	EquivalentClasses ObjectUnionOf	Equivalence to a union of collection of OWL classes
Aggregated data type	Set	Set constraints and allow multivalues	Create new		ObjectProperty	Nonfunctional object property with specified domain and range
	Bag	–	Classes		Create new class	–
	List	List		Class with two objects	ObjectProperty	Class by setting
	Array	–		Properties	EquivalentClasses	Properties
Constraint operators	OneOf	–	–	–	EquivalentClasses ObjectUnionOf DisjointClasses	–
	AndOr	–	–	–	EquivalentClasses ObjectUnionOf	–
	And	–	–	–	EquivalentClasses ObjectIntersectionOf	–

evaluations. Figure 1 shows the logical sequence of the steps, the research was carried out with iterations where the evaluation led to improvement in the method.

The *method development* started with the analysis of the two languages and the analysis of the structure of reference manuals. This informed the selection of NLP techniques. Figure 2 gives an overview of the steps taken in developing the method.

The *method evaluation and application* was on three levels:

- *Level 1: evaluates the performance of the method*. The Pearson correlation coefficient and the Spearman correlation coefficient were used to measure the capability of NLP-based models for calculating the similarity. Three datasets (MC, RG, and Agirre) were used for word mappings and the STSBenchmark dataset for sentences mapping (see Sections "The combined RW and GloVe models" and "The DeBERTa ensemble").
- *Level 2: checks the mapping results* of the language elements of EXPRESS and OWL were carried out and compared to the literature (see Section "The mapping results of language elements and discussion").
- *Level 3: applies the proposed method to a data model* which presents the mapping results for 12 EXPRESS declarations about high-level product definition (see Section "Application to the mapping of data models").

The language reference manual analysis

The mapping process finds the pairs of similar language expressions, and thus, the documents that describe the definitions and usage of the two languages are analyzed. The language reference manuals compared in this research are "*Industrial automation systems and Integration – Product data representation and exchange*

https://doi.org/10.1017/S0890060423000185 Published online by Cambridge University Press

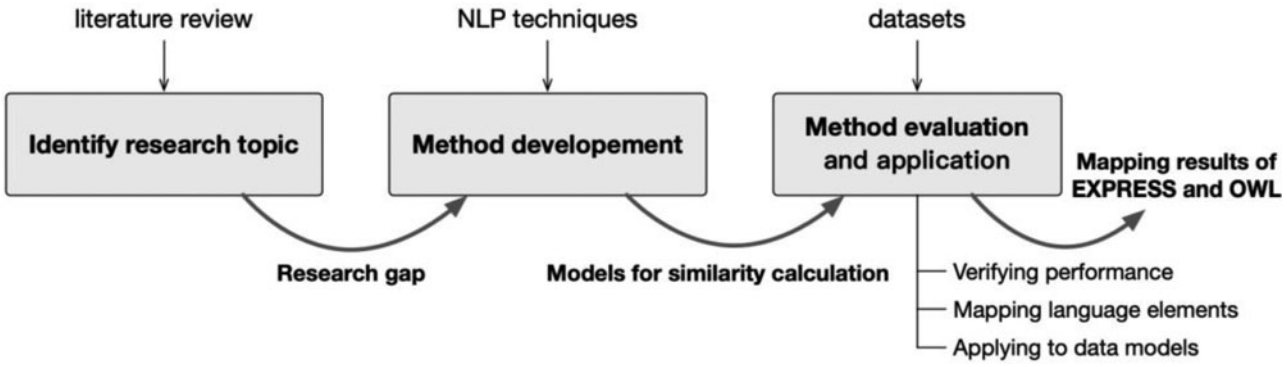


Figure 1. Research overview.

– *Part 11: Description methods: The EXPRESS language reference manual*" (ref. no. ISO 10303-11:2004(E)) and "*OWL 2 Web Ontology Language Structural Specification and Functional – Style Syntax*" (2nd Edition). Both documents define the languages, specifying their elements in terms of representation, meaning, and usage. The two manuals are the input of the proposed method for the semantic similarity analysis.

The EXPRESS manual consists of 16 sections and the OWL 2 manual has 11 sections, excluding the appendix. Section/subsection headings and the descriptive text under these headings are analyzed for each language and subsequently compared. The headings consist of words and short phrases, reflecting not only the essence of the sections' content but also the language elements that allow an unambiguous data definition and specification of constraints to construct the syntax. For example, both headings "Data types" in the EXPRESS document and "Datatype Maps" in the OWL 2 document show that the sections define the data types. Their subheadings "String data type" and "Strings", respectively, describe a particular data type – String. The semantic similarity of the heading directly points out the mapping between the elements.

The descriptive text under each section/subsection describes the language elements in detail. Take *subtype* in EXPRESS and *subclass* in OWL 2 for example. From the two words, it is not obvious that the two can be mapped to each other. The following two paragraphs from the manuals suggest a possible mapping because both indicate a "child–parent" hierarchical relation.

> EXPRESS allows for the specification of entities as subtypes of other entities, where a subtype entity is a specialization of its supertype. This establishes an inheritance (that is, subtype/supertype) relationship between the entities in which the subtype inherits the properties (that is, attributes and constraints) of its supertype. – from section 9.2.3 of EXPRESS manual

> A subclass axiom SubClassOf (CE1 CE2) states that the class expression CE1 is a subclass of the class expression CE2. Roughly speaking, this states that CE1 is more specific than CE2. Subclass axioms are a fundamental type of axioms in OWL 2 and can be used to construct a class hierarchy. – from section 9.1.1 of OWL 2 manual

Both the headings and the text can be analyzed by NLP techniques but need different models, because headings are short whereas text consists of sentences. An examination of the manuals revealed that for the elements with similar functions their headings are also at similar levels of hierarchy, for example subtype and subclass. The heading hierarchy is, therefore, used as an extra dimension for mapping analysis.

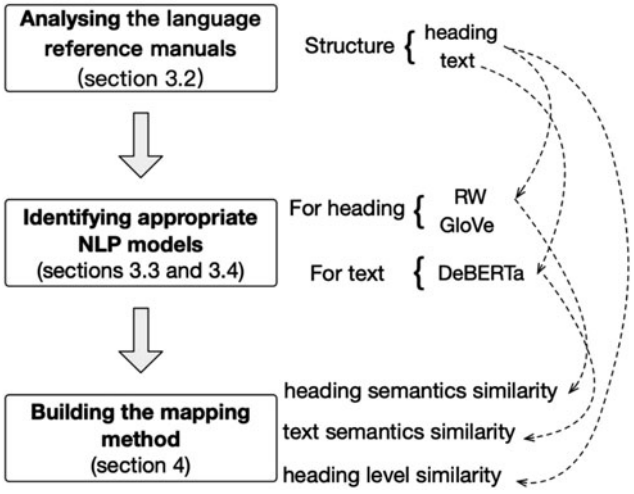


Figure 2. The steps of developing the mapping method.

RW and GloVe for heading similarity analysis

The headings of the manual sections are composed of single or few words. The semantic matching of two headings can be handled by NLP models for word similarity tasks. Random Walks (RW) (Goikoetxea et al., 2015) and Global Vectors for Word Representation (GloVe) (Pennington et al., 2014) are two such models with outstanding performance.

The RW model

The RW model is obtained through two stages, that is establishing the corpus by random walk in WordNet (Miller, 1995) and training on the established corpus using Word2Vec (Mikolov et al., 2013). WordNet is a large lexical database of English designed at Princeton University, which is a network of meaningfully related words and concepts. Nouns, verbs, adjectives, and adverbs are grouped into sets of cognitive synonyms (synsets) and these synsets are further interrelated by means of conceptual-semantic and lexical relations. In this sense, the adjacent words in the established corpus from WordNet are interlinked with semantics and thus the word vectors trained on the corpus can better capture the semantic relationship between words. The random walk

https://doi.org/10.1017/S0890060423000185 Published online by Cambridge University Press

algorithm for establishing the corpus is described in Algorithm 1 based on Goikoetxea et al. (2015).

Algorithm 1. Establishing the corpus by random walk

Input: $G = (V, E)$: undirected graph of WordNet 3.0
$V = \{v_1, v_2, \ldots, v_n\}$: vertex set of synonyms
$E = \{e_1, e_2, \ldots, e_m\}$: edge set of semantic relationships
$D(v,w)$: the probability that a concept v is lexicalized by a word w in dictionary D
Output: C: the corpus set
initialise: $C = \{\}$
corpus_set_length = 0
set: lc: the maximum corpus set length
ls: the maximum sequence length
α^2: the damping coefficient
Repeat
initialise sequence set $S = \{\}$
sequence_length = 0
choose $v \in V$ at random
Repeat
choose w with probability $D(v,w)$
$S = S \cup w$
sequence_length = sequence_length + 1
choose $v' \in$ NeighbourVertex(v) at random with probability α
$v = v'$
Until probability = 1-α **or** sequence_length = ls
$C = C \cup S$
corpus_set_length = corpus_set_length + 1
Until corpus_set_length = lc

Two neural network models from Word2Vec, that is CBOW and Skip-gram, are then trained with several iterations of random walks over WordNet (Mikolov et al., 2013). CBOW predicts the current word based on the context, whereas Skip-gram predicts surrounding words given the current word (Mikolov et al., 2013).

The GloVe model

The GloVe model produces a word vector space with meaningful substructure through training on global word-word co-occurrence counts. It is based on the simple observation that ratios of word-word co-occurrence probabilities have the potential for encoding some form of meaning. The training objective of GloVe is to learn word vectors of high consistency with the established matrix of word-word co-occurrence counts. The model is established in three stages:

Stage 1: *construct the co-occurrence matrix*, denoted $X = [X_{ij}]$, whose entry X_{ij} is the number of times word j occurs in the context of word i. A decreasing weighting function is used to reflect the influence of word distance on the contribution of the words' relationship to one another. In other words, very distant word pairs are expected to contain less relevant information and thus word pairs that are m words apart contribute $1/m$ to the total count.

Stage 2: *establish the symmetry exchange relationship* between word vectors and the co-occurrence matrix. Equation (1) approximately expresses this relationship where w_i and $\tilde{w}_j$ are the word vectors to be learned, and b_i and $\tilde{b}_j$ are the bias for w_i and $\tilde{w}_j$, respectively, to restore the symmetry:

$$w_i^T \tilde{w}_j + b_i + \tilde{b}_j = \log(X_{ij}). \tag{1}$$

Stage 3: *construct and optimize the loss function* and trains the model. The loss function as Eq. (2) is derived from Eq. (1) where V is the size of the vocabulary and $f(X_{ij})$ is the weighting function. AdaGrad (Duchi et al., 2011), a subgradient method, is used to train the model through optimizing the loss function. The sum w_i and $\tilde{w}_j$ are the resulting word vectors:

$$J = \sum_{i,j=1}^{V} f(X_{i,j})(w_i^T w_j + b_i + b_j - \log(X_{i,j}))^2. \tag{2}$$

DeBERTa for text similarity analysis

DeBERTa (He et al., 2021) is a pre-trained neural language model, which enhances the BERT (Devlin et al., 2019) model using two novel techniques: a disentangled attention mechanism and an enhanced mask decoder. Two main stages can be summarized for establishing the model.

Stage 1: *improve the architecture of BERT*. The disentangled attention mechanism makes use of relative positions as well as the contents of a word pair. It represents each word in two vectors encoding its content and position, and computes the attention weights among words using disentangled matrices. The disentangled self-attention with relative position bias is calculated using Eq. (3). $\tilde{A}_{i,j}$ is the attention score from token i to token j. Q_i^c and K_j^c are the i-th and j-th rows of Q^c and K^c which are two projected content vectors. $Q^r_{\delta(i,j)}$ and $K^r_{\delta(j,i)}$ are the $\delta(i,j)$-th and $\delta(j,i)$-th rows of Q^r and K^r which are two projected relative position vectors, regarding the relative distances $\delta(i,j)$ and $\delta(j,i)$.

$$\begin{aligned} \tilde{A}_{i,j} &= Q_i^c K_j^{cT} + Q_i^c {K^r_{\delta(i,j)}}^T + Q^r_{\delta(j,i)} K_j^{cT}, \\ H_o &= \mathrm{softmax}\left(\frac{\tilde{A}}{\sqrt{3d}}\right) V_c. \end{aligned} \tag{3}$$

As the decoding component of DeBERTa, the enhanced mask decoder then captures the absolute positions of words as complementary information when decoding the masked words. In this way, both relative and absolute positions as well as contents are used in the training stage.

Stage 2: *train the model*. A virtual adversarial training algorithm, namely Scale-invariant-Fine-Tuning, is used for the training, which applies the perturbations to the normalized word embeddings to improve the stability.

Multiple similarity-based method for mapping

Based on the NLP models, this paper proposes a framework for language element mapping between EXPRESS and OWL 2, which consists of five parts as shown in Figure 3.

https://doi.org/10.1017/S0890060423000185 Published online by Cambridge University Press

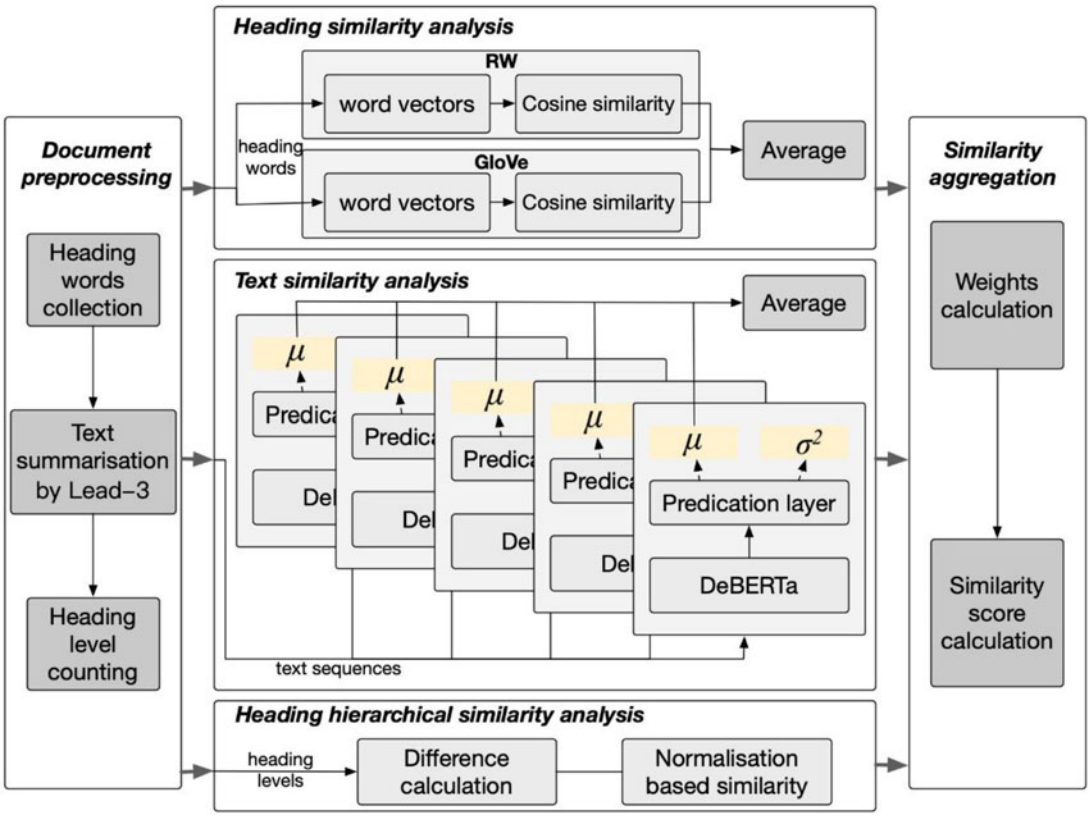


Figure 3. The proposed framework for mapping EXPRESS and OWL.

The details of the following concepts are presented in this section:

- *Document preprocessing* shortens the long sentences in the two language manuals and prepares the inputs for the NLP models.
- *Heading similarity analysis* calculates the semantic similarities of the headings in both manuals using RW and GloVe.
- *Text similarity analysis* calculates the semantic similarities of the text under each section/subsection where a DeBERTa ensemble is suggested.
- *Heading hierarchical similarity analysis* calculates the similarities of the heading levels.
- *Similarity aggregation* generates the overall similarity scores of the language elements, combined with the weights of different types of similarities.

Document preprocessing

In both manuals, language elements are mainly described by text. There are also figures, tables, and codes which are mostly examples to help understanding. These have been removed from the documents for this research.

Each section or subsection targets a particular language element type or language element. The length of the text for each section/subsection varies by up to more than 500 words (seen in Fig. 4a). This increases the difficulty of semantic similarity calculation by NLP models, as a very long text could lead to inaccurate semantic extraction. To make the length of each section/subsection text in the two manuals relatively short and balanced, this research adopts Lead-3 algorithm to extract the representative text without losing semantic information. Lead-3 is a baseline method for text summarization and relies on the observation of the semantic distribution of the text that the first three sentences of a text can generally summarize the main semantic information of the whole text. Lead-3 was chosen because it is simple to implement and capable of obtaining effective results (Nallapati et al., 2017). After processing the manuals, most of the texts are within 100 words, as shown in Figure 4b.

The preprocessed documents from the two manuals are used for analyzing semantic similarities. The words of the headings are collected as input to the heading similarity analysis model; the summarized paragraphs are the input to the text similarity analysis model; and heading levels are the input to the heading hierarchical similarity analysis model.

Analysis of heading similarity based on RW and GloVe

Two aspects of a language element impact the translation of manual headings:

- *Keywords*: RW focuses on analyzing words through their embedded context. For example, both *subtype* and *subclass* indicate a hierarchical relation.
- *Synonyms*: GloVe performs well on these word analogies, such as *subtype* and *subclass* or *entity* and *class*.

To include both aspects, this paper combines RW and GloVE to analyze the semantic similarities of headings. The headings from both documents are the input to the RW and GloVe models. Two word-embedding matrices are then generated, denoted W^{RW} and W^{GloVe}, respectively. Given two words w_i and w_j, their similarity is calculated as follows.

https://doi.org/10.1017/S0890060423000185 Published online by Cambridge University Press

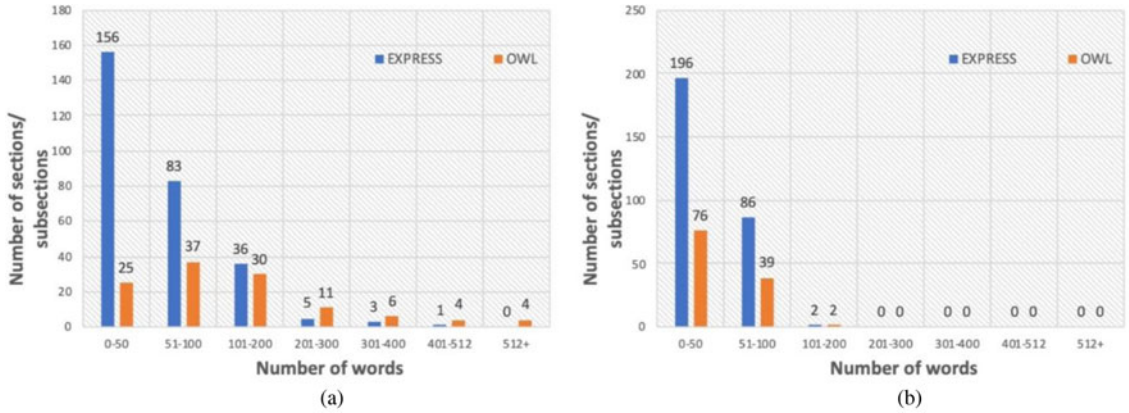


Figure 4. Words count of (a) before and (b) after text summarization.

Step 1: locate the word in the word-embedding matrix. As Eq. (4) indicates, the word vectors of the two input words are found from the matrices W^{RW} and W^{GloVe}.

$$\begin{aligned} vec_i^{\mathrm{RW}} &= W^{\mathrm{RW}}(w_i),\ vec_j^{\mathrm{RW}} = W^{\mathrm{RW}}(w_j), \\ vec_i^{\mathrm{GloVe}} &= W^{\mathrm{GloVe}}(w_i),\ vec_j^{\mathrm{GloVe}} = W^{\mathrm{GloVe}}(w_j), \end{aligned} \tag{4}$$

where vec_i^{RW} and vec_j^{RW} are the word vectors of w_i and w_j in the matrix W^{RW}, and vec_i^{GloVe} and vec_j^{GloVe} are the word vectors of w_i and w_j in the matrix W^{GloVe}.

Step 2: compute the similarity in the individual model. For element i in EXPRESS and element j in OWL 2, two semantic similarity values of comparing their headings are computed by Eq. (5). sim_{RW} and sim_{GloVe} denote the values from the RW model and GloVe model, respectively.

$$\begin{aligned} sim_{\mathrm{RW}}^{ij} &= \mathrm{cosin}(vec_i^{\mathrm{RW}}, vec_j^{\mathrm{RW}}), \\ sim_{\mathrm{GloVe}}^{ij} &= \mathrm{cosin}(vec_i^{\mathrm{GloVe}}, vec_j^{\mathrm{GloVe}}), \end{aligned} \tag{5}$$

where $\mathrm{cosin}(v_i,v_j)$ is the function calculating the cosine similarity between the two vectors v_i and v_j.

Step 3: compute the final heading similarity by averaging the two semantic similarity values from step 2.

$$HSim_{ij} = \frac{1}{2}(sim_{\mathrm{RW}}^{ij} + sim_{\mathrm{GloVe}}^{ij}). \tag{6}$$

When there are multiple words in the headings, the word vectors in step 1 are the averaged vectors of all the words.

Text similarity analysis based on DeBERTa

The text under the headings explains the language elements in terms of their definitions, usage, and characteristics. After preprocessing the documents with the Lead-3 algorithm, the length of the text in each section/subsection is within 100 words. Similarity analysis on sentence level is carried out with a DeBERTa model. DeBERTa model performs well in classification and regression tasks but tends to produce overconfidence predication when handling samples out of the distribution. To overcome this problem and produce accurate text similarity results, the DeBERTa ensemble proposed by Jian and Liu (2021) is applied, which adds a predication layer to the original DeBERTa. This additional layer is setup with two neurons, the outputs of which are the mean of the predicated values and the variance. To enhance the discrimination of learned results, the negative log-likelihood loss function shown as Eq. (7) is employed, where σ^2 is the variance, μ is the mean of the predicated values, and y is the actual labeled value.

$$\mathrm{NLL} = \frac{\log \sigma^2}{2} + \frac{(y-\mu)^2}{2\sigma^2}. \tag{7}$$

The models with added layer are further gathered to build an ensemble to improve the reliability, which are trained with the same data but set with different initial weights. As illustrated in the framework (referred to Fig. 3), the ensemble consists of five models of the same structure to balance the consumed resource and the expected performance. In this case, the text similarity is the aggregated result of the five models, as shown in Eq. (8). $TSim_t^{ij}$ is the similarity score of the two sentences i and j from the t-th model.

$$TSim_{ij} = \frac{1}{5}\sum_{t=1}^{5} TSim_t^{ij}. \tag{8}$$

Heading hierarchical similarity analysis

The heading hierarchy reflects the granularity of language elements. According to the examination of the documents, if the functions of a pair of elements from the two languages are similar, their headings are usually at similar levels. In this case, this paper suggests taking the heading hierarchical similarity into consideration for the mapping between EXPRESS and OWL 2. The analysis follows three steps.

Step 1: obtain the heading hierarchy sets. Through the statistics of the official documents, two sets (denoted $headinglevel_{EXPRESS}$ for EXPRESS and $headinglevel_{OWL}$ for OWL 2 separately) are established, which consists of the heading levels of all the sections/subsections.

Step 2: compute the heading level difference. Given element i in EXPRESS and element j in OWL 2, the difference is computed by Eq. (9), where $headinglevel_{EXPRESS}[i]$ and $headinglevel_{OWL}[j]$ are the heading levels of the sections that describe element i

https://doi.org/10.1017/S0890060423000185 Published online by Cambridge University Press

and j, respectively.

$$diff_{ij} = |headinglevel_{EXPRESS}[i] - headinglevel_{OWL}[j]|. \quad (9)$$

Step 3: calculate the heading hierarchical similarity. Linear normalization is used to limit the value to [0,1]. The similarity is calculated by Eq. (10) where d^+ is the maximum level difference and d^- is the minimum level difference.

$$LSim_{ij} = \frac{d^+ - diff_{ij}}{d^+ - d^-}. \quad (10)$$

Similarity aggregation based on fuzzy AHP-SAW

The three types of similarity values contribute to the overall similarity between the language elements, but their contributions are not of equal importance. Thus, their weights are determined by Analytic Hierarchy Process (AHP) (Saaty, 1980) first and then aggregated by a simple additive weighting (SAW) method. Both AHP and SAW are multiple criteria decision-making (MCDM) methods. AHP calculates the weights by pairwise comparison which makes it outperform most other methods. SAW produces an aggregated result by combining the weights of the criteria and the values under the criteria. It is a simple and practical method, which offers a transparent and understandable calculation process for the ranking results (Kaliszewski and Podkopaev, 2016; Wang, 2019). The following steps calculate the weights of the three types of similarities and the overall similarity values of the language elements. Steps 1 and 2 come from AHP, while step 3 draws on SAW.

Step 1: establish the similarity pairwise comparison matrix $F = [c_{st}]_{n \times n}$. n is the number of types of similarities ($n = 3$ in our case), and s_{st} is a value from 1 to 9 representing the relative importance of type s over type t.

Step 2: obtain the weights of the similarity types. The weight of similarity type i is computed by Eq. (11) where geometric mean is applied.

$$w_s = \frac{\sqrt[n]{\prod_{t=1}^{n} c_{st}}}{\sum_{s=1}^{n} \sqrt[n]{\prod_{t=1}^{n} c_{st}}}. \quad (11)$$

Step 3: calculate the overall similarity values. Given element i in EXPRESS and element j in OWL 2, their similarity value is calculated by the SAW method shown in Eq. (12). The result is further translated by Eq. (13) with the three types of similarity in this research, where w_{head}, w_{text}, and w_{level} are the weights of heading, text, and hierarchy, respectively.

$$OSim_{ij} = \sum_{s=1}^{n} w_s \times sim_s^{ij}, \quad (12)$$

$$OSim_{ij} = w_{\text{head}} HSim_{ij} + w_{\text{text}} TSim_{ij} + w_{\text{level}} LSim_{ij}. \quad (13)$$

Evaluation and mapping results

To test the feasibility of elements mapping, the two semantic analysis models are evaluated first. This section then presents the mapping results from the proposed method and compares them with those from the existing research. The approach is also applied to the data model of a linear bearing.

The combined RW and GloVe models

As the headings in the language manuals consist of nouns, three datasets targeted at the similarities of nouns were used to verify the performance of the model:

- MC consists of 30 pairs of nouns with their semantic similarities that were judged by 38 native English speakers (Miller and Charles, 1991).
- RG contains 65 word pairs collected, each associated with 51 human subject for the judgement on their similarity (Rubenstein and Goodenough, 1965).
- Agirre includes 203 word pairs using search engine methods to calculate the similarities (Agirre et al., 2009).

Tables 2 and 3 show the Pearson correlation coefficient and the Spearman correlation coefficient obtained on the three datasets. The two indicators are often used to measure the capability of NLP models for calculating the similarity. The last column lists the mean value. With a Pearson correlation value of 0.854 and a Spearman correlation value of 0.880, it can be seen that the combined model performs better than individual RW and GloVe on the three datasets of nouns.

The DeBERTa ensemble

The efficiency of the DeBERTa ensemble for text similarity has been proved by comparing with BERT, DeBERTa, and M-MaxLSTM-CNN (Tien et al., 2019) models on STSBenchmark dataset. This dataset contains 8628 labeled sentence pairs which have been extracted from image captions, news headlines, and user forums (Cer et al., 2017). Each sentence pair is labeled with a score from 0 to 5, denoting how similar the two sentences are in terms of semantic meaning.

Table 4 shows the Spearman correlation coefficient and Pearson correlation coefficient scores. It can be seen that the

Table 2. Pearson correlation results

Model \ Dataset	MC	RG	Agirre	Mean value
RW	0.835	0.797	0.773	0.802
GloVe	0.845	0.770	0.797	0.804
Combined model	0.890	0.834	0.839	0.854

Table 3. Spearman correlation results

Model \ Dataset	MC	RG	Agirre	Mean value
RW	0.909	0.823	0.784	0.839
GloVe	0.862	0.769	0.795	0.809
Combined model	0.928	0.860	0.852	0.880

Table 4. The scores of the DeBERTa ensemble

Model \ Correlation	Pearson	Spearman
M-MaxLSTM-CNN	0.8245	–
BERT	0.876	0.865
DeBERTa	0.928	0.925
DeBERTa ensemble	0.9293	0.9277

https://doi.org/10.1017/S0890060423000185 Published online by Cambridge University Press

Table 5. Mapping results for simple elements

	OWL 2 element					
EXPRESS element	Krima et al. (2009) and Barbau et al. (2012)	Mapping candidate section	*HSim* (0.129)	*TSim* (0.818)	*LSim* (0.053)	*OSim*
Schema	Ontology	**3.1 Ontology IRI and Version IRI**	0.441	0.337	1.000	0.385
		3.3 Versioning of OWL 2 Ontologies	0.455	0.246	1.000	0.313
		5.8.2 Declaration Consistency	0.483	0.244	0.666	0.299
		5 Entities, Literals, and Anonymous Individuals	0.503	0.245	0.666	0.298
Entity	Class	9.3.2 Equivalent Data Properties	0.533	0.458	1	0.496
		8.4.2 Universal Quantification	0.411	0.470	1	0.490
		8 Class Expressions	0.58	0.445	0.333	0.457
		5.4 Data Properties	0.474	0.440	0.666	0.457
Instance	Individual	4 Datatype Maps	0.200	0.569	0.333	0.508
		5.6.1 Named Individuals	0.491	0.399	1.000	0.442
		5.6 Individuals	0.147	0.465	0.666	0.435
		9.2.9 Reflexive Object Properties	0.184	0.438	1.000	0.434
Attribute	DataProperty ObjectProperty	5.8.2 Declaration Consistency	0.530	0.296	1.000	0.363
		5.4 Data Properties	0.621	0.300	0.666	0.360
		9.3.6 Functional Data Properties	0.633	0.275	1.000	0.359
		9.6.6 Positive Data Property Assertions	0.674	0.268	1.000	0.359
Number data type	Decimal	**4.1 Real Numbers, Decimal Numbers, and Integers**	0.610	0.423	0.666	0.459
		7 Data Ranges	0.603	0.413	0.333	0.433
		9.3.4 Data Property Domain	0.687	0.335	1.000	0.415
		4.5 Binary Data	0.583	0.369	0.666	0.412
Real data type	Double	**4.1 Real Numbers, Decimal Numbers, and Integers**	0.637	0.347	0.666	0.401
		4.2 Floating-Point Numbers	0.336	0.330	0.666	0.348
		9.3.4 Data Property Domain	0.716	0.225	1.000	0.329
		5.4 Data Properties	0.626	0.256	0.666	0.325
Integer data type	Integer	**4.1 Real Numbers, Decimal Numbers, and Integers**	0.651	0.377	0.666	0.427
		4 Datatype Maps	0.462	0.399	0.333	0.403
		7 Data Ranges	0.380	0.303	0.333	0.348
		4.2 Floating-Point Numbers	0.645	0.314	0.666	0.341
Logical data type	–	5.4 Data Properties	0.640	0.335	0.666	0.392
		5 Entities, Literals, and Anonymous Individuals	0.473	0.365	0.333	0.377
		4 Datatype Maps	0.469	0.353	0.333	0.367
		7.4 Enumeration of Literals	0.478	0.321	0.666	0.359
		4.4 Boolean Values	0.463	0.288	0.666	0.330
Boolean data type	Boolean	5.4 Data Properties	0.697	0.351	0.666	0.412
		4.4 Boolean Values	0.622	0.347	0.666	0.399
		9.6.7 Negative Data Property Assertions	0.686	0.289	1.000	0.377
		9.3.4 Data Property Domain	0.725	0.280	1.000	0.375
String data type	String	4 Datatype Maps	0.420	0.489	0.333	0.471
		4.3 Strings	0.485	0.359	0.666	0.391
		3.7 Functional-Style Syntax	0.244	0.386	0.666	0.382
		7 Data Ranges	0.622	0.322	0.333	0.361

(*Continued*)

https://doi.org/10.1017/S0890060423000185 Published online by Cambridge University Press

Table 5. (*Continued.*)

EXPRESS element	OWL 2 element					
	Krima et al. (2009) and Barbau et al. (2012)	Mapping candidate section	*HSim* (0.129)	*TSim* (0.818)	*LSim* (0.053)	*OSim*
Binary data type	–	***4.5 Binary Data***	0.651	0.366	0.666	0.419
		4 Datatype Maps	0.376	0.403	0.333	0.395
		9.3.4 Data Property Domain	0.669	0.228	1.000	0.325
		7 Data Ranges	0.588	0.283	0.333	0.324

performance of the proposed model is close to DeBERTa and better than the rest two models.

The mapping results of language elements and discussion

Table 5 shows the mapping results of EXPRESS to OWL 2 by applying the proposed multiple similarity-based method. The first column lists the language elements of EXPRESS. The second column presents the OWL 2 mappings from Krima et al. (2009) and Barbau et al. (2012). Their work provides the most mappings among the reviewed research (as seen in Table 1) and thus was chosen by this research to validate the method. The third column presents the four OWL sections of the top similarity scores as the mapping candidates. The remaining columns show the similarity scores of the headings (*HSim*), text (*TSim*), heading hierarchies (*LSim*), and the overall (*OSim*) with their weights below. Three digits are shown in Table 5, but more were used for calculation.

For some EXPRESS elements that express complex information, for example *bag* and *set*, no one-to-one mapping in OWL exists and a combination of several OWL elements is required. The existence of multiple candidates makes the combination feasible. The results are shown in Table 6.

It can be seen from the tables that most mapped OWL 2 elements from Krima's work are covered by the top four identified candidates of this research (as highlighted in **bold**) except *array*, *select,* and *AND*. Their mapping candidate sections have a lower rank. The method also finds the potential mapping sections for elements that are not included in Krima, as highlighted in *italics* in the table. For example, the top candidate for 4.5 Binary Data was the language element *xsd:hexBinary* locates. The mappings of INVERSE and CONSTANT are another two examples. For the logical data type, despite no feasible mapping in the first four candidate sections, the follow-up candidate "4.4 Boolean Values" provides a solution that both have TRUE and FALSE values as their domain. Because OWL 2 follows the Open World Assumption that every undefined item is considered as unknown, which corresponds to the third domain value of logical data type in EXPRESS, that is UNKNOWN.

Application to the mapping of data models

The language elements are used to establish data models. ISO 10303 provides a representation of product information with EXPRESS declarations to enable data exchange. For example, ISO 10303-41: Integrated generic resources: Fundamentals of product description and support (Part41 for short hereafter) specifies the generic product description resources, generic management resources, and support resources. The following entity is extracted from Part41. It is a collector of definitions of a product.

```
ENTITY product_definition_formation;
    id : identifier;
    description : OPTIONAL text;
    of_product : product;
    UNIQUE
    UR1 : id, of_product;
END_ENTITY;
```

OWL has no standards for their definitions of product data. This allows users to define their own models but also leads to the difficulty of mapping an EXPRESS data model to an OWL data model. A solution is to extract the parts with similar semantics to EXPRESS declarations from OWL files. This research applies the proposed model to comparing the similarities in an example of a linear bearing in stamping outer ring (see the right upper part of Fig. 5). The STEP file was downloaded from an online CAD part database – LinkAble PARTcommunity (https://linkable.partcommunity.com/3d-cad-models/sso/khm-%E5%86%B2%E5%8E%8B%E5%A4%96%E5%9C%88%E5%9E%8B%E7%9B%B4%E7%BA%BF%E8%BD%B4%E6%89%BF-%E4%B8%8A%E9%9A%86samlo?info=samlo%2Flinear_bushings_ball_retainers%2Fkhm.prj&cwid=7526), which describes the category and geometry of a linear bearing. Its description conformed to the predefined data models provided by ISO 10303 series standards. The file was encoded in the format of ISO 10303-21: Implementation methods: Clear text encoding of the exchange structure (Part 21 for short), which is described according to ISO 10303-203: Application protocol: Configuration controlled design. ISO 10303-203 specifies an application protocol for exchanging configuration-controlled 3D product design data of mechanical parts and assemblies. Other parts of ISO 10303 such as Part41 constitute provisions of this standard. Figure 5 shows a segment of the file. This research takes use of the EXPRESS declarations in the example file, for example entities such as *product_definition_formation* as inputs of EXPRESS side.

The ontologies of the part were defined in OWL. Since there is no fixed format to describe a product, different descriptions could exist. For example, property of *id: identifier* can either be defined in OWL as an object data property linking to a class named *identifier* (as indicated in Fig. 6a) or as a data property in string type (as indicated in Fig. 6b). This research takes the two types of definition as the inputs from the OWL side.

The proposed model was run to match the segments in OWL to the EXPRESS declarations. In the first round, the OWL file

https://doi.org/10.1017/S0890060423000185 Published online by Cambridge University Press

Table 6. Mapping results for complex elements

	OWL 2 element	
EXPRESS element	Mapping element	Candidate section
Subtype of	*SubClassOf*	3.1 Ontology IRI and Version IRI
		9.3.1 Data Subproperties
		8.1.2 Union of Class Expressions
		9.1.1 Subclass Axioms
Array	ObjectProperty ObjectMinCardinality ObjectMaxCardinality	7 Data Ranges
		8.1.4 Enumeration of Individuals
		9.3.5 Data Property Range
		4.5 Binary Data
		8.5.2 Maximum Cardinality 8.5.1 Minimum Cardinality
List	ObjectProperty ObjectMinCardinality ObjectMaxCardinality	7 Data Ranges
		8.5.2 Maximum Cardinality
		8.5.1 Minimum Cardinality
		8.3.1 Minimum Cardinality
Bag	ObjectProperty ObjectMinCardinality ObjectMaxCardinality	**8.3.2 Maximum Cardinality**
		8.3.1 Minimum Cardinality
		7 Data Ranges
		8.5.2 Maximum Cardinality
Set	ObjectProperty ObjectMinCardinality ObjectMaxCardinality	**8.3.1 Minimum Cardinality**
		8.3.2 Maximum Cardinality
		7 Data Ranges
		5 Entities, Literals, and Anonymous Individuals
Enumeration	ObjectOneOf EquivalentClasses	7.4 Enumeration of Literals
		8.1.4 Enumeration of Individuals
		10.2.3 Annotation Property Domain
		9.3.4 Data Property Domain
Select	ObjectUnionOf EquivalentClasses	9.3.4 Data Property Domain
		5.4 Data Properties
		5.6.1 Named Individuals
		4.5 Binary Data
		9.1.2 Equivalent Classes 8.1.2 Union of Class Expressions
INVERSE	*InverseObjectProperties*	6.1.1 Inverse Object Properties
		9.2.4 Inverse Object Properties
		9.2.8 Inverse-Functional Object Properties
		5.8 Entity Declarations and Typing
UNIQUE	*HasKey* *ObjectProperty*	***9.5 Keys***
		3.1 Ontology IRI and Version IRI
		5.8 Entity Declarations and Typing
		5.4 Data Properties
ONEOF	ObjectUnionOf DisjointClasses EquivalentClasses	**8.1.2 Union of Class Expressions**
		3.1 Ontology IRI and Version IRI

(*Continued*)

https://doi.org/10.1017/S0890060423000185 Published online by Cambridge University Press

Table 6. (*Continued.*)

EXPRESS element	OWL 2 element	
	Mapping element	Candidate section
		5.8 Entity Declarations and Typing
		8.1.1 Intersection of Class Expressions
AND	ObjectIntersectionOf EquivalentClasses	3.1 Ontology IRI and Version IRI
		9.1 Class Expression Axioms
		8.1.4 Enumeration of Individuals
		9.2.4 Inverse Object Properties
		9.1.2 Equivalent Classes 8.1.1 Intersection of Class Expressions
ANDOR	ObjectUnionOf EquivalentClasses	**8.1.2 Union of Class Expressions**
		8.1.1 Intersection of Class Expressions
		5.8 Entity Declarations and Typing
		3.1 Ontology IRI and Version IRI
CONSTANT	*NamedIndividual*	9.3.4 Data Property Domain
		9.3.6 Functional Data Properties
		5.6 Individuals
		5.6.1 Named Individuals

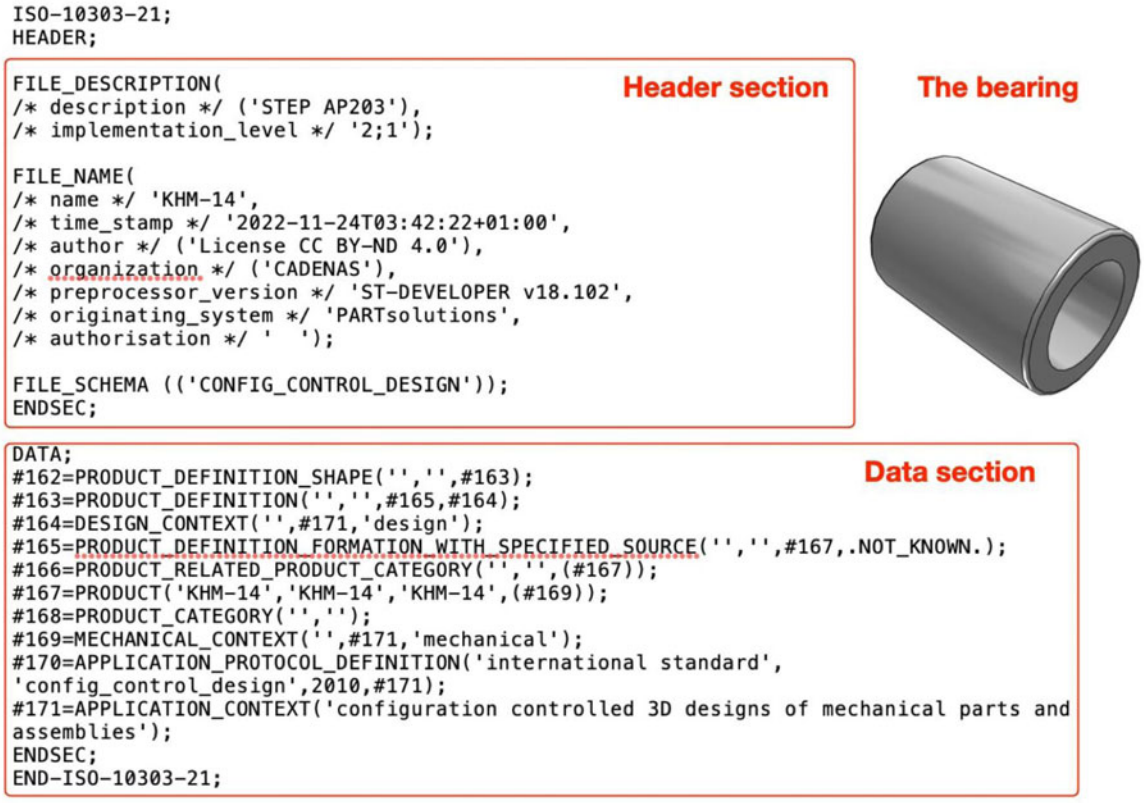


Figure 5. A segment of the STEP file of the example product bearing.

https://doi.org/10.1017/S0890060423000185 Published online by Cambridge University Press

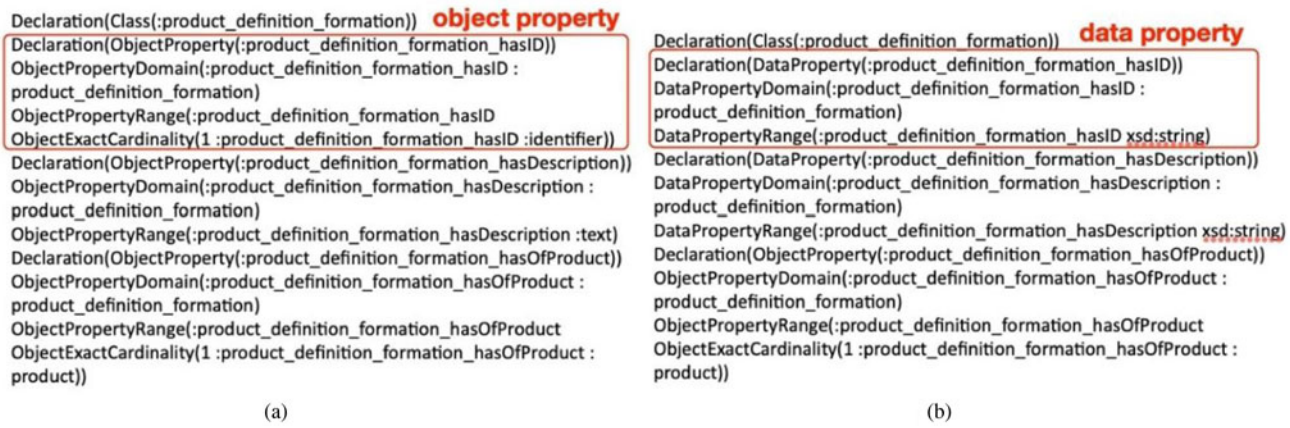


Figure 6. Examples OWL expressions for Entity *product_definition_formation*: (a) using object property and (b) using data property.

defined in the way of Figure 6a was considered. Table 7 shows the OWL segments of top 4 similarities for entity *product_definition_formation* and entity *product_definition_formation_with_specified_source*. Appendix A presents the mapping results for 12 EXPRESS declarations about high-level product definition. In the second round, the OWL file defined as Figure 6b was considered (the result is listed in Appendix B). The results indicate that the best matched OWL segments have the highest similarities.

Table 7. Mapping result of OWL segments to two EXPRESS entities in the example

EXPRESS declaration	OWL segments	Similarity
ENTITY product_definition_formation; id : identifier; description : OPTIONAL text; of_product : product; UNIQUE UR1 : id,of_product; END_ENTITY;	Declaration(Class(:product_definition_formation)) Declaration(ObjectProperty(:product_definition_formation_hasID)) ObjectPropertyDomain(:product_definition_formation_hasID : product_definition_formation) ObjectPropertyRange(:product_definition_formation_hasID ObjectExactCardinality(1 : product_definition_formation_hasID :identifier)) Declaration(ObjectProperty(:product_definition_formation_hasDescription)) ObjectPropertyDomain(:product_definition_formation_hasDescription : product_definition_formation) ObjectPropertyRange(:product_definition_formation_hasDescription :text) Declaration(ObjectProperty(:product_definition_formation_hasOfProduct)) ObjectPropertyDomain(:product_definition_formation_hasOfProduct : product_definition_formation) ObjectPropertyRange(:product_definition_formation_hasOfProduct ObjectExactCardinality(1 :product_definition_formation_hasOfProduct :product))	0.566
	Declaration(Class(:product_definition_formation_with_specified_source)) SubClassOf(:product_definition_formation_with_specified_source : product_definition_formation) Declaration(ObjectProperty(: product_definition_formation_with_specified_source_hasMakeOrBuy)) ObjectPropertyDomain(: product_definition_formation_with_specified_source_hasMakeOrBuy : product_definition_formation_with_specified_source) ObjectPropertyRange(: product_definition_formation_with_specified_source_hasMakeOrBuy :source)	0.434
	Declaration(Class(:product)) Declaration(ObjectProperty(:product_hasID)) ObjectPropertyDomain(:product_hasID :product) ObjectPropertyRange(:product_hasID ObjectExactCardinality(1 :product_hasID : identifier)) Declaration(ObjectProperty(:product_hasName)) ObjectPropertyDomain(:product_hasName :product) ObjectPropertyRange(:product_hasName :label) Declaration(ObjectProperty(:product_hasDescription)) ObjectPropertyDomain(:product_hasDescription :product) ObjectPropertyRange(:product_hasDescription :text) Declaration(ObjectProperty(:product_hasFrameOfReference)) ObjectPropertyDomain(:product_hasFrameOfReference :product) ObjectPropertyRange(:product_hasFrameOfReference ObjectMinCardinality(1 : product_hasFrameOfReference :product_context))	0.432

(*Continued*)

https://doi.org/10.1017/S0890060423000185 Published online by Cambridge University Press

Table 7. (*Continued.*)

EXPRESS declaration	OWL segments	Similarity
	Declaration(Class(:application_context_element)) Declaration(ObjectProperty(:application_context_element_hasName)) ObjectPropertyDomain(:application_context_element_hasName :application_context_element) ObjectPropertyRange(:application_context_element_hasName :label) Declaration(ObjectProperty(:application_context_element_hasFrameOfReference)) ObjectPropertyDomain(:application_context_element_hasFrameOfReference : application_context_element) ObjectPropertyRange(:application_context_element_hasFrameOfReference : application_context)	0.391
ENTITY product_definition_formation_with_specified_source; SUBTYPE OF (product_definition_formation); make_or_buy : source; END_ENTITY;	Declaration(Class(:product_definition_formation_with_specified_source)) SubClassOf(:product_definition_formation_with_specified_source : product_definition_formation) Declaration(ObjectProperty(:product_definition_formation_with_specified_source_hasMakeOrBuy)) ObjectPropertyDomain(:product_definition_formation_with_specified_source_hasMakeOrBuy : product_definition_formation_with_specified_source) ObjectPropertyRange(:product_definition_formation_with_specified_source_hasMakeOrBuy :source)	0.579
	Declaration(Class(:product_definition_formation)) Declaration(ObjectProperty(:product_definition_formation_hasID)) ObjectPropertyDomain(:product_definition_formation_hasID : product_definition_formation) ObjectPropertyRange(:product_definition_formation_hasID ObjectExactCardinality(1 : product_definition_formation_hasID :identifier)) Declaration(ObjectProperty(:product_definition_formation_hasDescription)) ObjectPropertyDomain(:product_definition_formation_hasDescription : product_definition_formation) ObjectPropertyRange(:product_definition_formation_hasDescription :text) Declaration(ObjectProperty(:product_definition_formation_hasOfProduct)) ObjectPropertyDomain(:product_definition_formation_hasOfProduct : product_definition_formation) ObjectPropertyRange(:product_definition_formation_hasOfProduct ObjectExactCardinality(1 :product_definition_formation_hasOfProduct :product))	0.454
	Declaration(Class(:application_context_element)) Declaration(ObjectProperty(:application_context_element_hasName)) ObjectPropertyDomain(:application_context_element_hasName :application_context_element) ObjectPropertyRange(:application_context_element_hasName :label) Declaration(ObjectProperty(:application_context_element_hasFrameOfReference)) ObjectPropertyDomain(:application_context_element_hasFrameOfReference : application_context_element) ObjectPropertyRange(:application_context_element_hasFrameOfReference : application_context)	0.413
	Declaration(Class(:product_definition_context)) SubClassOf(:product_definition_context :application_context_element) Declaration(ObjectProperty(:product_definition_context_hasLifeCycleStage)) ObjectPropertyDomain(:product_definition_context_hasLifeCycleStage : product_definition_context) ObjectPropertyRange(:product_definition_context_hasLifeCycleStage :label)	0.409

In most cases, the parent-class or sub-class (e.g., *product_definition_formation* to *product_definition_formation_with_specified_source* or vice versa) or sibling-class (*product_definition_context* and *product_concept_context*) are identified as the secondary mapping. This is followed by those classes with similar properties, for example *product_definition_formation* and *product* both have property *id* and the former is also linked to the later via property *of_product*.

Conclusion

Converting EXPRESS-based models to OWL described models helps transfer product data with semantic information. The conversion is usually based on manually mapping the language elements in EXPRESS and OWL 2. To our knowledge, this paper is the first attempt to locate potential language elements automatically by applying NLP techniques. The proposed method first analyzes the semantic similarities of section headings in the two language reference manuals using a combination of RW and GloVe models, and then calculates the semantic similarities of the text in each section/subsection by the DeBERTa ensemble. The heading hierarchical similarity is also considered. The three types of similarities are aggregated for an overall score by the AHP-SAW method that combines the similarity values with their weights. The results not only cover the currently mapped language elements, but also identify the mappings of elements that have not previously been included.

The proposed method narrows down the look-up scope by providing four potential candidates of OWL language elements

https://doi.org/10.1017/S0890060423000185 Published online by Cambridge University Press

for EXPRESS language elements, which reduces the effort in finding the mapping manually and the potential mistakes from different human understandings. The ideal solution is to provide a one-to-one mapping result however, this is a hard task due to the discrepancies between the inheritance mechanism of EXPRESS and OWL. The discrepancies can also be found when translating the properties of an EXPRESS entity on the data model level. OWL could handle the properties either as object properties by adding new classes or as data properties by linking to appropriate built-in datatypes. More accurate and appropriate mappings might be achieved by training the models with more datasets and refining the NLP models to diminish the discrepancies. The underlying principle of the proposed models is to train them first by telling what items (words, phrases, sentences, and paragraphs) are similar and to what extent they are similar. For example, Dataset SemEval-2014 Task 3 can be used for semantic similarity across different lexical levels, including paragraph to sentence, sentence to phrase, phrase to word, and word to sense (Jurgens et al., 2014). The more example data sets are learned, the better results the models can produce, especially when the models are fed with various representation formats of similar expressions.

Though targeting at the mapping between EXPRESS and OWL, the two NLP models for semantic similarity analysis were trained with general English datasets. It indicates a potential generalized application so that the models might also be used for converting other descriptive modeling languages, for example converting XML to OWL. This could also consolidate product information expressed by different modeling languages. This paper focuses on finding the corresponding language elements, which is the fundamental step to convert a STEP model to an Ontology model. The full conversion requires replacing the statements defined by the language elements in particular format. The proposed model was also applied to a data model level. It shows the potential of identifying OWL expressions for EXPRESS declarations. This could help experts to extract the mapping patterns and then to design an inference engine for the translation. It could also enable the use of machine learning techniques where the program learns the conversion rules with the identified EXPRESS declarations and OWL expression, and further translates the data models or even the population of data models such as a physical STEP file. In this research, only the EXPRESS declarations in the example STEP file are mapped. More declarations will be extracted and tested in the future work.

Another limitation is that this research has not been applied to the population of data models. Theoretically, the model would work in the same way on this level. Because the free-form text strings consist of words and sentences, the NLP models are capable of analyzing the semantics. However, when translating the EXPRESS based models to OWL models of Layer 3, these free form text strings should be kept as they are. For example, the following is from Part 21 file of the illustrative example in the paper.

#171 = APPLICATION_CONTEXT("configuration controlled 3D designs of mechanical parts and assemblies")

The EXPRESS declaration of APPLICATION_CONTEXT is:

```
ENTITY application_context;
  application : text;
  INVERSE
  context_elements : SET [1:?] OF application_context_element
                                  FOR frame_of_reference;
END_ENTITY;
```

The OWL expression is:

```
Declaration(Class(:application_context))
Declaration(DataProperty(:application_context_hasApplication))
DataPropertyDomain(:application_context_hasApplication :
  application_context)
DataPropertyRange(:application_context_hasApplication xsd:string)
Declaration(ObjectProperty(:application_context_
  hasContextElement))
ObjectPropertyDomain(:application_context_hasContextElement
:application_context)
ObjectPropertyRange(:application_context_hasContextElement
  ObjectMinCardinality(1 :application_context_
  hasContextElement
:application_context_element))
InverseObjectProperties(:application_context_element_
  hasFrameOfReference
:application_context_hasContextElement)
```

When translating this "#171" instance of APPLICATION_CONTEXT to OWL, an individual should be created first and then the text "*configuration controlled 3D designs of mechanical parts and assemblies*" should be kept and filled as data property.

```
Declaration(NamedIndividual(<khm-14:application_context>))
ClassAssertion(<application_context><khm-14:
  application_context>)
DataPropertyAssertion(<application_context_hasApplication>
<khm-14:application_context>"configuration controlled 3D
  designs of mechanical parts and assemblies"^^xsd:string))
```

The mappings on the population of data models would also be an interesting piece of future work.

Acknowledgements. The authors would like to thank the anonymous reviewers and the editors for the valuable comments that helped them in improving the quality of this article.

Funding. This work was supported by the National Natural Science Foundation of China (Grant number: 62002031) and Scientific Research Start-up Fund of Shantou University (Grant number: NTF21042).

Competing interests. The authors declare none.

References

Agirre E, Alfonseca E, Hall K, Kravalova J, Pasca M and Soroa A (2009) A study on similarity and relatedness using distributional and WordNet-based approaches. *Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the ACL*, Boulder, Colorado, pp. 19–27.

Agostinho C, Dutra M, Jardim-Gonçalves R, Ghodous P and Steiger-Garção A (2007) *EXPRESS to OWL Morphism: Making Possible to Enrich ISO10303 Modules*. London: Springer London.

Alkahtani M, Choudhary A, De A and Harding JA (2019) A decision support system based on ontology and data mining to improve design using warranty data. *Computers & Industrial Engineering* **128**, 1027–1039.

Andres B, Poler R and Sanchis R (2021) A data model for collaborative manufacturing environments. *Computers in Industry* **126**, 103398.

Barbau R, Krima S, Rachuri S, Narayanan A, Fiorentini X, Foufou S and Sriram RD (2012) OntoSTEP: enriching product model data using ontologies. *Computer-Aided Design* **44**, 575–590.

Beetz J, van Leeuwen J and de Vries B (2008) IfcOWL: a case of transforming EXPRESS schemas into ontologies. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **23**, 89–101.

https://doi.org/10.1017/S0890060423000185 Published online by Cambridge University Press

Cer D, Diab M, Agirre E and Specia L (2017) SemEval-2017 Task 1: semantic textual similarity multilingual and cross-lingual focused evaluation.

Devlin J, Chang M-W, Lee K and Toutanova K (2019) BERT: pre-training of deep bidirectional transformers for language understanding. *2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Duchi J, Hazan E and Singer Y (2011) Adaptive subgradient methods for online learning and stochasticoptimization. *Journal of Machine Learning Research* **12**, 2121–2159.

Eslami MH, Lakemond N and Brusoni S (2018) The dynamics of knowledge integration in collaborative product development: evidence from the capital goods industry. *Industrial Marketing Management* **75**, 146–159.

Farias TMd, Roxin A and Nicolle C (2018) A rule-based methodology to extract building model views. *Automation in Construction* **92**, 214–229.

Fraga AL, Vegetti M and Leone HP (2020) Ontology-based solutions for interoperability among product lifecycle management systems: a systematic literature review. *Journal of Industrial Information Integration* **20**, 100176.

Goikoetxea J, Soroa A and Agirre E (2015) Random walks and neural network language models on knowledge bases. *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1434–1439.

Gong H, Shi L, Liu D, Qian J and Zhang Z (2021) Construction and implementation of extraction rules for assembly hierarchy information of a product based on OntoSTEP. *Procedia CIRP* **97**, 514–519.

González E, Piñeiro JD, Toledo J, Arnay R and Acosta L (2021) An approach based on the ifcOWL ontology to support indoor navigation. *Egyptian Informatics Journal* **22**, 1–13.

He P, Liu X, Gao J and Chen W (2021) DeBERTa: decoding-enhanced BERT with disentangled attention. *2021 International Conference on Learning Representations*.

Jian Q and Liu Y (2021) A mapping method between EXPRESS and OWL based on text similarity analysis. *2021 International Conference on Electronic Information Engineering and Computer Science*, IEEE.

Jurgens D, Pilehvar MT and Navigli R (2014) SemEval-2014 Task 3: cross-level semantic similarity. *International Conference on Computational Linguistics*.

Kaliszewski I and Podkopaev D (2016) Simple additive weighting—A metamodel for multiple criteria decision analysis methods. *Expert Systems with Applications* **54**, 155–161.

Krima S, Barbau R, Fiorentini X, Sudarsan R and Sriram RD (2009) OntoSTEP: OWL-DL ontology for STEP. Gaithersburg, MD 20899, USA, National Institute of Standards and Technology, NISTIR 7561.

Kwon S, Monnier LV, Barbau R and Bernstein WZ (2020) Enriching standards-based digital thread by fusing as-designed and as-inspected data using knowledge graphs. *Advanced Engineering Informatics* **46**, 101102.

Mikolov T, Chen K, Corrado G and Dean J (2013) Efficient estimation of word representations in vector space. arXiv:1301.3781.

Miller GA (1995) Wordnet: a lexical database for English. *Communications of the ACM* **38**, 39–41.

Miller GA and Charles WG (1991) Contextual correlates of semantic similarity. *Language and Cognitive Processes* **6**, 1–28.

Nallapati R, Zhai F and Zhou B (2017) SummaRuNNer: a recurrent neural network based sequence model for extractive summarization of documents. *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*. San Francisco, California, USA: AAAI Press, pp. 3075–3081.

Pauwels P and Terkaj W (2016) EXPRESS to OWL for construction industry: towards a recommendable and usable ifcOWL ontology. *Automation in Construction* **63**, 100–133.

Pauwels P, Krijnen T, Terkaj W and Beetz J (2017) Enhancing the ifcOWL ontology with an alternative representation for geometric data. *Automation in Construction* **80**, 77–94.

Pennington J, Socher R and Manning CD (2014) GloVe: Global Vectors for Word Representation. *2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar.

Qin Y, Lu W, Qi Q, Liu X, Zhong Y, Scott PJ and Jiang X (2017) Status, comparison, and issues of computer-aided design model data exchange methods based on standardized neutral files and web ontology language file. *Journal of Computing and Information Science in Engineering* **17**, 010801.1–010801.10.

Ramos L (2015) Semantic web for manufacturing, trends and open issues. *Computers & Industrial Engineering* **90**, 444–460.

Rubenstein H and Goodenough JB (1965) Contextual correlates of synonymy. *Communications of the ACM* **8**, 627–633.

Saaty TL (1980) *The Analytic Hierarchy Process: planning, Priority Setting, Resources Allocation*. New York: McGraw.

Schevers H and Drogemuller R (2005) Converting the industry foundation classes to the Web Ontology Language. *2005 First International Conference on Semantics, Knowledge and Grid*.

Terkaj W and Šojić A (2015) Ontology-based representation of IFC EXPRESS rules: an enhancement of the ifcOWL ontology. *Automation in Construction* **57**, 188–201.

Tien NH, Le NM, Tomohiro Y and Tatsuya I (2019) Sentence modeling via multiple word embeddings and multi-level comparison for semantic textual similarity. *Information Processing & Management* **56**, 102090.

Wagner A, Sprenger W, Maurer C, Kuhn TE and Rüppel U (2022) Building product ontology: core ontology for linked building product data. *Automation in Construction* **133**, 103927.

Wang Y-J (2019) Interval-valued fuzzy multi-criteria decision-making based on simple additive weighting and relative preference relation. *Information Sciences* **503**, 319–335.

Zheng C, Xing J, Wang Z, Qin X, Eynard B, Li J, Bai J and Zhang Y (2022) Knowledge-based program generation approach for robotic manufacturing systems. *Robotics and Computer-Integrated Manufacturing* **73**, 102242.

Yan Liu received the bachelor degree in software engineering from Harbin Institute of Technology, China; master degree in software engineering from Shanghai Jiaotong University, China; and the PhD degree from the School of Engineering and Innovation, The Open University, U.K. She is an associate professor at Shantou University, Guangdong, China. Her current research interests include intelligent decision-making, knowledge representation and acquisition, and product development process modelling.

Qingquan Jian received his master degree from the School of Computer Science and Technology, Changchun University of Science and Technology, China. He is working as an engineer in a technology company, Guangdong, China. His research work focuses on natural language processing techniques and their applications.

Claudia M. Eckert received the Ph.D. degree in design from The Open University, Milton Keynes, U.K., in 1997, and the M.Sc. degree in applied artificial intelligence from the University of Aberdeen, Aberdeen, U.K., in 1990. She is a professor of design at The Open University, Milton Keynes, U.K. Her research interests include understanding and supporting design processes, and in particular, engineering change and processes planning. She is also working on comparisons between design domains.

https://doi.org/10.1017/S0890060423000185 Published online by Cambridge University Press

Appendix A

This section presents the mapping results for 12 EXPRESS declarations about high-level product definition where the OWL models are defined in the way of Figure 6a that the properties of the entity in EXPRESS are defined as object properties in OWL. For a clear presentation, the mapped OWL segments are labeled in Table A1 and the results are listed in Table A2.

Table A1. The label of OWL segments (in the format of Fig. 6a)

Label	OWL segment
A1	Declaration(Class(:application_context_element)) Declaration(ObjectProperty(:application_context_element_hasName)) ObjectPropertyDomain(:application_context_element_hasName : application_context_element) ObjectPropertyRange(:application_context_element_hasName :label) Declaration(ObjectProperty(:application_context_element_hasFrameOfReference)) ObjectPropertyDomain(:application_context_element_hasFrameOfReference : application_context_element) ObjectPropertyRange(:application_context_element_hasFrameOfReference : application_context)
A2	Declaration(Class(:application_context)) Declaration(ObjectProperty(:application_context_hasApplication)) ObjectPropertyDomain(:application_context_hasApplication :application_context) ObjectPropertyRange(:application_context_hasApplication :text) Declaration(ObjectProperty(:application_context_hasContextElement)) ObjectPropertyDomain(:application_context_hasContextElement :application_context) ObjectPropertyRange(:application_context_hasContextElement ObjectMinCardinality(1 :application_context_hasContextElement : application_context_element)) InverseObjectProperties(:application_context_element_hasFrameOfReference : application_context_hasContextElement)
A3	Declaration(Class(:product_context)) SubClassOf(:product_context :application_context_element) Declaration(ObjectProperty(:product_context_hasDisciplineType)) ObjectPropertyDomain(:product_context_hasDisciplineType :product_context) ObjectPropertyRange(:product_context_hasDisciplineType :label)
A4	Declaration(Class(:product_definition_context)) SubClassOf(:product_definition_context :application_context_element) Declaration(ObjectProperty(:product_definition_context_hasLifeCycleStage)) ObjectPropertyDomain(:product_definition_context_hasLifeCycleStage : product_definition_context) ObjectPropertyRange(:product_definition_context_hasLifeCycleStage :label)
A5	Declaration(Class(:product_concept_context)) SubClassOf(:product_concept_context :application_context_element) Declaration(ObjectProperty(:product_concept_context_hasMarketSegmentType)) ObjectPropertyDomain(:product_concept_context_hasMarketSegmentType : product_concept_context) ObjectPropertyRange(:product_concept_context_hasMarketSegmentType :label)
A6	Declaration(Class(:identifier)) SubClassOf(:identifier :String) DataPropertyDomain(:hasString :String) DataPropertyRange(:hasString xsd:string)
A7	Declaration(Class(:text)) SubClassOf(:text :String) DataPropertyDomain(:hasString :String) DataPropertyRange(:hasString xsd:string)
A8	Declaration(Class(:label)) SubClassOf(:label :String) DataPropertyDomain(:hasString :String) DataPropertyRange(:hasString xsd:string)
A9	Declaration(Class(:product)) Declaration(ObjectProperty(:product_hasID)) ObjectPropertyDomain(:product_hasID :product) ObjectPropertyRange(:product_hasID ObjectExactCardinality(1 :product_hasID :identifier)) Declaration(ObjectProperty(:product_hasName)) ObjectPropertyDomain(:product_hasName :product) ObjectPropertyRange(:product_hasName :label) Declaration(ObjectProperty(:product_hasDescription)) ObjectPropertyDomain(:product_hasDescription :product) ObjectPropertyRange(:product_hasDescription :text) Declaration(ObjectProperty(:product_hasFrameOfReference)) ObjectPropertyDomain(:product_hasFrameOfReference :product) ObjectPropertyRange(:product_hasFrameOfReference ObjectMinCardinality(1 :product_hasFrameOfReference :product_context))

(*Continued*)

https://doi.org/10.1017/S0890060423000185 Published online by Cambridge University Press

Table A1. (*Continued.*)

Label	OWL segment
A10	Declaration(Class(:source)) EquivalentClasses(:source ObjectOneOf(:bought :made :not_known))
A11	Declaration(Class(:product_definition_formation)) Declaration(ObjectProperty(:product_definition_formation_hasID)) ObjectPropertyDomain(:product_definition_formation_hasID : product_definition_formation) ObjectPropertyRange(:product_definition_formation_hasID ObjectExactCardinality(1 :product_definition_formation_hasID :identifier)) Declaration(ObjectProperty(:product_definition_formation_hasDescription)) ObjectPropertyDomain(:product_definition_formation_hasDescription : product_definition_formation) ObjectPropertyRange(:product_definition_formation_hasDescription :text) Declaration(ObjectProperty(:product_definition_formation_hasOfProduct)) ObjectPropertyDomain(:product_definition_formation_hasOfProduct : product_definition_formation) ObjectPropertyRange(:product_definition_formation_hasOfProduct ObjectExactCardinality(1 :product_definition_formation_hasOfProduct :product))”
A12	Declaration(Class(:product_definition_formation_with_specified_source)) SubClassOf(:product_definition_formation_with_specified_source : product_definition_formation) Declaration(ObjectProperty(:product_definition_formation_with_specified_source_hasMakeOrBuy)) ObjectPropertyDomain(:product_definition_formation_with_specified_source_hasMakeOrBuy :product_definition_formation_with_specified_source) ObjectPropertyRange(:product_definition_formation_with_specified_source_hasMakeOrBuy :source)

Table A2. The mapping results for 12 EXPRESS declarations

EXPRESS declaration	OWL segment	Similarity
ENTITY application_context_element ; SUPERTYPE OF(ONEOF (library_context, product_concept_context, product_context, product_definition_context)); name : label; frame_of_reference : application_context; END_ENTITY;	A1 A2 A4 A5	0.540 0.433 0.416 0.416
ENTITY application_context; application : text; INVERSE context_elements : SET [1:?] OF application_context_element FOR frame_of_reference; END_ENTITY;	A2 A1 A9 A4	0.571 0.476 0.423 0.405
ENTITY product_context; SUBTYPE OF (application_context_element); discipline_type : label; END_ENTITY;	A3 A5 A4 A1	0.567 0.438 0.432 0.430
ENTITY product_definition_context; SUBTYPE OF (application_context_element); life_cycle_stage : label; END_ENTITY;	A4 A5 A1 A3	0.564 0.445 0.413 0.396
ENTITY product_concept_context; SUBTYPE OF (application_context_element); market_segment_type : label; END_ENTITY;	A5 A4 A1 A3	0.563 0.428 0.420 0.406
TYPE identifier = STRING; END_TYPE;	A6 A11 A12 A2	0.493 0.439 0.421 0.418
TYPE text = STRING; END_TYPE;	A7 A11 A9 A5	0.491 0.431 0.411 0.404
TYPE label = STRING; END_TYPE;	A8 A11 A1 A5	0.491 0.433 0.415 0.412

(*Continued*)

https://doi.org/10.1017/S0890060423000185 Published online by Cambridge University Press

Table A2. (*Continued.*)

EXPRESS declaration	OWL segment	Similarity
ENTITY product; id : identifier; name : label; description : OPTIONAL text; frame_of_reference : SET [1:?] OF product_context; END_ENTITY;	A9 A11 A1 A2	0.534 0.416 0.415 0.393
TYPE source = ENUMERATION OF (made, bought, not_known); END_TYPE;	A10 A12 A11 A9	0.495 0.411 0.389 0.377
ENTITY product_definition_formation; id: identifier; description: OPTIONAL text; of_product: product; UNIQUE UR1: id,of_product; END_ENTITY;	A11 A12 A9 A1	0.566 0.434 0.432 0.391
ENTITY product_definition_formation_with_specified _source; SUBTYPE OF (product_definition_formation); make_or_buy: source; END_ENTITY;	A12 A11 A1 A4	0.579 0.454 0.413 0.410

https://doi.org/10.1017/S0890060423000185 Published online by Cambridge University Press

Appendix B

This section presents the mapping results for 12 EXPRESS declarations about high-level product definition are defined in the way of Figure 6b that the properties of the entity in EXPRESS are defined as data properties in OWL. For a clear presentation, the mapped OWL segments are labeled in Table B1 and the results are listed in Table B2.

Table B1. The label of OWL segments (in the format of Fig. 6b)

Label	OWL segment
B1	Declaration(Class(:application_context_element)) Declaration(DataProperty(:application_context_element_hasName)) DataPropertyDomain(:application_context_element_hasName :application_context_element) DataPropertyRange(:application_context_element_hasName xsd:string) ObjectPropertyDomain(:application_context_element_hasFrameOfReference : application_context_element) ObjectPropertyRange(:application_context_element_hasFrameOfReference : application_context)
B2	Declaration(Class(:application_context)) Declaration(DataProperty(:application_context_hasApplication)) DataPropertyDomain(:application_context_hasApplication :application_context) DataPropertyRange(:application_context_hasApplication xsd:string) Declaration(ObjectProperty(:application_context_hasContextElement)) ObjectPropertyDomain(:application_context_hasContextElement :application_context) ObjectPropertyRange(:application_context_hasContextElement ObjectMinCardinality(1 :application_context_hasContextElement : application_context_element)) InverseObjectProperties(:application_context_element_hasFrameOfReference :application_context_hasContextElement)
B3	Declaration(Class(:product_context)) SubClassOf(:product_context :application_context_element) Declaration(DataProperty(:product_context_hasDisciplineType)) DataPropertyDomain(:product_context_hasDisciplineType :product_context) DataPropertyRange(:product_context_hasDisciplineType xsd:string)
B4	Declaration(Class(:product_definition_context)) SubClassOf(:product_definition_context :application_context_element) Declaration(DataProperty(:product_definition_context_hasLifeCycleStage)) DataPropertyDomain(:product_definition_context_hasLifeCycleStage : product_definition_context) DataPropertyRange(:product_definition_context_hasLifeCycleStage xsd:string)
B5	Declaration(Class(:product_concept_context)) SubClassOf(:product_concept_context :application_context_element) Declaration(DataProperty(:product_concept_context_hasMarketSegmentType)) DataPropertyDomain(:product_concept_context_hasMarketSegmentType : product_concept_context) DataPropertyRange(:product_concept_context_hasMarketSegmentType xsd:string)
B6	Declaration(Class(:product)) Declaration(DataProperty(:product_hasID)) DataPropertyDomain(:product_hasID :product) DataPropertyRange(:product_hasID xsd:string) Declaration(DataProperty(:product_hasName)) DataPropertyDomain(:product_hasName :product) DataPropertyRange(:product_hasName xsd:string) Declaration(DataProperty(:product_hasDescription)) DataPropertyDomain(:product_hasDescription :product) DataPropertyRange(:product_hasDescription xsd:string) Declaration(ObjectProperty(:product_hasFrameOfReference)) ObjectPropertyDomain(:product_hasFrameOfReference :product) ObjectPropertyRange(:product_hasFrameOfReference ObjectMinCardinality(1 :product_hasFrameOfReference :product_context))
B7	Declaration(Class(:product_definition_formation)) Declaration(DataProperty(:product_definition_formation_hasID)) DataPropertyDomain(:product_definition_formation_hasID :product_definition_formation) DataPropertyRange(:product_definition_formation_hasID xsd:string) Declaration(DataProperty(:product_definition_formation_hasDescription)) DataPropertyDomain(:product_definition_formation_hasDescription : product_definition_formation) DataPropertyRange(:product_definition_formation_hasDescription xsd:string) Declaration(ObjectProperty(:product_definition_formation_hasOfProduct)) ObjectPropertyDomain(:product_definition_formation_hasOfProduct : product_definition_formation) ObjectPropertyRange(:product_definition_formation_hasOfProduct ObjectExactCardinality(1 :product_definition_formation_hasOfProduct :product))
B8	Declaration(Class(:product_definition_formation_with_specified_source)) SubClassOf(:product_definition_formation_with_specified_source : product_definition_formation) Declaration(DataProperty(:product_definition_formation_with_specified__source_hasMakeOrBuy)) DataPropertyDomain(:product_definition_formation_with_specified__source_hasMakeOrBuy :product_definition_formation_with_specified_source) DataPropertyRange(:product_definition_formation_with_specified__source_hasMakeOrBuy xsd:string)

https://doi.org/10.1017/S0890060423000185 Published online by Cambridge University Press

Table B2. The mapping results for 12 EXPRESS declarations

EXPRESS declaration	OWL segment	Similarity
ENTITY application_context_element; SUPERTYPE OF(ONEOF (library_context, product_concept_context, product_context, product_definition_context)); name: label; frame_of_reference : application_context; END_ENTITY;	B1 B2 B6 B4	0.522 0.426 0.402 0.401
ENTITY application_context; application: text; INVERSE context_elements : SET [1:?] OF application_context_element FOR frame_of_reference; END_ENTITY;	B2 B1 B6 B4	0.562 0.481 0.417 0.411
ENTITY product_context; SUBTYPE OF (application_context_element); discipline_type : label; END_ENTITY;	B3 B5 B4 B1	0.551 0.420 0.419 0.417
ENTITY product_definition_context; SUBTYPE OF (application_context_element); life_cycle_stage : label; END_ENTITY;	B4 B5 B1 B2	0.548 0.434 0.400 0.390
ENTITY product_concept_context; SUBTYPE OF (application_context_element); market_segment_type : label; END_ENTITY;	B5 B4 B1 B3	0.549 0.418 0.405 0.392
TYPE identifier = STRING; END_TYPE;	B7 B5 B1 B8	0.434 0.428 0.427 0.422
TYPE text = STRING; END_TYPE;	B7 B1 B5 B6	0.421 0.414 0.412 0.404
TYPE label = STRING; END_TYPE;	B7 B5 B1 B6	0.434 0.408 0.407 0.401
ENTITY product; id: identifier; name: label; description: OPTIONAL text; frame_of_reference: SET [1:?] OF product_context; END_ENTITY;	B6 B7 B1 B2	0.519 0.410 0.408 0.387
TYPE source = ENUMERATION OF (made, bought, not_known); END_TYPE;	B8 B7 B6 B4	0.398 0.393 0.377 0.360
ENTITY product_definition_formation; id: identifier; description: OPTIONAL text; of_product: product; UNIQUE UR1: id,of_product; END_ENTITY;	B7 B8 B6 B1	0.564 0.427 0.422 0.393
ENTITY product_definition_formation_with_specified _source; SUBTYPE OF (product_definition_formation); make_or_buy: source; END_ENTITY;	B8 B7 B1 B6	0.565 0.452 0.418 0.410

https://doi.org/10.1017/S0890060423000185 Published online by Cambridge University Press