The 6th International Conference on Construction Engineering and Project Management (ICCEPM 2015)
Oct. 11 (Sun) ~ 14 (Wed) 2015 • Paradise Hotel Busan • Busan, Korea
www.iccepm2015.org

A linked data system framework for sharing construction defect information

Doyeop Lee[1] and Chansik Park[2]

Abstract: Defect data contains experiential knowledge about specific work conditions. And the number of projects performed by a company is too limited for an individual to experience the various defects under the current complex construction environment. Therefore, in order to manage and prevent a reoccurrence of defects, a proper data feedback mechanism is required. However, most defect data are stored in unstructured ways, resulting in the fundamental problem of data utilization. In this paper, a new framework is proposed by using linked data technologies to improve defect data utilization. The target of this framework is to convert defect data to the ontology-based linked data format for sharing defect data from different data sources. To demonstrate it, some technical solutions are implemented by using real cases. The proposed approach can reduce data search time and improve the accuracy of search results as well. Moreover, the proposed approach can be applied to other domains that need to refer to external sources such as safety, specification, product, and regulation.

Keywords: defect management, linked data, ontology, SPARQL query

I. INTRODUCTION

Construction defect is perceived as one of the primary causes of low project productivity, which consumes unnecessary time, cost, materials and manpower for defect rectifications [1]. In order to manage and prevent a reoccurrence of defects, various researches on identifying defect causations [2], analyzing cost-time impacts [3] have been conducted. These research efforts provide important statistical information for decision-making. However, such information only provides generalized concepts or terminologies at a high level of abstraction. Therefore, practitioners cannot make a proper decision for defect prevention in practical situations by using fragmentary statistical information alone without access to actual defect cases that include complex information of specific design/construction tasks.

In order to address this issue, we need to look into researches emphasizing data feedback for preventing a defect. Palaneeswaran et al. [4] argue that the analysis of defect causation should be targeted on developing a suitable knowledge feedback mechanism. Chong and Low [5] point out that many defects continue to occur repeatedly as designers fail to obtain important feedback about defects and suggest developing a database system by using existing knowledge. However, there has been little research on the data structure required for implementing actual data feedback systems. Moreover, when looking into the data structures and search mechanisms currently used for managing defect data, traditional database systems and keyword-based searches appear to be insufficient to improve information discovery and retrieval.

Therefore, an appropriate information platform should be considered to enable users to easily search and directly access the individual defect cases. To achieve this object, functions of information technologies need to be adopted.

In information systems, ontologies are considered as ideal formal tools to represent domain knowledge, as they allow modeling concepts with semantic relationships. In the context of the Semantic Web, ontologies play an important role for publishing and connecting structured data on the Web known as Linked Data.

In this context, we propose the defect linked data framework for searching and sharing defect data. To develop this framework, various technical issues are addressed in this paper. 1) Development of a defect ontology for publishing defect data following linked data principles; 2) Conversion of defect data to the RDF data model; 3) Search of relevant defect cases by using SPARQL queries

Ⅱ. ISSUES OF UTILIZING DEFECT INFORMATION

A. Lack of formal structure in defect data representations

The number of defects that has occurred or identified in a project typically goes in the thousands, including minor and major defects. However, most data are scattered across various company systems and recorded in unstructured formats. Unstructured data is data that does not follow a machine-readable format, so, there is no reliable way to access, analyze, or search desired information. In addition, it is difficult to generate valuable knowledge through vast amounts of existing data as well.

Such defect data is commonly modified and published as a defect casebook or stored as individual data files for sharing knowledge. However, most of them exist as text-based unstructured data, such as PDF or Word documents, spreadsheets, HTML pages, and so on. Even if defect cases are stored using a machine-readable format, it cannot be easily accessed and analyzed without an appropriate data collection structure for describing situations of each defect case in the same way. Therefore, in order to retrieve and analyze defect data effectively, a comprehensive defect data collection template is needed and should include various types of content for various purposes [6].

[1] Ph.D.candidate, Department of Architectural engineering, Chung-Ang University, doyeop@cau.ac.kr
[2] Professor, Department of Architectural engineering, Chung-Ang University, cpark@cau.ac.kr(*Corresponding Author)

B. Only text-based search available

The ways to find the relevant information from existing defect data are quite limited. If you want to find some information in the defect case book, you should take a look at the whole contents because it is normally categorized as construction work method, space, defect type, etc. For instance, in the chapter classified as defect type such as water-leaking, various construction work methods related to water-leaking are mixed in the same chapter. Also, if the data is stored as a HTML or document file, you can access the data by the keyword searching based on text which is normally described in the case title and description. But not all the context information is included in the handbook. In addition, the text is written manually and subjectively without standardized vocabulary. Although keyword-based searching is the most common searching method, it is troublesome. If the keyword is too specific, the search results exclude documents that would have been of interest. On the contrary, if the keyword is too generic, the search results include too many irrelevant documents. Even if the data is stored in a well-classified and structured format, various keywords should be input to get the desired information.

C. Insufficient exchange of defect data

No doubt there are a lot of uncertainties compared with typical and repetitive projects of the past. And the number of projects performed by a company is too limited for an individual to experience the various defects that can occur due to various causes. Even commonly known defects, which are encountered in previous projects within a company, are hard to be shared across ongoing projects.

In general, defects are identified at various stages of the project lifecycle through activities like design review, constructability check, quality inspection, maintenance, etc. Also, these tasks are dealt with by different participants such as architect, contractor, sub-contractor, facility manager, etc. And the defect data are stored at each organization's repository. For this reason, if it is possible to share defect data occurring under different conditions, construction methods, regions and weather types, it might also be possible to integrate defect data that is scattered across different sources or data silos.

Ⅲ. ONTOLOGY AND LINKED DATA IN CONSTRUCTION AREA

Ontologies represent knowledge in specific domains and enable semantic interoperability by linking to other external data sources. They have the potential to improve the limitations toward earlier in sharing and reusing unstructured construction information. Thus, many different concepts and approaches have been tried and implemented, such as representation of construction domain knowledge, extraction or conversion of semantic data from BIM, connection to external data sources, etc.

The e-COGNOS is a research project aiming to develop construction domain ontologies. It provides a portal documentation and supports for consistent knowledge representation [7]. Some works focused on developing specific ontologies at the task level. Niu and Issa [8] have proposed the claim ontology to share the comprehensive and formal claim knowledge with all the participants within the claim work flow.

Making an agreement and commitment to use the same terms in the same way for a domain of interest is one of the ontology principles. The definition of ontology is "a formal, explicit specification of a shared conceptualization" [9]. In the specification and conceptualization stage, concepts are organized into a superclass-subclass hierarchical structure, which is also known as a tree structure. The backbone of ontology is often a hierarchical tree structure. A variety of taxonomies have been developed in the construction sector following a tree structure as well, ranging from domain dictionaries to specialized taxonomies such as bcXML, IFC, Omniclass etc. Ontologies are formalized in a computer-readable format that allows query and reasoning processes. For example, construction safety specification documents are used to extract concepts and relationships from a specification by using a text analysis tool, such as Text2Onto [10]. The research efforts on combining linked data with BIM technologies are reviewed. Corry et al. [11] have conceptually illustrated a need to link building objects in BIM environments to external data sources such as CO^2 level, temperature and humidity for thermal comfort assessment.

While ontology researches are at an early stage, this technology is being applied in various construction management areas and the research efforts provide insights in the representation of domain knowledge or existing data semantically for data interoperability, reasoning and querying. However, literature reviews showed that no study attempts to utilize defect data as linked data, which is to be addressed in this paper.

Ⅳ. LINKED DATA FRAMEWORK

A. Defect Linked Data Framework

The overall process of proposed defect linked data framework is illustrated in Fig 1. First, developing data structure is the underlying process for connecting other processes. To do this, defect ontology has been proposed based on our previous research [6], and it has been developed using the Protégé 4.3 ontology authoring tool.

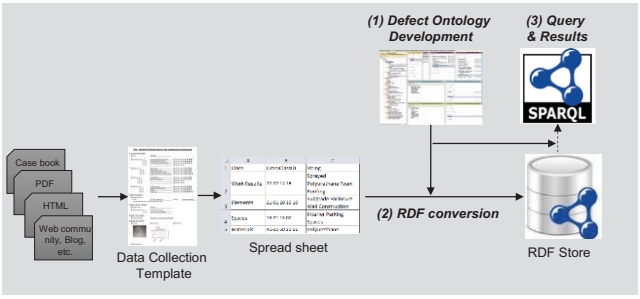

Fig 1. Defect linked data system framework

Second part is converting defect data to structured format as RDF. For establishing the linked data for data searching and sharing on the Web, the context information extracted from existing defect cases has been input in a spreadsheet manually and then converted to an RDF format by using an RDF converter. Third, the data search and share have been tested through the SPARQL query functionality in the ontology editing tool.

B. Ontology development

The defect ontology is illustrated in Fig. 2. The defect ontology consists of seven classes, six object properties and seven data properties, which represent the main content of the defect data collection template which is well described in [16].

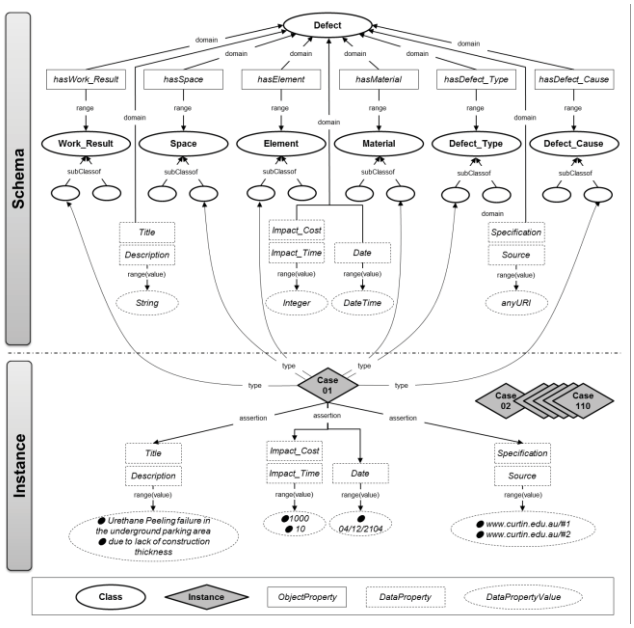

Fig 2. Defect ontology schema

The classes *Defect, Work_Result, Space, Element,* and *Material* mean the context information. Each of these classes has several subclasses, thereby following the Omniclass hierarchy. For instance, *Concrete* and *Thermal_Moisture_Protection* are subclasses of *Work_Result.* The *Defect_Type* clsss contains the subclasses *Crack, Peeling, Water_leaking,* etc. The *Defect_Cause* class consists of three levels of subclasses. The first represents the moment when an initial cause was generated such as *Design* and *Construction*. The second represents the responsibility, such as *Architect, Manager* and *Worker*. The third describes the actual cause, such as *Design_omission*, *Work_interference*, etc. The object properties of each class are defined using the same method. For instance, the *hasWork_Result* is an object property and the domain and range are restricted to the class *Defect, Work_Result* respectively. Data properties have the same domain but different *DataPropertyValues*, depending on the data type, such as text, number and URI. For example, the *Title* and *Description* data properties as refer to a string, *Impact_Cost* refers to an *Integer*, *Date* refers to a *dateTime* value, and *Specification* and *DataSource* refer to an *anyURI* value for linking actual defect documents or web pages.

C. RDF Conversion

An open source library called dotNetRDF is utilized to handle the conversion process. The library is developed based on the .NET platform, and helps handling RDF, SPARQL, and the Semantic Web. The customized conversion tool utilizing this library is developed and written in C# language. To use the tool, the user needs to specify the path of the target spreadsheet. The tool will then handle the relationship generation among different data items according to the defect ontology.

A file in RDF/XML format will then be output illustrated in Fig. 3, which has been converted from a spreadsheet by the developed tool. All this information is defined using the classes in the defect ontology. Other type information, data properties can be explicitly expressed in the RDF file. All these data properties are formed as triples in the file and are easy to retrieve in response of various data queries and further analysis.

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE rdf:RDF [ <!ENTITY rdf 'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
    <!ENTITY rdfs 'http://www.w3.org/2000/01/rdf-schema#'>
    <!ENTITY xsd 'http://www.w3.org/2001/XMLSchema#'>
    <!ENTITY dm 'http://example.org/'>
]>
<rdf:RDF xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:dm="http://example.org/"                    # Declare URL of
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">   defect ontology schema
 <rdf:Description rdf:about="&dm;Case1">
  <rdf:type rdf:resource="&dm;Cold_Fluid-Applied_Waterproofing" />   # Work_Result
  <rdf:type rdf:resource="&dm;Structural_Slabs-on-Grade" />          # Element
  <rdf:type rdf:resource="&dm;Interior_Parking_Spaces" />            # Space
  <rdf:type rdf:resource="&dm;Polyurethane" />                      # Material
  <rdf:type rdf:resource="&dm;Lack_of_the_workmanship" />           # Defect_Type
  <rdf:type rdf:resource="&dm;Peeling" />                           # Defect_Cause
  <dm:Date rdf:datatype="&xsd;dateTime">21092014</dm:Date>
  <dm:Description rdf:datatype="&xsd;string">due to lack of construction thickness </dm:Description>
  <dm:Impact_Cost rdf:datatype="&xsd;integer">3000</dm:Impact_Cost>
  <dm:Impact_Time rdf:datatype="&xsd;integer">15</dm:Impact_Time>
  <dm:Title rdf:datatype="&xsd;string">Urethane Peeling failure in the underground parking area</dm:Title>
  <dm:Specification rdf:datatype="&xsd;anyURI"> http://www.curtin.edu.au/#1</dm:Specification>
  <dm:Webpage rdf:datatype="&xsd;anyURI"> http://www.curtin.edu.au/#2</dm:Webpage>
 </rdf:Description>
</rdf:RDF>
```

Fig 3. Defect case conversion to RDF

D. SPARQL Qurey

Query statements have been tested in the Protégé platform, and the results of the queries show the ability to retrieve meaningful information from organized RDF/XML data.

The first example shows how to query similar defect cases compared to certain specific context information. In this query, a variable representing the cases of interest is used and listed as *?Case*. *SELECT* is a SPARQL reserved word for listing the query results of the variable. All the conditions are specified within the *WHERE* section. It is stated that cases of interest should all belong to the classes, *dm:Cold_Fluid_Applied_Waterproofing* subclass of *Work_Result*, *dm:Structural_Slabs_on_Grade* subclass of *Element*, *dm:Polyurethane* subclass of *Material*, and *dm:Interior_Parking_Spaces* subclass of *Space*.

```
SPARQL Query 1
SELECT ?Case
  WHERE {    ?Case rdf:type dm:Polyurethane .
    ?Case rdf:type dm:Interior_Parking_Spaces .
    ?Case rdf:type dm:Structural_Slabs_on_Grade .
    ?Case rdf:type dm:Cold_Fluid-Applied_Waterproofing . }
```

The second example shows how to identify the inherited relationships and the locations of external resources. This query allows the user to recognize what kind of *Work_Result* is applied, even if the user does not exactly know about any existing defects that belong to class or subclass of *dm:Thermal_Moisture_Protection*. The three variables are *?Case*, *?Work_Result*, and *?anyURL* and *SELECT* is a SPARQL reserved word for listing the query results of these three variables. All the conditions that describe the cases of interest are specified within the *WHERE* section. The conditions also include listing all the resource URLs of those cases for users to access the related documents.

```
SPARQL Query 2
SELECT ?Case ?Work_Result ?anyURL
  WHERE {      ?Case rdf:type ?Work_Result .
    ?Case dm:Source ?anyURL .
    ?Work_Result rdfs:subClassOf* dm:Thermal_Moisture_Protection .}
```

The third example allows analyzing defect impact and identifies critical management factors. The following query statements, Query 3, helps users to identifying the defect cases of type *Water_Leaking* and with an impact cost higher than 1500 dollars. The reserved word *FILTER* helps placing further restrictions on data properties.

```
SPARQL Query 3
SELECT ?Case ?Cost
  WHERE {      ?Case rdf:type dm:Water_Leaking.
    ?Case dm:Impact_Cost ?Cost .
      FILTER (?Cost > 1500) }
```

The forth example can be used to retrieve statistical information, such as the frequency of the defect occurrence, in the time period of interest. In this query, the reserved word *COUNT* is used to show the numbers of defect cases which fit the conditions. It allows the users to find the number of defect cases of type *Thermal_Moisture_Protection* that happened in 2014.

```
SPARQL Query 4
SELECT (COUNT(*) AS ?Case)
  WHERE {      ?Case rdf:type dm:Thermal_Moisture_Protection.
    ?Case dm:Date ?Time .
      FILTER (?Time >= "2014-01-01T00:00:00Z"^^xsd:dateTime
      && ?Time < "2015-01-01T00:00:00Z"^^xsd:dateTime) }
```

V. CONCLUSIONS

This paper has addressed only an initial investigation of the applicability of linked data as possible solutions for effective data usability in managing defect data. Three technical solutions are implemented in this research. First, developing data structure was conducted through building defect ontology schema by using the Protégé tool. Second, for converting defect data to structured format, existing defect cases has been input in a spreadsheet manually and then converted to an RDF format by using an RDF converter. Third, the data search and share have been tested through the SPARQL query functionality in the Protégé ontology editing tool.

The proposed framework is expected to be useful because it can provide new approaches of finding the right information when it is needed and sharing the defect information to other parties. Moreover, the proposed approach that enables to link other external data sources can be applied to other tasks that need to refer to external sources describing safety, specification, product, and regulation information.

However, we have found limitations in the usability of the SPARQL query language, which need to be solved in future research. In order to use the SPARQL query language, the user has to be familiar with the defect ontology. Therefore, user friendly query interfaces are required to allow general users to use SPARQL without the need to understand the SPARQL query language or the defect ontology.

ACKNOWLEDGEMENT

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(NRF-2013R1A1A2062181)

REFERENCES

[1] P.E. Love, H. Li, "Quantifying the causes and costs of rework in construction", *Construction Management & Economics,* vol. 18, no. 4, pp. 479-490, 2000.

[2] P.E. Love, J. Smith, "Benchmarking, benchaction, and benchlearning : rework mitigation in projects", *Journal of Management in Engineering*, vol. 19, no. 4, pp. 147-159, 2003.

[3] P.E.D. Love, Z. Irani, "A project management quality cost information system for the construction industry", *Information Management*, vol. 40, no. 7, pp. 649-661, 2003.

[4] E. Palaneeswaran, "Reducing rework to enhance project performance levels", Proceedings of the one day seminar on recent developments in project management in Hong Kong, 2006.

[5] W.-K. Chong, S.-P. Low, "Latent building defects: causes and design strategies to prevent them", *Journal of Performance of Constructed Facilities,* vol. 20, no. 3, pp. 213-221, 2006.

[6] C.-S. Park, D.-Y. Lee, O.-S. Kwon, X. Wang, "A framework for proactive construction defect management using BIM, augmented reality and ontology-based data collection template", *Automation in Construction*, vol. 33, pp. 61-71, 2013.

[7] T. El-Diraby, C. Lima, B. Feis, "Domain taxonomy for construction concepts: toward a formal ontology for construction knowledge", *Journal of Computing in Civil Engineering,* vol. 19, no. 4, pp. 394-406, 2005.

[8] J. Niu, R. Issa, "Framework for production of ontology-based construction claim documents", ASCE International Conference on Computing in Civil Engineering, Florida, pp. 9-16, 2012

[9] T.R. Gruber, "A translation approach to portable ontology specifications", *Knowledge acquisition,* vol. 5, no. 2, pp. 199-220, 1993.

[10] H.-H. Wang, F. Boukamp, "A context ontology development process for construction safety", Joint CIB Conf.: W102 Information and Knowledge Management in Building and W096 Architectural Management, Netherlands, pp. 297-308, 2008.

[11] E. Corry, D. Coakley, J. O'Donnell, P. Pauwels, M. Keane, "The role of linked data and the semantic web in building operation", 13th annual International Conference for Enhanced Building Operations (ICEBO), 2013.