www.itcon.org - Journal of Information Technology in Construction - ISSN 1874-4753

GENERATING PARTIAL CIVIL INFORMATION MODEL VIEWS USING A SEMANTIC INFORMATION RETRIEVAL APPROACH

SUBMITTED: March 2019
REVISED: August 2019
PUBLISHED: January 2020 at https://www.itcon.org/2020/2
EDITOR: Kumar B.
DOI: 10.36680/j.itcon.2020.002

Tuyen Le, Assistant Professor, (Corresponding author)
Department of Civil Engineering, Clemson University, SC, USA;
tuyenl@clemson.edu

H. David Jeong, Professor,
Department of Construction Science, Texas A&M, College Station, TX, USA;
djeong@arch.tamu.edu

Stephen B. Gilbert, Associate Professor,
Department of Industrial & Manufacturing Systems Engineering, Iowa State University, IA, USA;
gilbert@iastate.edu

Evgeny Chukharev-Hudilainen, Associate Professor,
Applied Linguistics & Technology Program, Iowa State University, IA, USA;
evgeny@iastate.edu

SUMMARY: *Open data standards (e.g. LandXML, TransXML, CityGML) are a key to addressing the interoperability issue in exchanging civil information modeling (CIM) data throughout the project life-cycle. Since these schemas include rich sets of data types covering a wide range of assets and disciplines, model view definitions (MVDs) which define subsets of a schema are required to specify what types of data to be shared in accordance with a specific exchange scenario. The traditional procedure for generating and implementing MVDs is time-consuming and laborious as entities and attributes relevant to a particular data exchange context are manually identified by domain experts. This paper presents a method that can locate relevant information from a source XML data schema for a specific domain based on the user's keyword. The study employs a semantic resource of civil engineering terms to understand the semantics of a keyword-based query. The study also introduces a novel context-based search technique for retrieving related entities and their referenced objects. The developed method was tested on a gold standard of several LandXML subschemas. The experiment results show that the semantic MVD retrieval algorithm achieves a mean average precision of nearly 90%. The research is original, being a novel method for extracting partial civil information models given a keyword from the end user. The method is expected to become a fundamental tool assisting professionals in extracting data from complex digital datasets.*

KEYWORDS: *Civil Information Modeling, Model View Definition, Civil Engineering Lexicon, Information Retrieval, Context-Aware*

REFERENCE: *Tuyen Le, H. David Jeong, Stephen B. Gilbert, Evgeny Chukharev-Hudilainen (2020). Generating partial civil information model views using a semantic information retrieval approach. Journal of Information Technology in Construction (ITcon), Vol. 25, pg. 41-54, DOI: 10.36680/j.itcon.2020.002*

1. INTRODUCTION

Neutral data standards have been widely accepted as an effective means for transferring the Civil Information Modeling (CIM) data of a civil infrastructure asset between project stakeholders. Several open standards have been proposed, for instance the Industry Foundation Classes (IFC) Extension for alignment (buildingsmart 2017), LandXML (landxml.org 2017), and TransXML (Ziering et al. 2007). These standardized data models contain rich sets of data elements covering various business processes and disciplines across the project life-cycle. However, a specific data exchange scenario needs only a subset of data. For example, of those data created from the design stage for a roadway project, a contractor using automated machine guidance needs only the data associated with earthwork such as three-dimensional (3D) surface models and alignment lines. Neutral data standards alone are insufficient to facilitate seamless digital data exchange among project stakeholders (Froese 2003; East et al. 2012}. There is a need for formal definitions of subschemas specifying what types of data are needed for specific data exchange use cases.

Model View Definition (MVD), a concept introduced by the buildingSMART alliance (buildingSmart 2016), aims to fulfill the above need. An MVD is a subset of a standardized schema that represents the data interests of a particular context. The availability of these model views underpins the extraction of data from complicated sets generated throughout the project life cycle. The horizontal construction sector has adopted this MVD concept to develop a number of CIM schema views. A pioneering effort in this area is the InfraModel project carried out by the Technical Research Center of Finland that aims to define subsets of LandXML for different types of transportation assets (inframodel.fi 2017). In spite of significant research efforts, existing MVDs are yet inadequate to meet the industry demand. This is because the conventional method for developing MVDs is on a manual basis which is time-consuming and labor-intensive (Venugopal et al. 2012; Eastman 2012; Hu 2014; Lee et al. 2016a). MVD developers are required to manually translate data exchange requirements presented in a paper-based Information Delivery Manual (IDM) into a machine-readable MVD. Due to the dynamic nature of the industry, existing MVDs need to be adjusted to reflect changes in practices. In addition, the implementation of MVDs requires much effort from software vendors to develop and test data exporters. In order to remove this MVD bottleneck, it is imperative to develop a more effective methodology (Venugopal et al. 2012). The need for an automated methodology has been raised by various researchers (Yang and Eastman 2007, Venugopal et al. 2012, Lee et al. 2016b).

The primary purpose of the MVD development process is to identify classes, properties, and their referenced elements to be included in the view that reflects the data needs of a particular domain. In the current practice, developers need to interpret the semantics of the data keywords in an IDM and look for matching entities, attributes, and types in the source schema. This task becomes extremely challenging especially for such large standards as LandXML and IFC which keep growing every year. These schemas are composed of thousands of classes and attributes along with complex relations such as superclasses and subclasses. Manually finding relevant classes for a given data need is tedious, time-consuming, and error-prone (Yang and Eastman 2007; Lee et al. 2016a). Although open standards are structured using a systematic categorization method, i.e., by assets in CityGML or by disciplines in IFC, developers may still need to go through the entire schema since a use case typically requires data from across different groups. This is even more problematic when the terms used in the IDM are inconsistent with the entity names of the source schema. Such terminology discrepancies may lead to a wrong inclusion of irrelevant entities, or to a failure to find those that are actually relevant. Because of those reasons, the task of identifying relevant entities and properties becomes an important hurdle for developers and possibly involves semantic errors. A method that can assist the developers in identifying related items in the source schema can afford significant time savings and reduce the number of errors in MVD development.

Previous work on enhancing the efficiency and effectiveness of the MVD development has focused on providing tools and methods that support syntactic validation of MVDs (Yang and Eastman 2007) and improve their reusability (Lee et al. 2016a). Also, few studies were conducted to support MVD developers in searching for relevant entities and properties such as the xPPM tool (Lee et al. 2013). The existing systems, however, assume the availability of IDMs and generate MVDs based on keyword mapping between an IDM and the IFC. Due to the terminology discrepancy, keyword searching without human interference would involve many mismatches. To date, no studies were conducted that can generate a subschema without an IDM. With the state-of-the-art methods, selecting entities in the source schema to meet the data needs of particular domains still heavily relies on domain experts. A method that can allow users from diverse disciplines to search for data entities and properties in neutral

data standards would considerably accelerate the transition to seamless digital data exchange throughout the project life cycle.

This research has developed a novel IDM-free method for generating partial views from a CIM schema which is encoded in the eXtensible Markup Language (XML). XML is chosen because it is an international open standard used as the basis of various neutral data schemas for the civil infrastructure sector, such as, LandXML, TransXML, and RoadXML. The proposed method is an Information Retrieval (IR) technique that generates a ranked list of XML branches related to a given keyword query. The top retrieved results serve as suggested branches that should be used to form the desired XML subschema. The next sections describe related studies, knowledge gaps, the architecture of the framework, followed by discussions on implications and conclusions.

2. RELATED STUDIES AND KNOWLEDGE GAP

2.1. Previous Studies on Automated MVD generation

A few studies on automated translation of IDMs into machine-readable MVDs are found in the literature. A common objective of these studies was to reduce the manual task performed by developers in finding syntactically referenced elements for a domain-related class of a certain exchange need.

The method proposed by Yang and Eastman (2007) is one of the most notable studies on automated generation of IFC subsets. The study defines various rules to construct two types of subsets including `base sets' and `valid subsets'. A base set includes a base IFC entity and its dependent data types, and a valid subset is an aggregation of a base set and its syntactically referenced base sets. A semantically valid subschema for data exchange is composed of one or more valid subsets which contain domain-specific information. With respect to the semantic level, the method of Yang and Eastman (2007) offers developers with a mechanism for defining semantic rules. For example, to define the concept of a building with fire exits, the FireExit boolean attribute of the *ifc*Door object must be `TRUE'. While the syntactic rules for generating syntactically valid subsets are generic and can be applied in broad applications, developing semantically valid subschemas is largely dependent on the rules of domains of interest (Yang and Eastman 2007). Thus, the definition of semantically valid subsets formulated for a certain context might not be reusable to others.

As an attempt to address the reusability weakness of the above method, Lee (2009) introduced the concept of `minimal set' that serves as a basic legal semantic unit and can be shared by different contexts. The rules used to define a minimal set are extended from those proposed by Yang and Eastman (2007). A minimal set is a complete set of entities, properties, and their references, presenting a single real-life concept such as ‘wall’ or ‘building.’ Since this basic set itself is a complete semantic subset, it can be shared between model views. To successfully implement this method, however, a standardized ontology of concepts that is agreed by all relevant domains is needed. Lee (2009) also developed a model view generator that can support concept and entity matching. This mapping function is purely based on the label of an entity without considering its semantics. Manual selection of entities is still required for establishing the minimal set for every domain concept.

As discussed, validating the syntactic correctness of a model view is currently well supported by various rule-based algorithms. This automated validation function can significantly reduce the burden on software developers; they, however, are still required to have a deep understanding of the semantics of the IFC schema to properly match with the data exchange elements in an IDM.

2.2. Knowledge Gap

The state-of-the-art on model view generation focuses on validating the syntax of a subset. The task of finding relevant entities for particular information need is still heavily dependent on developers. Prior studies assume that developers are aware of the semantics of the entity terminology in the source schema. Under this assumption, it is especially challenging for developers to find relevant source entities in a large and complex schema. Therefore, research is needed to offer effective tools and methods that can assist developers in quickly retrieving relevant classes given an input query. Generating MVDs using a language closer to human being would provide ease of use to the end user (Jiang et al. 2015).

3. KEYWORD-DRIVEN METHODOLOGY FOR GENERATING XML SUBSCHEMAS

The majority of the existing data standards in the civil engineering sector are presented in the XML format such as LandXML, TransXML, CityGML. The buildingSMART IFC schema, which is originally developed in the EXPRESS structure, is also available in the ifcxml format. Because of the popularity of XML, the MVD generation method proposed in this study is specially designed for XML schemas for civil infrastructure projects.

The keyword-driven MVD generator developed in this study is an IR-like system that aims to obtain, from the source schema network, a ranked list of XML schema paths relevant to a keyword-based query. Keyword-based queries are a popular means for the user to express their information needs in finding relevant data and information. Most of the queries are found to be shorter than three words (Arampatzis and Kamps 2008). Exploring the semantics behind an input keyword would help to capture the user's data and information interests. For example, given the query 'traffic', the required partial view of LandXML must include every entity associated with traffic such as 'Lanes', 'TrafficControl', 'Speeds'. In this study, we will look at single keyword queries rather than free-length ones. The top branches retrieved from the system are expected to be a useful starting point for developers to select entities and attributes that will form a schema subset. The architecture of the system, as shown in Figure 1, encompasses the following key components which will be discussed in more detail in the next sections.

1. Schema network construction and indexing. This first stage aims to construct a directional network of the source CIM schema.
2. Query expansion. This step allows the system to search for semantically related entities rather than pure name matches.
3. Entity search. At this stage, those nodes of the network that are the most semantically relevant to the user's input are located/.
4. MVD branch traversal. This phase aims to implement a traversal technique on the source schema network to collect syntactically related classes (superclasses, referenced elements, data types, etc.) for those semantically related entities found in the previous stage.

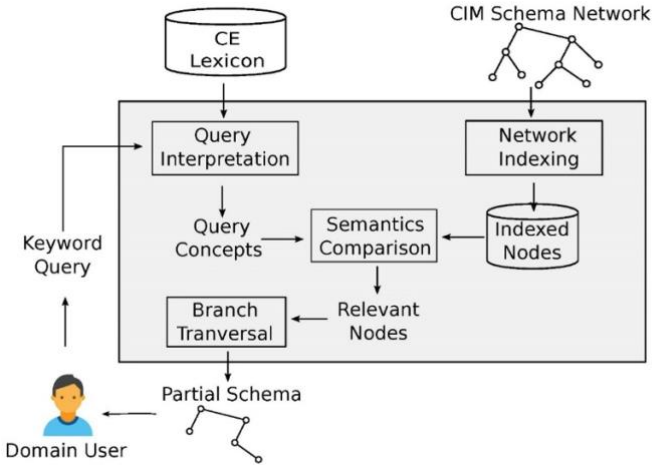

FIG. 1: Overall architecture of the MVD generation method

4. SCHEMA NETWORK CONSTRUCTION AND INDEXING

In this study, an XML schema is re-structured as an unlabelled directed network where every vertex is an XML schema component, and every edge is a relationship between components. The root node represents the entire schema. The components of an XML schema are classified into *elements*, *types* (simpleTypes and complexTypes), and *attributes* (Thompson et al. 2004). All of these components are utilized for generating the network vertices. Types specify the data range of elements and attributes. An unlabelled directed edge from node A to node B is established if A relates to B through either a non-inheritance relation (*whole-part, has-attribute, type-of*) or an inheritance relation (*is-a*). Figure 2 illustrates different types of non-inheritance relations of an XML schema. Whole-part (or has-part) refers to the compositional relation between elements. The has-attribute relation defines a characteristic of an element or a type. The type-of relation assigns a data type to an attribute or element. The inheritance is-a relationship, as shown in Figure 3, refers to the extension or restriction definition of a base type.

The is-a is from the extension or restriction type to the base type. In this study, a predecessor is called a parent, and a successor is called a child.

FIG. 2: XML non-inheritance relations

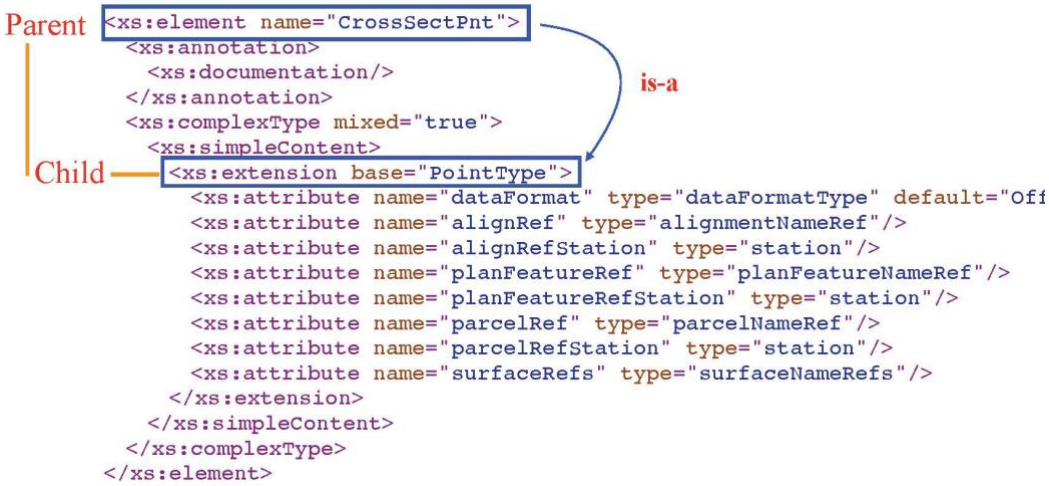

FIG. 3: XML inheritance relation

Indexing the information sources is a prerequisite task for any IR system. This step represents the source items in such a format that the search engine can effectively evaluate their relevance to the user's query. The relevance is measured by analyzing the representative information of a source item. Each node is assigned a unique identification (ID) number. The semantics of a node is defined by the name of itself and of surrounding nodes. Context nodes include parent and child nodes. In specific, a source node e is indexed by three features including (1) node name, (2) parent nodes, and (3) child nodes as follows.

$$e = (e.name, e.parents, e.children) \quad (1)$$

where *e.parents* is a set of the names of parents and grandparents of the node e, and *e.children* set includes the names of children and grandchildren nodes in the XML network. Stop nodes that are common geometric attributes of physical objects (e.g., width, length, area, type) are eliminated from the *e.children* set. Stop nodes are shared information of various classes and they provide little representative characteristics. Since this study evaluates the semantic similarity based on the number of common context nodes (as presented later), the discard of stop properties will help to dismiss their effects on the similarity score.

Entity names in a data standard do not necessarily follow the grammatical rules of English. Developers can define their own rules to name classes. Therefore, they may not match technical terms verbatim. For example, the label `ProfAlign' in the LandXML schema stands for the real term `profile alignment'. Searching for entities based on such abbreviated labels might fail to properly evaluate their relevance. To mitigate this problem, the indexed nodes are renamed to a form called `natural name' that is closer to the domain terminology. The natural name of an entity is inferred by comparing its original computer-friendly label with the description texts provided in the referenced meta data document of a data schema. This process includes two steps. First, the class label is split into tokens at special characters (e.g., '.', '-', '_') or upper case characters. Second, the referenced description text of the class in the schema is scanned for the full-word version of every token. The similarity between a token and a word in the description is measured using the Levenshtein edit-distance algorithm (Gale and Church 1993). The top match is used as the guessed natural name of a token. For example, the label `ProfAlign' will be tokenized into `Prof' and 'Align' in the first step. These tokens are then compared with the words in the description of the `ProfAlign' in the schema ("The "ProfAlign" element will typically represent a proposed vertical alignment for a profile"). `Profile' and `Alignment' are the most similar words of `Prof' and `Align' respectively; thus, `Profile Alignment' is accepted as the natural name of the class label `ProfAlign'.

5. QUERY SEMANTICS INTERPRETATION

This stage aims to infer the intention of the user by computing the semantics of the input query. Keyword-based queries are a common way for the user to express their needs when searching for information. To interpret the meaning of a query, this study implements a civil engineering lexicon, namely CeLex, as a domain knowledge base. This is a lexical resource of civil infrastructure concepts, storing the semantics of the technical keywords in the domain. Using this knowledge base, the context terms relevant to a query can be identified. By using the relevant terms as supplementary queries during the retrieval process, the system is able to capture all source entities related to the user's need. Sections below explain in detail CeLex and the query expansion process.

5.1. Domain knowledge base

A domain knowledge base is critical to understand the user's input keyword. To support inferring the semantics of a user query, this study utilizes CeLex[1] as the underlying domain knowledge base. CeLex is a lexicon that we constructed using our NLP toolkit, namely CeTermClassifer (Le and Jeong 2017). This is a machine learning based system that can automatically extract civil engineering terms and learn their lexical relations from a corpus of domain texts. The outcome provided by the system is a lexical space in which the semantics of a term is represented as a high-dimensional vector. The closeness between points in this space represents the semantic relatedness between the corresponding terms. In this model, terms are connected to one another through one of the following lexical links: synonymy (similar-to), hyponymy (is-a), hypernym (reverse is-a), meronymy (part-of), and holonymy (reverse part-of). A pair of terms that are close to each other, but their specific lexical type is not detected by the system are called `fuzzynyms'.

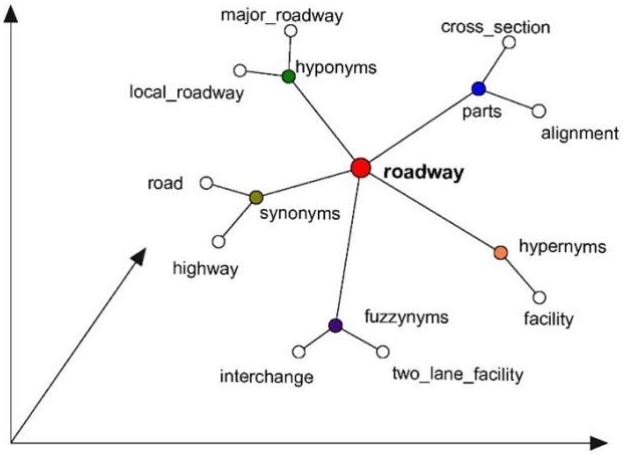

FIG. 4: Partial Civil Engineering Lexicon

The CeLex lexicon utilized in this study was obtained by implementing the CeTermClassifier system on a 16-million-word corpus comprising 38 highway design manuals for 30 State Departments of Transportation. CeLex provides semantically equivalent and related terms for 17,000 individual technical keywords. Figure 4 illustrates

the classified related terms of `roadway' in the CeLex lexical space. By mapping the user's keyword query to this space, synonyms and context terms where the keyword occurs can be identified. Those related terms are used for expanding the original input query.

5.2. Keyword expansion and query concept formulation

Keyword-based search, which returns only items that contain the input term string, is widely recognized to have a number of important weaknesses. Most problems of this approach are associated with terminological discrepancy. For example, synonymy can lower the recall of relevant entities. Also, polysemy, which refers to the multiplicity of term meanings, would lead to an inclusion of wrong items. In addition, it is not able to capture those source entities that are hypernyms or hyponyms of the input term. To overcome these problems, the user tends to conduct various search queries. However, they are required to have a deep knowledge of the target domain vocabulary. Additionally, finding relevant entities by manually trying all possible keywords is time-consuming and a number of good input options can still be missed.

In this study, a semantic model view generation method was developed that can allow for the retrieval of all relevant schema components given a single keyword by the end user. For example, with the keyword `drainage', all the related entities representing different drainage structures such as `ditch', `channel', `pipe' should be obtained. This semantic search feature provides flexibility to the end user and helps to minimize the number of keywords used for entity retrieval. Given this requirement, interpreting the semantics of a query keyword to understand the user's intent is crucial to the quality of retrieved list.

In order to analyze the data needs suggested by an input keyword, its semantically equivalent and related terms need to be identified and included in the query. These additional keywords are those terms (e.g., synonyms, hyponyms) that directly link to the user's original term in the CeLex network. As a result, an original query q_0 is extended to a set of queries Q which is the union of q_0 and a set of additional terms K directly connected to q_0 in the CeLex lexicon. For example, the keyword `drainage' will generate an expanded set of keywords including drainage, ponding, storm sewer, outlet, surface runoff, channel, etc. The equation below presents the expanded query set for a certain single keyword. If the lexicon vocabulary does not contain q_0, the K set becomes empty and Q includes only the original query.

$$Q = q_0 \cup K \tag{2}$$

Since the expanded query set includes synonyms and other important terminologies of a specific context, it can be used for finding entities and generating a subschema. The terminologies are ambiguous due to the *polysemy* issue. Polysemy refers to the issue in which different concepts are represented by the same term. Simply binding a term to an entity based on string similarity, wrong items may be retrieved. In order to address this ambiguity problem, this study introduces the idea of *concept query*. A concept query q_c for a keyword query q is defined as a triple of concept name, parent context terms, and children context terms as follow:

$$q^c = (q, q.parents, q.children) \tag{3}$$

where *q* is a keyword in the expanded keyword set, *q.parents* is a set of CeLex terms to which *q* relates through the *is-a* edge, and *q.children* are the parts or hyponyms of q. With the above `keyword to concept query' transformation method, the expanded keyword set *Q* correspondingly generates a set of concept queries Q_c.

6. NODE MATCHING AND RANKING

The relevance between a node of the schema network and the end user keyword is measured in this stage. The relatedness measure of a certain node *e*, as defined in Equation 4, is the accumulation of its similarity with every concept query q_c in Q_c. The nodes are ranked by their relatedness scores and those lower than a threshold σ are eliminated.

$$\alpha_e = \sum_{q^c \epsilon Q^c} \alpha_{e,q^c} \tag{4}$$

In the equation above, α_{e,q^c} is the similarity between concept query q_c in the Q_c set and node *e* of the schema network. As given in Equation 5, the concept similarity α_{e,q^c} is a weighted sum of two different measures of similarity: concept name matching(α^n_{e,q^c}) and context matching (α^c_{e,q^c}) where ω_n and ω_c respectively represent the weight of each matching type. Concept name matching measures the similarity in name between a pair of a

concept query and a source entity, and context matching is based on the commonality of their parents and children. Both of these measures are based on `label string similarity'. Sections below explain the measures in detail.

$$\alpha_{e,,q^c} = \omega_n \alpha^n_{e,q_c} + \omega_c \alpha^c_{e,q_c}, \qquad where \qquad \omega_n + \omega_c = 1 \tag{5}$$

6.1. Label String Similarity

Classes and properties in a data schema are represented by their names. The matching between the target and source sequences is a key metric for finding matched instances. The Levenshtein edit-distance algorithm (Gale and Church 1993) is one of the most common methods for measuring the similarity between a given pair of sequences. However, like other popular string-based algorithms, this measure is computed on characters instead of words. Thus, using this method, a certain high level of similarity might be given to a pair of two different labels that share little common semantics such as `trail' and `rail'. To eliminate such matching errors, this study proposes to evaluate similarity based on words rather characters.

In this study, the similarity between a pair of labels (*a*,*b*) is determined by the ratio of the number of common words over the total unique words, as defined in Equation 7, where the *words(x)* function returns a list of words composing a given label name. For example, with the pair *(traffic sign, road sign)*, there is 1 common word (sign) in a total of 3 unique words (traffic, sign, and road); thus, their similarity is 1/3 (33%).

$$sim_{string}(a,b) = \frac{words(a) \cap words(b)}{words(a) \cup words(b)} \tag{6}$$

6.2. Concept name matching - α^n_{e,q_c}

Concept name is an important indicator of semantic similarity. The degree of overlapping in name between an input concept query q_c and a source entity e reflects a certain level of semantic similarity between them. Concept name similarity α^n_{e,q_c} is based on the label string similarity measure and is computed as:

$$\alpha^n_{e,q_c} = sim_{string}(e.name, q^c.name) = \frac{|words(e.name) \cap words(q^c.name)|}{|words(e.name) \cup words(q^c.name)|} \tag{7}$$

6.3. Context matching - α^c_{e,q_c}

This measure compares the context items of the target and source concepts. The consideration of context is to reduce mismatches due to the polysemy issue. By comparing their attributes and other related entities, their meaning difference can be detected. The similarity from this viewpoint can be measured by the commonality and difference between their context entities. In this study, context is classified into parent and children contexts. The overall context similarity measure is the average of parent α^{cp}_{e,q_c} and children similarity α^{cc}_{e,q_c}as follows.

$$\alpha^c_{e,q_c} = \frac{\alpha^{cp}_{e,q_c} + \alpha^{cc}_{e,q_c}}{2} \tag{8}$$

Context similarity is commonly measured as a function of common and distinctive features of entities compared (Tversky 1977). In this study, the context similarity between a concept query and a source entity disregards the degree of difference in their features. Since the existing civil data schemas reflect only a small number of disciplines, the attributes included to define a class in the source schema are still incomplete. Whereas, a concept query which is formulated based on the CeLex lexicon, includes a large number of attributes. Assessing the similarity using the distinction information may involve a great bias. This paper evaluates the similarity based only on what they share in common. A context similarity is defined as a logarithm function of the total string similarity score for all pairs of a context term in the concept query with one another in the source entity. The context similarity score is within the range [0-1]. The context similarity measures for parent and children context similarity are respectively shown in Equations 9 - 10:

$$\alpha^{cp}_{e,q_c} = \min\left(1, log_{10}\left[1 + \sum_{m \epsilon M_p} \sum_{n \epsilon N_p} sim_{string}(e_{p,m}, q^c_{p,n})\right]\right) \tag{9}$$

$$\alpha^{cc}_{e,q_c} = \min\left(1, log_{10}\left[1 + \sum_{m \epsilon M_c} \sum_{n \epsilon N_c} sim_{string}(e_{c,m}, q^c_{c,n})\right]\right) \tag{10}$$

where M_p and N_p respectively denote the collection of parent context terms of a source entity e and a concept query q_c. M_c and N_c represent children context term sets.

7. BRANCH SEARCH AND MVD COMPOSITION

A subschema must include all referenced entities (e.g., datatypes, superclasses, referenced elements) to be syntactically complete. This stage aims to identify syntactic references of those matched nodes identified in the previous stage to generate a partial network. As explained earlier, every component of the source schema is included in the network and connects to each other. Thus, tracing referenced entities through the edges can help to find all required nodes for a legal subschema. Sections below describe the proposed method for generating a subschema given a list of nodes.

7.1. Branch Traversal Algorithm

To retrieve referenced nodes of a given node, this study adapted a traversal algorithm for ontology view extraction proposed by Seidenberg and Rector (2006). Ontology and directed network both represent concepts and relations using a graph structure. The algorithm is illustrated in Figure 5. As shown, the traversal algorithm aims to search for chains of related nodes. The algorithm includes upwards and downwards traversals. The upwards propagation starts with a target node and follows links that connect to its parents, then the parents of propagated nodes, and so on. For example, upwards tracing will help to identify the *Alignments* and *Landxml* elements from the *Alignment* element in the LandXML schema. The downwards is to add to the extract with children and grandchildren at all levels below the original target nodes. This is to ensure the inclusion of all definition such as attributes, constituent elements, data types of the target nodes and their children. The algorithm does not apply the downwards path on those new nodes obtained from the upwards traversing to avoid the adding of the entire schema to the extract. The traversal is an iteration process which continues until no new node can be found. In order to avoid endless traversing due to the issue of circular self-reference and cyclical dependencies between schema elements (as shown in Figure 6), the constraint of `meeting just once' is applied for both upwards and downwards directions. This condition ensures that the traversal will ignore paths that lead to an old node which is already added.

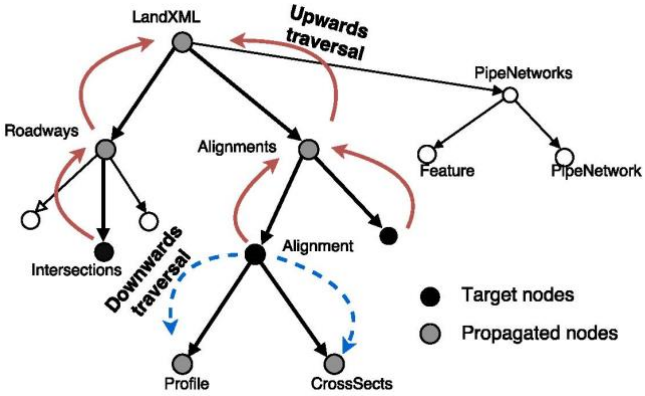

FIG. 5: Traversal approach to populate schema branches

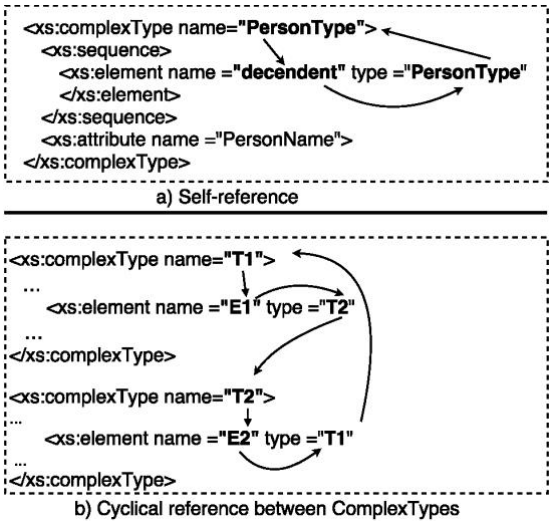

FIG. 6: Cyclic dependencies

When traversing throughout the network is completed, a collection of many paths can be obtained for every target node. A path (also called branch) is a chain starting from the root to a leaf node through a target node. A given single target node *e* will generate a set of paths *L*. The relatedness between a schema path *l* in *L* and the user query (as illustrated in Equation 11) is inherited from the relatedness score of the target node.

$$\beta_l = \alpha_e \tag{11}$$

7.2. Branch Merging for Subnetwork Formation

For a given input keyword, multiple target nodes can be found, and accordingly various sets of branches will be obtained. Since the matched paths might be overlapped, merging is necessary to eliminate duplication and allow for the generation of a single subschema.

In order to merge different branches, they are broken down into a set of separate segments each of which is an edge linking a certain pair of vertices. For example, the path 'Landxml →Roadway → Alignments → Alignment' will accordingly generate the following segments {Landxml → Roadway, Roadway → Alignments, Alignments → Alignment}. The final subnetwork is defined as a union of all sets of branch segments. The relatedness between segment *g* of the path *l* and the user keyword inherits the relatedness score of its original path *l*, defined as follows.

$$\lambda_{g,l} = \beta_l \tag{12}$$

Since a unique segment may appear in multiple branches, its score is aggregated from all retrieved paths. The equation below shows the method for accumulating the score of a branch segment in the set of accepted paths *L*:

$$\lambda_g = \sum_{l \in L} \beta_l \cdot \pi(g, l) \tag{13}$$

where $\pi(g, l)$ is 1 if segment *g* appears in path *l*, and 0 otherwise.

8. EVALUATION AND DISCUSSION

8.1. Experiment setup

To evaluate the proposed method, an experiment was carried out on the LandXML 2.0 schema which consisted of nearly 2,5000 entities and attributes. In this experiment, a gold standard which serves as a testing data set was developed to evaluate the accuracy of the developed subschema generator. The gold standard contains seven keywords ('surface model', 'alignment', 'roadside', 'pavement', 'bridge', 'drainage', and 'traffic') and their corresponding manually constructed LandXML subschemas which are represented as a list of branch segments. Table 1 shows the test keywords along with several excerpts of segments in the gold standard[2]. To construct this testing dataset, seven keywords representing different contexts of civil engineering were selected. The test queries selected must represent a domain of knowledge that is covered in the source schema. We manually identified all the relevant network branches and segments in the source schema for each of the keywords to construct the test MVDs. The existing Finnish specification of LandXML subschemas was used as a reference resource to find the relevant information to be included in the views. We also interviewed several local contractors who actively implemented automated construction technologies such as Automated Machine Guidance and Stringless Paving, to verify those views regarding the construction domain. The testing data set was compared with the retrieved MVDs returned by the system to evaluate the system's performance.

TABLE 1: Gold standard of LandXML subsets

No.	Query	Segment Count	Segment example
1	alignment	2	LandXML→Aligments
2	pavement	4	GradeSurface→Zones
3	surface model	31	Surface→SourceData
4	roadside	7	Roadway→Roadside
5	drainage	18	Roadside→Ditch
6	bridge	7	RoadSurface→Zones
7	traffic	28	Roadway→Lanes

This study adopted popular evaluation measures in IR research to assess the proposed method. For every keyword in the gold standard, the system generates a ranked list of schema branches/segments. If the system is perfect, all the true branches will occupy the top positions. In order to evaluate the system, the Mean Average Precision (MAP) measure was used. MAP is a unique measure that represents both precision and recall of the system. Recall is defined as the ratio of correctly retrieved segments to the total true segments in the gold standard. It varies from 0% to 100% depending on the cut off position of the ranked list. Precision is the fraction of retrieved segments that are truly relevant to the end user's keyword. The MAP value for a set of input keywords is the mean of the average precision at different recall levels for each keyword, defined as follows.

$$MAP = \frac{1}{Q}\sum_{j=1}^{j=Q}\frac{1}{N_j}\sum_{k=1}^{k=N_j} Precision(R_{jk}) \quad (14)$$

In the equation above, Q is the test keyword set, N_j is the number of true relevant branch segments for keyword j_{th} in the gold standard. *Precision(R_{jk})* refers to the precision at *k%* recall for keyword j_{th}.

To demonstrate the success of this study, a comparison in retrieval accuracy between the proposed context-based model and a baseline keyword-based search model was conducted. This baseline is purely based on the matching of an input keyword with entity names. These two models were compared using the MAP metric. The precisions at several recall levels including 10%, 30%, and 50% were also reported. In addition, to explore the importance of concept name matching and context matching to the system performance, different weight settings were tested. The evaluation results are discussed in the following section.

8.2. Results and discussions

Tables 2 and 3 illustrate the LandXML subschema retrieved by the designed system for the query `drainage' in two different representation formats. With the first format (see Table 2), the retrieved subnetwork is represented as a ranked list of full branches connecting the root and the relevant data entities. Presenting the results in this way helps to visualize all the referenced nodes for a semantical leaf node, but the relatedness measure for a specific segment of the subnetwork is not explicitly presented. Alternatively, the system can present the MVD subset as a ranked collection of segments (see Table 3). Presenting the results in this way will help to clearly show the relevance for each of the branch segments in the network with the input keyword. However, users are required to connect segments if they need to find a full path to the root. To fully benefit from the advantages of these two formats, users would need to read the results in both ways.

TABLE 2: Top retrieved branches for query 'drainage'

Rank	Top Retrieved Branches	β_l	Relevant?
1	LandXML→Roadways→Roadway→Roadside→Ditch	3.33	yes
2	LandXML→GradeModel→GradeSurface→Zones→Zone	3.19	yes
3	LandXML→PipeNetworks→PipeNetwork→Pipes→Pipe→PipeFlow	3.10	yes
4	LandXML→PipeNetworks→PipeNetwork→Pipes→Pipe	2.06	yes
5	LandXML→PipeNetworks→PipeNetwork	2.05	yes

TABLE 3: Top retrieved segments for query 'drainage'

Rank	Top Retrieved Segments	λ_g	Relevant?
1	LandXML→PipeNetworks	9.08	yes
2	PipeNetworks→PipeNetwork	8.93	yes
3	PipeNetwork→Pipes	5.44	yes
4	LandXML→Roadways	5.13	yes
5	Pipes→Pipe	5.10	yes

Table 4 shows the comparison results between the proposed semantics-based model and the baseline model when the weights ω_n and ω_c are both set to 0.5. As shown, the incorporation of semantic features significantly enhances the performance of the system whereby the MAP value improves from 58.43% to 86.75%. For the baseline model, the precision dramatically drops when the recall increases from 10% to 50%. It means that the true segments occur at low positions in the ranked list. In contrast, using the semantic model, the precision just slightly decreases. This implies that most the true segments occur at the top positions of the ranked list. The semantic algorithm utilizes related terms for entity retrieval; it, therefore, allows the system to capture relevant entities with names different from the input keyword. For example, for the query `drainage', the baseline model fails to capture relevant entities since the source LandXML schema does contain any classes or attributes with the name `drainage'. Whereas the semantic algorithm is able to obtain related entities such as `PipeNetworks' and `Ditch' (see Table 2).

The system performance variation was analyzed by changing the weights of the two matching components. Three different combinations of weights were created and tested. The results of this experiment are illustrated in Table 5. As shown, the system performance largely depends on the weight setting. The MAP score was found to significantly vary from just nearly 70% to over 86%. The results also show that there is a notable difference in the level of importance between the matching factors. A significant increase from 0 to 50% towards the context matching weight ω_c leads to only a slight improvement of 3% in the overall system performance. When concept name matching weight becomes zero, the system performance noticeably falls to just below 70%. These observations indicate that both of the two matching factors are important for ensuring the quality of the retrieved results. However, it is evident that the major contribution of the semantic algorithm's outperformance over the keyword-base search is from the concept name matching. In other words, the expansion of the user keyword to consider other related terms during the retrieval process plays a significant role in the system enhancement. The best performance in the experiment was obtained from the case where their weights were both set to 0.5. This finding indicates that the name and context have equal importance in contributing to the semantics of a schema component.

TABLE 4: Effect of semantic search on performance. Precisions (%) are calculated for different recall levels. The semantic model performance is according with the weights w_n and w_c are both set to 0.5.

Model	R@10%	R@30%	R@50%	MAP (%)
keyword-base search	86.23	65.94	47.63	58.43
Semantic search	100.00	93.22	93.54	**86.75**

TABLE 5: Effect of weight setting on the system performance. Precisions (%) are calculated for different recall levels

Weight setting	R@10%	R@30%	R@50%	MAP (%)
ω_n=1.0; ω_c=0.0	100.00	89.59	82.13	83.63
ω_n =0.5; ω_c =0.5	100.00	93.22	93.54	**86.75**
ω_n =0.0; ω_c =1.0	85.71	78.78	73.38	68.32

9. RESEARCH CONTRIBUTIONS AND IMPLICATIONS

The main contribution of this study is the development of an effective method for automated generation of model views from an XML civil engineering data standard. The method allows for building a semantic IR system that can analyze the user's data interest from their single input keyword and return a corresponding network portion of the source schema. Previous studies focused on developing rule-based methods for automatically validating the syntax completeness of model views. Because of the lack of an effective method, the construction of a semantically correct view still relies on MVD developers. This research has overcome this barrier by providing a methodology to support users in searching for entities relevant to the context of a domain. Given a keyword of a particular context, the system is able to find semantically related data entities.

The system developed in this study is expected to offer an enabling tool that can support many different stages of the MVD development. First, the proposed method can help MVD developers quickly identify data entities and properties of an open data standard. The process of mapping the data exchange requirement concepts of an IDM to those entities and properties of a data standard is a primary task of MVD developers. Due to the linguistics issues such as synonymy and polysemy, the mapping process solely relies on MVD developers who need to browse the entire schema to find a matching. With the support from this proposed method, a ranked list of the most related entities generated by the system enables developers to work on a short list rather than to manually scroll and examine the entire large and complex set of entities. Rather than conducting a costly and time-consuming process of IDM development, the end user can simply use a query to search for subschemas. For example, using the keyword `drainage system' the system will suggest a ranked list of the most related data in that context such as ditch, pipe network, inlet, outlet, etc. without a need to interview domain professionals. With a list of the most semantically related items, the focus is on only a limited number of items; thus, less effort is required to generate MVDs. In addition, less restriction is required for the end user to choose a keyword for searching relevant entities. The knowledge base utilized in this system is an extensive resource that covers a large number of domain terms. Users with little background in the domain are still able to extract a subschema without deep understanding of the source schema.

10. CONCLUSIONS

Model view definition has been widely recognized as a means for facilitating seamless information exchange throughout the project life-cycle. Although many MVDs have been developed, they are still limited considering the large and various demands in the construction industry. The current time-consuming and error-prone practices of MVD development may result in incomplete MVDs if the developer does not have a high level of domain knowledge. The contribution of this study is an automated method for generating MVDs using the user's keywords. The designed algorithm leverages a domain data dictionary to interpret the user's intention. It utilizes a context-aware approach to match the interpreted concepts to those entities in the source XML data schema. The algorithm takes into account the variation in the name of concepts; thus, it can reduce mismatches due to the inconsistent use of terminology between the user and the data standard.

The proposed method was tested on a gold standard of seven subnetworks manually extracted from the LandXML schema for different input keywords. The result shows that the algorithm can serve as an effective tool for extracting subsets of data schema when the mean average precision approaches 90%. To enhance the system performance and applicability, future studies should be conducted to address the following limitations. Incorporating the use of other large-scale generic lexicons such as WordNet in addition to the CeLe lexicon. Since the coverage of lexicons decides how well the system understand the meaning of keywords, the use of WordNet will help to improve the semantic matching. Additionally, the current study is limited to a single keyword per query. It is necessary to include a new technique that can analyze complex queries such operators as `or' and `and' to merge views. Also, the method is broad and can be applied not only to the civil infrastructure sector. In order for the system to be implemented on the data standards of other domains such as IFC, their special structures should be considered in the traversal algorithm.

As using keywords is one preferable method for information search, the algorithm is expected to become a fundamental tool assisting professionals in extracting data from complex digital datasets. Once keywords can be used for quickly extracting required data for a particular context, the manual processing of developing IDMs and MVD can be avoided. As a result, the transition from paper-based information sharing to fully digital project delivery can be accelerated. Moreover, the technique presented in this article offers a foundational platform for future studies on transforming the way the end user interacts with CIM models. As keywords are a basic unit of human language, the capacity of understanding of this basic semantic unit allows computers to interpret the end user's needs in a more complex input. The method allows computer systems to understand the intention of users through their keywords, it opens a gate to a new research area that focuses on enabling professionals to communicate with digital models using natural language instead of comprehending a computer query language.

REFERENCES

Arampatzis, A., and Kamps, J. (2008). "A study of query length." Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval, ACM, 811–812.

buildingSMART. (2016). "Model view definition summary, <http://www.buildingsmart-tech.org/specifications/ifc-view-definition>. Accessed: April 12, 2016.

buildingSMART. (2017). "IFC Alignment, <http://www.buildingsmart-tech.org/infrastructure/projects/alignment>. Accessed: April 1, 2017.

East, E. W. (2007). "Construction operations building information exchange (cobie)." Report no., DTIC Document.

East, E. W., Nisbet, N., and Liebich, T. (2012). "Facility management handover model view." Journal of Computing in Civil Engineering, 27(1), 61–67.

Eastman, C. (2012). "The future of IFC: Rationale and design of a SEM IFC layer. Presentation at the IDDS workshop

Froese, T. (2003). "Future directions for IFC-based interoperability." ITcon, 8, 231–246.

Gale, W. A., and Church, K. W. (1993). "A program for aligning sentences in bilingual corpora." Computational Linguistics, 19(1), 75–102.

Hu, H. (2014). "Development of interoperable data protocol for integrated bridge project delivery." Ph.d., Ph.d. UMI Dissertations.

inframodel.fi. (2017). "Inframodel, <http://www.inframodel.fi/en/>. Accessed: April 1, 2017.

Jiang, Y., Yu, N., Ming, J., Lee, S., DeGraw, J., Yen, J., Messner, J., and Wu, D. (2015). "Automatic building information model query generation." Journal of Information Technology in Construction.

landxml.org. (2017). "About landxml.org, <http://www.landxml.org/About.aspx>. Accessed: April 1, 2017.

Le, T., and Jeong, H. D. (2017). "NLP-Based Approach to Semantic Classification of Heterogeneous Transportation Asset Data Terminology." Journal of Computing in Civil Engineering, 31(6), 04017057.

Lee, G. (2009). "Concept-based method for extracting valid subsets from an express schema." Journal of Computing in Civil Engineering, 23(2), 128–135.

Lee, G., Park, Y. H., and Ham, S. (2013). "Extended Process to Product Modeling (xPPM) for integrated and seamless IDM and MVD development." Advanced Engineering Informatics, 27(4), 636–651.

Lee, Y.-C., Eastman, C. M., and Solihin, W. (2016a). "An ontology-based approach for developing data exchange requirements and model views of building information modeling." Advanced Engineering Informatics, 30(3), 354–367.

Lee, Y. C., Eastman, C. M., Solihin, W., and See, R. (2016b). "Modularized rule-based validation of a BIM model pertaining to model views." Automation in Construction, 63, 1–11.

Seidenberg, J., and Rector, A. (2006). "Web ontology segmentation: analysis, classification and use." Proceedings of the 15th international conference on World Wide Web, ACM, 13–22.

Thompson, H. S., Beech, D., Maloney, M., and Mendelsohn, N. (2004). "XML schema part 1: structures second edition." W3C Recommendation.

Tversky, A. (1977). "Features of similarity." Psychological Review, 84(4), 327.

Venugopal, M., Eastman, C. M., Sacks, R., and Teizer, J. (2012). "Semantics of model views for information exchanges using the industry foundation class schema." Advanced Engineering Informatics, 26(2), 411–428.

Yang, D., and Eastman, C. M. (2007). "A Rule-Based Subset Generation Method for Product Data Models." Computer-Aided Civil and Infrastructure Engineering, 22(2), 133–148.

Ziering, E., Harrison, F., and Scarponcini, P. (2007). TransXML: XML schemas for exchange of transportation data (Vol. 576). Transportation Research Board.