9-1-2015

Integrating Distributed Sources of Information for Construction Cost Estimating using Semantic Web and Semantic Web Service technologies

Mehrdad Niknam
Marquette University

Saeed Karshenas
Marquette University, saeed.karshenas@marquette.edu

Accepted version. *Automation in Construction,* Vol. 57 (September 2015): 222-238. DOI. © 2015 Elsevier. Used with permission.

Automation in Construction, Vol. 57 (September 2015): 222-238. DOI. This article is © Elsevier and permission has been granted for this version to appear in e-Publications@Marquette. Elsevier does not grant permission for this article to be further copied/distributed or hosted elsewhere without the express permission from Elsevier.

Integrating distributed sources of information for construction cost estimating using Semantic Web and Semantic Web Service technologies

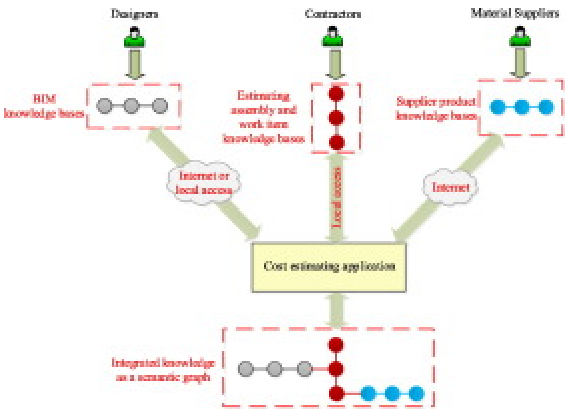
Mehrdad Niknam
Department of Civil, Construction, and Environmental Engineering, Marquette University, Milwaukee, WI
Saeed Karshenas
Department of Civil, Construction, and Environmental Engineering, Marquette University, Milwaukee, WI

Abstract

A construction project requires collaboration of several organizations such as owner, designer, contractor, and material supplier organizations. These organizations need to exchange information to enhance their teamwork. Understanding the information received from other organizations requires specialized human resources. Construction cost estimating is one of the processes that requires information from several sources including a building information model (BIM) created by designers, estimating assembly and work item information maintained by contractors, and construction material cost data provided by material suppliers. Currently, it is not easy to integrate the information necessary for cost estimating over the Internet.

This paper discusses a new approach to construction cost estimating that uses Semantic Web technology. Semantic Web technology provides an infrastructure and a data modeling format that enables accessing, combining, and sharing information over the Internet in a machine processable format. The estimating approach presented in this paper relies on BIM, estimating knowledge, and construction material cost data expressed in a web ontology language. The approach presented in this paper makes the various sources of estimating data accessible as Simple Protocol and Resource Description Framework Query Language (SPARQL) endpoints or Semantic Web Services. We present an estimating application that integrates distributed information provided by project designers, contractors, and material suppliers for preparing cost estimates. The purpose of this paper is not to fully automate the estimating process but to streamline it by reducing human involvement in repetitive cost estimating activities.

Graphical abstract


Designers
Contractors
Material Suppliers
BIM knowledge bases
Estimating assembly and work item knowledge bases
Supplier product knowledge bases
Internet or local access
Local access
Internet
Cost estimating application
Integrated knowledge as a semantic graph

Keywords

Cost estimate, Construction project, BIM, Construction resource, Semantic Web, Knowledge-based system, Semantic Web Service, Ontology, RDF, OWL

1. Introduction

Various qualitative and quantitative methods for construction and manufacturing cost estimation in terms of their requirements, methodology, limitations, and strengths have been investigated by Aram et al.[1] The scope of our paper is limited to quantitative cost estimates prepared by contractors based on a complete set of design documents. With the advent of building information modeling technologies, estimators can digitally extract building element properties from a building information model (BIM) and transfer the data to an estimating application. Current commercial estimating applications extract BIM data using proprietary add-in programs. An add-in is a custom-made program that is designed for retrieving data from another application. For example, WinEst[2] software has an add-in program for transferring BIM element properties from an Autodesk Revit model[3] to a WinEst estimate. Mapping BIM data to estimating software assemblies requires human involvement and is usually a time consuming process.

Another time consuming activity in cost estimating is the process of updating the estimating application's material resource cost databases. Current estimating applications keep built-in databases of resource unit costs. Since resource costs are affected by economic conditions and continuously change based on supply and demand, estimating applications' unit-cost databases must be updated before starting a new estimate. Currently, material supplier data are quoted in formats that are suitable for human consumption and are not processable by computer applications. Therefore, estimating application databases can only be updated manually.

The process of understanding information that is created in other sources is human-intensive and requires employment of specialized human resources. Presenting the required information for cost estimating in a computer processable format can greatly improve estimator's efficiency.[4] Semantic Web[5]technology provides an infrastructure and a data modeling format that enables sharing and combining information over the Internet in a machine processable format. Semantic Web uses formal ontologies[6] to describe the organization of data distributed over the Internet. Ontologies are explicit formal specifications of the concepts in a domain and relations among them.[7] Ontologies can be shared with computer applications to enable processing data that are generated in other sources. A knowledge base is an information repository created based on ontologies that provides means for collecting, organizing, and sharing information. Semantic Web is intended to build distributed knowledge-based systems.[8]

In this paper, we present a semantics-based estimating application that combines information from a building information model (BIM) knowledge base, an estimating assembly knowledge base, and material suppliers' Semantic Web Services to prepare a cost estimate. We investigate how such knowledge bases and Semantic Web Services can be developed, and how a semantics-based estimating application can access these distributed sources of data over the Internet for cost estimating. We investigate how a semantics-based estimating application can reduce human involvement in estimating activities for: 1) mapping BIM element data to estimating assemblies, and 2) updating estimating material resource cost databases.

2. Current construction cost estimating applications

To estimate the cost of a work item in a project, one needs to know the work item quantity and the unit costs for the resources necessary for its construction. Current estimating applications keep built-in databases of assemblies, work items, crew make-ups, crew productivities, and resource (material, equipment, and labor) unit costs. A cost estimating application provides the infrastructure to digitally map an element in a BIM to an estimating assembly which is a predefined group of work items, as shown in Fig. 1.

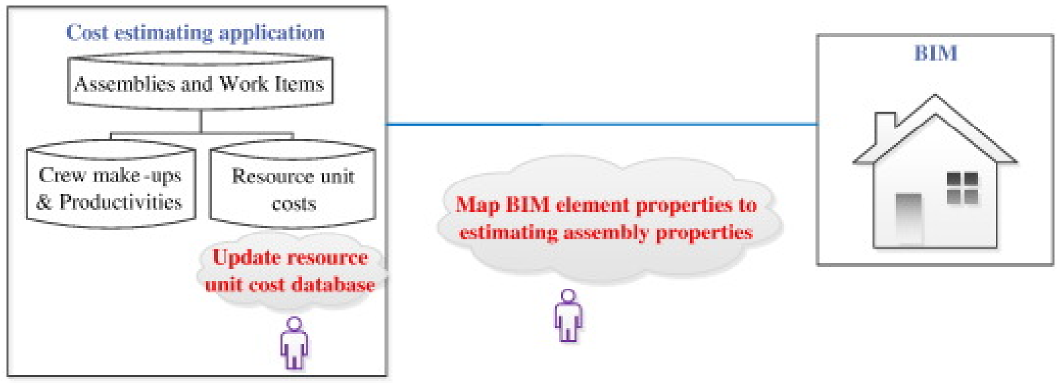

Cost estimating application
Assemblies and Work Items
Crew make-ups & Productivities
Resource unit costs
Update resource unit cost database
Map BIM element properties to estimating assembly properties
BIM

Fig. 1. Current estimating application.

Current computer cost estimating applications require a number of time consuming, repetitive, and error-prone steps including:

1. An add-in program that facilitates digital transfer of BIM element properties to the estimating application. An add-in program requires human intervention for mapping BIM element properties to their corresponding estimating assembly properties which is a time consuming process.
2. Resource unit costs continuously change; therefore, an estimating application's resource database must be updated before a new estimate. This is a time consuming process that requires estimator

involvement for obtaining the latest unit costs from various material suppliers and updating the estimating software's material resource databases.

The above inefficiencies occur because of the way estimating applications are developed and estimating data are stored. Almost all current computer cost estimating applications are modeled and developed using object-oriented technology and estimating data are stored in relational or object databases. As discussed by the W3C Semantic Web Best Practices and Deployment Working Group,[9] the reusability of an object-oriented domain model is often limited because they are domain specific and only take into consideration abstractions needed to solve a problem within the confines of their own individual problem space. Some of the limitations of an object-oriented approach to domain modeling are:[9]

1. The domain schema is local and cannot be shared among applications or on the web. Therefore, two applications cannot share information without a custom-made add-in program or using a standard data format.
2. Application schemas cannot be dynamically modified. Any changes in an application schema would require revising the software written for the schema. In construction cost estimating, this would require a new add-in program for a new version of a BIM application.
3. Each domain develops a separate class hierarchy for the same set of objects; for example, the same building would be modeled using two different class hierarchies in the design and the estimating domains. This means the same element in the same building would belong to two different classes. However, in object-oriented systems, classes defined in different domains to represent the same object cannot share instance properties.

These limitations have made it difficult to share and combine information among various construction domain applications. In the following sections, Semantic Web technology is introduced and a semantics-based construction cost estimating approach is presented. The new cost estimating approach significantly reduces human involvement in routine, repetitive cost estimating tasks.

3. Semantic Web

The current web infrastructure is a distributed network of web pages that can refer to each other using Uniform Resource Locators (URLs) and is suitable for human consumption. Semantic Web is a network of connected data that are machine accessible and processable.[10] It can be seen as a graph in which each node is an instance that is pointing to other nodes. Therefore, a semantic definition of a construction project enables project participants to represent their information in a graph structure and easily combine and connect their information about the project.[11] Such an infrastructure makes distributed data connected and enables applications to search for and find data distributed on the web. Semantic Web enables creating data models, drawing meaningful conclusions from encoded knowledge, and sharing information on the web and between computer applications.[12] Semantic Web allows sharing model schemas and enables computer applications to process and draw conclusions on data that are generated in other sources.[13]

Semantic Web technology uses ontologies[6] to describe a domain. Ontologies explicitly define the concepts, relationships among the concepts, and the relevant terminology in a domain of interest.[7] Ontologies can be imported and used for knowledge representation. This gives computer applications awareness of the organizations of the data distributed over the Internet. A domain ontology and a set of the domain concept instances constitute a domain knowledge base.[14] What can be expressed using an ontology can be stored and used in a knowledge base; Semantic Web is intended to build distributed knowledge-based systems.[8]

4. A semantics-based cost estimating approach

An estimating application must access and use design, estimating, and resource information across organizational, specialty, and geographic divides. In the Architecture, Engineering and Construction (AEC) domain, design knowledge is supplied by design organizations, estimating knowledge is maintained by construction companies, and material resource cost knowledge is provided by material suppliers. Therefore, a flexible estimating approach must be able to access and use independently created domain knowledge that is accessible over the Internet.

We used Semantic Web[5] technology for sharing and integrating information distributed over the Internet. Fig. 2shows a Semantic Web based estimating approach developed in this study. The new estimating application uses distributed domain knowledge bases that are created by domain experts. For example, designers create BIM knowledge bases, construction estimators create estimating assembly and work item knowledge bases, and material suppliers provide material cost knowledge bases.

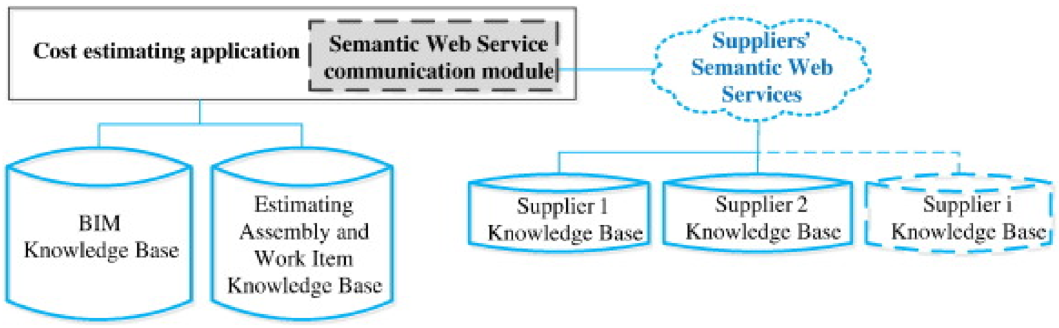

Cost estimating application
Semantic Web Service communication module
Suppliers' Semantic Web Services
BIM Knowledge Base
Estimating Assembly and Work Item Knowledge Base
Supplier 1 Knowledge Base
Supplier 2 Knowledge Base
Supplier i Knowledge Base

Fig. 2. Semantics-based estimating approach developed in this study.

A knowledge base can be developed as a Simple Protocol and Resource Description Framework Query Language (SPARQL) endpoint[15] or a Semantic Web Service.[16] In this study, we created BIM and estimating assembly and work item knowledge bases as SPARQL endpoints and material suppliers' knowledge bases as Semantic Web Services. SPARQL is a semantic query language that is able to retrieve and manipulate data in Semantic Web. A SPARQL endpoint is directly queried by the estimating application whereas accessing a Semantic Web Service requires a communication module. We developed a prototype estimating application that can access these knowledge bases to create an estimate.

Ontologies are needed to define the organization of information used by our semantics-based estimating application. Several methods have been proposed to build ontologies such as Uschold and King,[17] Gruber,[18] METHONTOLOGY,[19] On-To-Knowledge,[20] DILIGENT,[21] and NeOn[22,23] methodologies. In this study, we used the NeOn methodology because of the flexibility it provides for a variety of scenarios instead of prescribing a rigid workflow. NeOn is a scenario based methodology that emphasizes reuse of ontological and non-ontological resources, the reengineering and merging, and taking into account collaboration and dynamism.[24] The NeOn methodology divides the general problem of ontology development into nine sub-problems, also referred to as NeOn scenarios.[22] To obtain a solution to a general problem, solutions to different NeOn scenarios are combined.

The estimating approach shown in Fig. 2 requires an ontology that defines the estimating domain knowledge. Instead of developing a large ontology that covers all estimating domain concepts, we developed a separate ontology for each of the knowledge bases shown in Fig. 2. This means an ontology network that consists of an ontology for the BIM knowledge base, an ontology for the estimating assembly and work item knowledge base, and a set of ontologies for material suppliers' Semantic Web Services.

NeOn scenario 1 requires documentation of the ontology requirements that defines the purpose, scope and implementation language of the ontology. The requirements for the ontologies developed in this study are discussed in Sections 5, 6 and 7 of this paper. We implemented ontologies in Resource Description Framework (RDF)[25] and Web Ontology Language (OWL).[26] RDF is a standard model to represent data in Semantic Web. RDF uses subject-predicate-object triples to represent information in a graph form. It is a standard model for data interchange on the web. RDF facilitates merging data from sources with different underlying schemas and supports the evolution of data schemas over time without requiring changes in the applications consuming data. OWL extends RDF to build ontologies on the web and to enhance Semantic Web with reasoning power of description logic. RDF and OWL are different layers of Semantic Web that work together to facilitate creating ontologies, merging data distributed over the internet, supporting the evolution of data schemas, and reasoning on data from different sources. RDF and OWL allow building and connecting knowledge bases distributed over the Internet. We used the Protégé[27] software to code the ontologies in RDF and OWL. Protégé is an open source java tool providing an extensible architecture for the creation of customized ontologies and knowledge bases.

NeOn scenario 2 requires using all non-ontological resources available when developing a new ontology. Non-ontological resources include all published documents in the domain of the ontology being developed. We used non-ontological resources related to the estimating domain including cost estimating books, and cost estimating references such as RSMeans reference books,[28] UNIFORMAT II classification system,[29] and CSI MasterFormat.[30]

NeOn scenarios 3, 4, and 5 require reusing, reengineering, or merging existing ontological resources, respectively. Scenario 6 recommends both reengineering and merging existing ontological resources, if necessary. We did not need to reengineer or merge any existing ontologies. The existing ontological resources that we used are:

1. QUDT ontology:[31] This ontology expresses quantities and units of measurement. QUDT ontology was developed for the NASA Exploration Initiatives Ontology Models (NExIOM) project. QUDT provides a standardized and consistent vocabulary for the terminology used in science and engineering to represent units of measurements.
2. Free Class OWL ontology (FC):[32] FC is developed by the European Building and Construction Materials Database for describing construction materials and services.
3. Good Relations ontology (GR):[33,34] This ontology was developed to allow businesses to semantically define their product offerings and publish them on the web. GR provides a conceptual model for general concepts such as company, store location, offer, product descriptions, price, payment, shipment, and warranty information.
4. OWL-S ontolog:[16] This ontology is a framework to provide computer-interpretable descriptions of web services and the means by which they are accessed.
5. Organization Ontology.[35] W3C has defined an organization ontology that is designed to enable representation of information on organizations and organizational structures. Organization ontology is intended to provide a generic, reusable core ontology that can be extended or specialized for use in particular situations.

We imported and reused the above mentioned ontologies as will be discussed in different subsequent sections of this paper. Protégé user interface allows importing ontologies available on the web and reusing them when developing a new ontology. We mapped the above mentioned ontologies to the ontologies that we developed when necessary.

NeOn scenario 7 recommends that ontology developers use Ontology Design Patterns (ODPs) as a guide when developing new ontologies. For ontology development, we did not use any ontological design patterns but our ontology development benefitted from guidelines and suggestions in.[10,27,36–39]

NeOn scenario 8 is required when restructuring ontological resources. We did not need to restructure ontological resources used in this study. NeOn scenario 9 is required when localizing ontological resources to other natural languages. We developed our ontologies in English, and did not try to localize it to other natural languages.

5. Building information model (BIM) knowledge base

5.1. Related work

Staub-French et al.[40] developed a formal process to help estimators represent their reasons and logical basis for customizing work items, resources, and productivity rates. Their formal process allows for inferring the impact of building element and element intersection features on cost. Later, Nepal et al.[41] extended and built on the element and intersection features. They used ifcXML (an XML representation of IFC[42]) to transform designer focused BIM into a construction specific feature-based model that included the effect of features such as element shape, element penetration, and locations of penetrations on cost. They reused the relevant predefined IFC properties and defined new attributes that are important from a construction perspective. Their feature-based model can facilitate some of the cost estimating tasks such as choosing an appropriate crew and crew productivity. XML requires a high level of manual human engineering involvement in achieving interoperability. XML is a serialization format to encode information so that when it's passed between machines it can be parsed; the challenge is that any other program that needs to read an XML file would require a special code.[43]

Semantic Web allows computer applications to process and draw conclusions on data that are generated in other sources[13] and enhances interoperability among distributed information sources. Pauwels et al.[44] used a Semantic Web approach to interlink geometry information in two different schemas; their implementation included only geometry and did not include parameters and features needed for cost estimating. Other studies have used ontologies and semantic modeling to (1) infer features that may not be provided in a BIM in early stages of design,[45] (2) infer construction methods[46] and (3) infer work items for each building element.[47]

A few studies have used EXPRESS-to-OWL conversion methods to develop an ifcOWL ontology.[48–51] In this study, we did not use an EXPRESS-to-OWL conversion method to create a BIM ontology from IFC. Our approach to BIM ontology development is discussed in.[52] We developed software that creates a BIM knowledge base directly from an Autodesk Revit model.[3] The knowledge base includes element types and instance properties that are extracted from a Revit model using the Revit API. Our semantics-based estimating application queries element properties from the model knowledge base for quantity takeoff. The following section discusses the organization of the developed BIM knowledge base.

5.2. BIM knowledge base

The cost estimating application developed in this study accesses a semantically defined BIM knowledge base. The purpose of this knowledge base is to provide the estimating application information required to calculate work item quantities. This knowledge base allows machine processing of model element properties necessary for calculating work item quantities. In an earlier work, we presented an ontology-based BIM approach[52] that used a shared ontology to represent BIM element types and their relationships. Different domains of knowledge such as design and estimating domains would create their own ontologies by extending the shared building ontology. For example, the design domain ontology contains specific knowledge related to the design of a building. This includes building element design properties such as geometry and material. In this paper, the approach developed in[52] is used for developing a BIM knowledge base. The shared building ontology is organized according to the UNIFORMAT II classification system;[29] Uniformat II is an ASTM standard that has been revised by Construction Specifications Institute (CSI) and Construction Specifications Canada (CSC).

Fig. 3 shows part of the knowledge base that describes a 1.5 m × 1.5 m × 1.2 m spread footing element. In Fig. 3, "muso" and "mudo" represent the prefixes we used to define URIs (http://www.w3.org/Addressing/) for the shared and design ontologies in the BIM knowledge base. A design firm can use a URI to uniquely identify its projects. We assigned the prefix "abc" to the designer of the example building project used in this study.

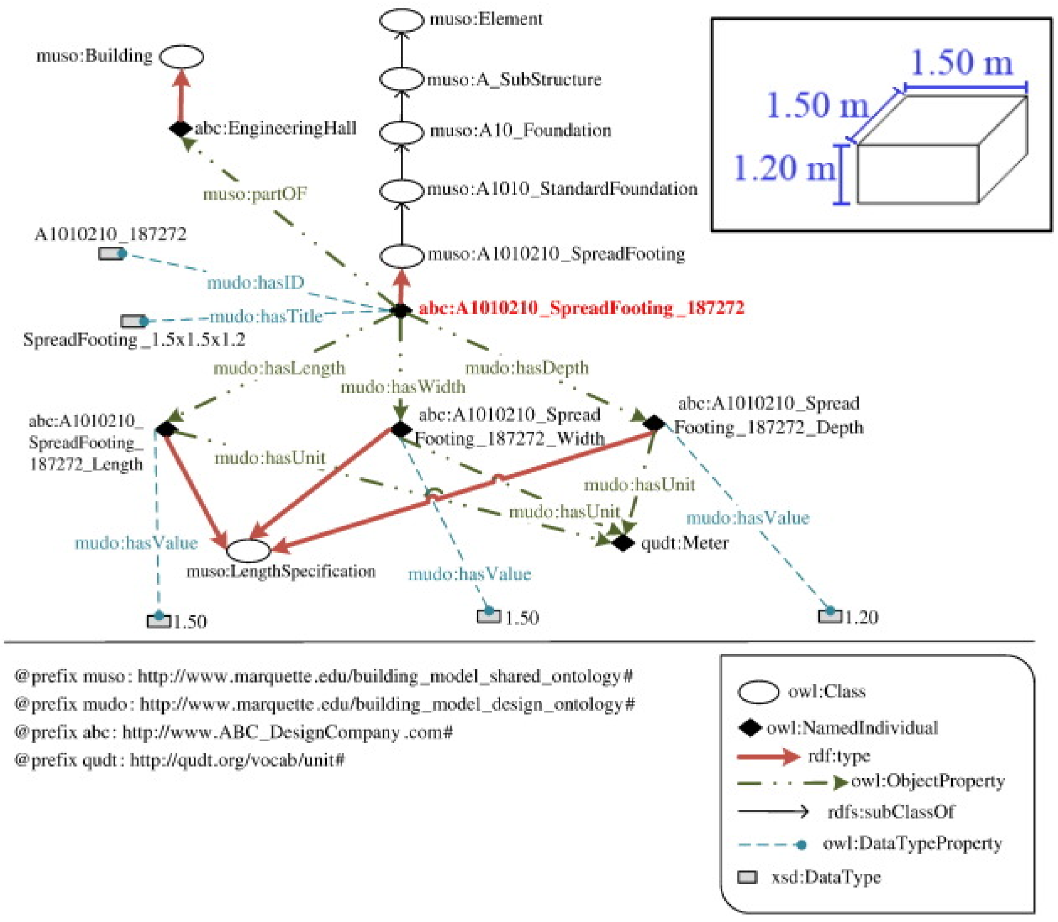

muso:Element
muso:Building
muso:A_SubStructure
abc:EngineeringHall
muso:A10_Foundation
muso:A1010_StandardFoundation
muso:partOF
A1010210_187272
muso:A1010210_SpreadFooting
mudo:hasID
mudo:hasTitle
abc:A1010210_SpreadFooting_187272
SpreadFooting_1.5x1.5x1.2
mudo:hasLength
mudo:hasWidth
mudo:hasDepth
abc:A1010210_SpreadFooting_187272_Length
abc:A1010210_Spread Footing_187272_Width
abc:A1010210_Spread Footing_187272_Depth
mudo:hasUnit
mudo:hasUnit
mudo:hasUnit
mudo:hasValue
mudo:hasValue
mudo:hasValue
muso:LengthSpecification
qudt:Meter
1.50
1.50
1.20
1.50 m
1.50 m
1.20 m
@prefix muso: http://www.marquette.edu/building_model_shared_ontology#
@prefix mudo: http://www.marquette.edu/building_model_design_ontology#
@prefix abc: http://www.ABC_DesignCompany.com#
@prefix qudt: http://qudt.org/vocab/unit#
owl:Class
owl:NamedIndividual
rdf:type
owl:ObjectProperty
rdfs:subClassOf
owl:DataTypeProperty
xsd:DataType

Fig. 3. A spread footing representation in a BIM knowledge base.

Fig. 3 shows ID, title, length, width, and depth properties of a footing instance. The URI assigned to the footing is “abc:A1010210_SpreadFooting_187272”. To define a URI for an element instance, we combined designer's URI, element's UNIFORMAT II ID and type name, and the unique number assigned to the element in the Autodesk Revit[3] model of the building. Some building element properties are multi-valued and require special representations. To properly represent multivalued properties such as length, width, and depth, we have specified a “muso:LengthSpecification” class with “hasUnit” and “hasValue” properties. As a result, the length of the spread footing element in Fig. 3 is “abc:A1010210_SpreadFooting_187272_Length”, which refers to the value “1.50” and to the unit “qudt:Meter”.

All other building element instances are included in the BIM knowledge base in a similar way. In this study, we used the Autodesk Revit API to convert a Revit model file to a semantic knowledge base. We stored the BIM knowledge base in an OpenRDF Sesame triplestore.[53] OpenRDF Sesame provides a SPARQL Endpoint[15] interface for adding, editing, and querying RDF data. Sesame provides an API for local and remote access (over the Internet) to its data.

6. Estimating assembly and work item knowledge base

An estimating assembly represents the field activities (work items) that must be completed in order to create a building element. For example, an assembly for a footing element includes work items such as forming, reinforcing, and placing concrete. Fig. 4 shows the estimating assembly ontology developed in this study. The purpose of the ontology is to provide a semantic model for estimating assemblies. The ontology includes assembly properties such as cost, ID, title, special conditions (e.g., weather condition), and a list of work items that are part of the assembly. The ontology allows adding a list of job conditions that influence work item productivities. Currently, there are no ontological resources available for estimating assemblies. We organized the estimating assembly ontology according to the UNIFORMAT II classification system; therefore, the ID and title of an estimating assembly would be the same as those of its UNIFORMAT II counterpart. An estimating assembly includes one or more work items and the cost of the assembly is the sum of the costs of its respective work items. Fig. 4 also shows a screenshot of the estimating assembly ontology implementation in Protégé.[27]

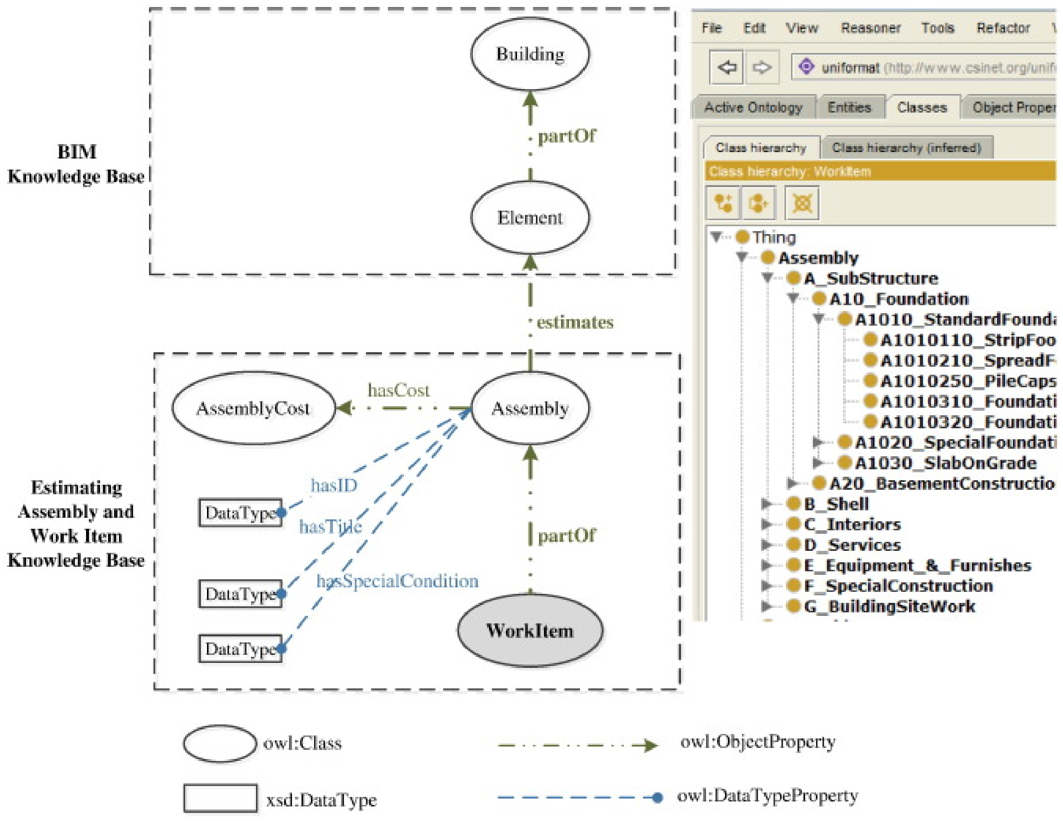


Fig. 4. Estimating assembly ontology.

The purpose of the work item ontology is to provide a semantic model for construction work items. The work item ontology we developed is shown in Fig. 5. The concepts in the work item ontology are extracted from non-ontological resources including estimating books, and estimating references such as RSMeans reference books,[28] and CSI MasterFormat.[30]

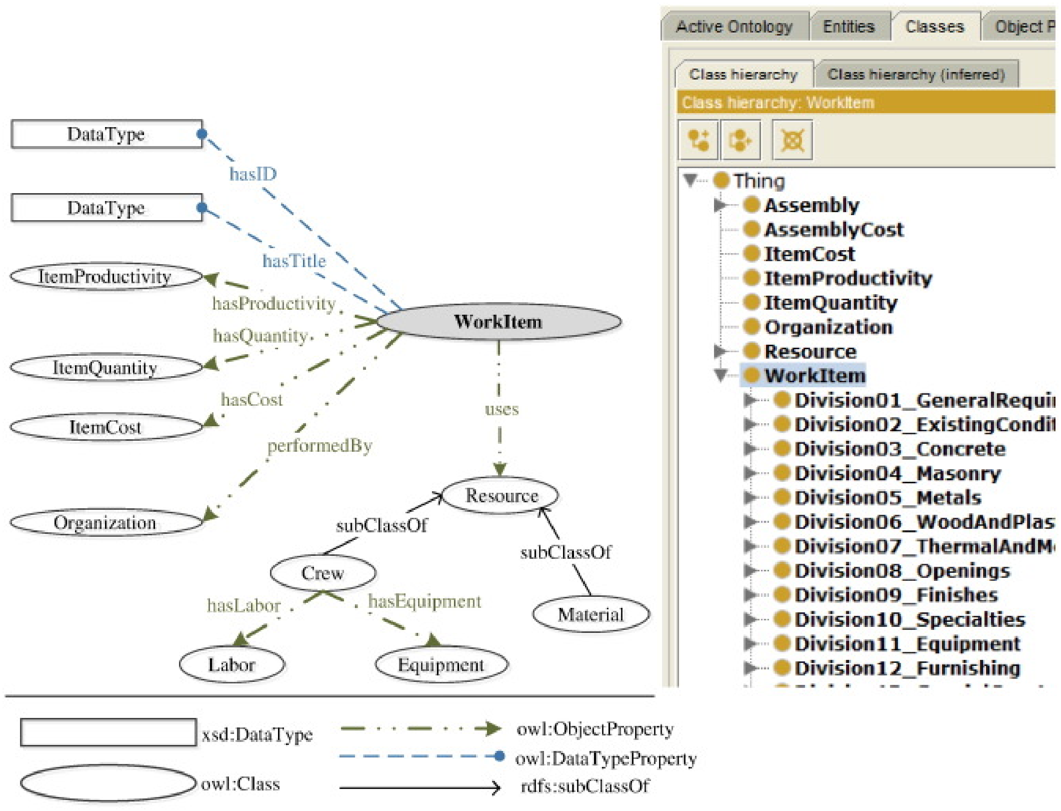


Fig. 5. Work item ontology.

The work item ontology includes the following concepts:

1. ID & Title: Each work item needs an ID and a title for identification. We used CSI MasterFormat[30] work item IDs and titles to identify work items in our ontology.
2. ItemProductivity: This concept specifies the productivity of the crew building the item. Productivity is affected by job conditions and labor and equipment used.
3. ItemQuantity: This concept defines the quantity of work involved in a work item. The information for calculating a work item's quantity is obtained from the BIM knowledge base.
4. Resource: This concept represents the type, quantity, and unit cost of resources used in a work item. It can be divided into the following subclasses:
    1. Crew: This concept represents the type, quantity, and unit cost of any labor or equipment used in a work item.[28] The quantities of labor and equipment are calculated using the work item quantity and crew productivity.
    2. Material: This concept represents the type, quantity and unit cost of the material used in a work item. The quantity of material for a work item is calculated directly from the work item quantity using an appropriate waste factor. In this study, we used the Free Class OWL ontology (FC)32 (Ontology URI: http://www.freeclass.eu/freeclass_v1#) developed by the European Building and Construction Materials Database for describing construction materials and services to represent work item material ontology.
5. ItemCost. This concept represents the estimated cost of a work item.
6. Organization: This concept represents the companies involved in a work item construction such as the contractor performing the work, the inspecting organization, and material suppliers. W3C has defined

an organization ontology (Ontology URI: http://www.w3.org/ns/org#) which we used in the work item ontology in this study. For the latest W3C candidate recommendation for organization ontology refer to W3C.[35]

We coded the above mentioned ontologies in Protégé. The Protégé user interface allows opening ontologies available on the web and reusing them when developing a new ontology. As explained above, Organization and Free Class OWL Ontologies are examples of existing ontologies used in developing the assembly and work item ontologies.

We used the above-mentioned ontologies to develop RDF and OWL knowledge bases for the assemblies and work items used in building construction cost estimating. Fig. 6 shows a section of this knowledge base representing an estimating assembly instance for building a spread footing. In Fig. 6, “mueo” represents the estimating assembly and work item ontologies shown in Figs. 4 and 5. We used prefix “xyz” to identify the company that prepared the cost estimate for the example project used in this study.

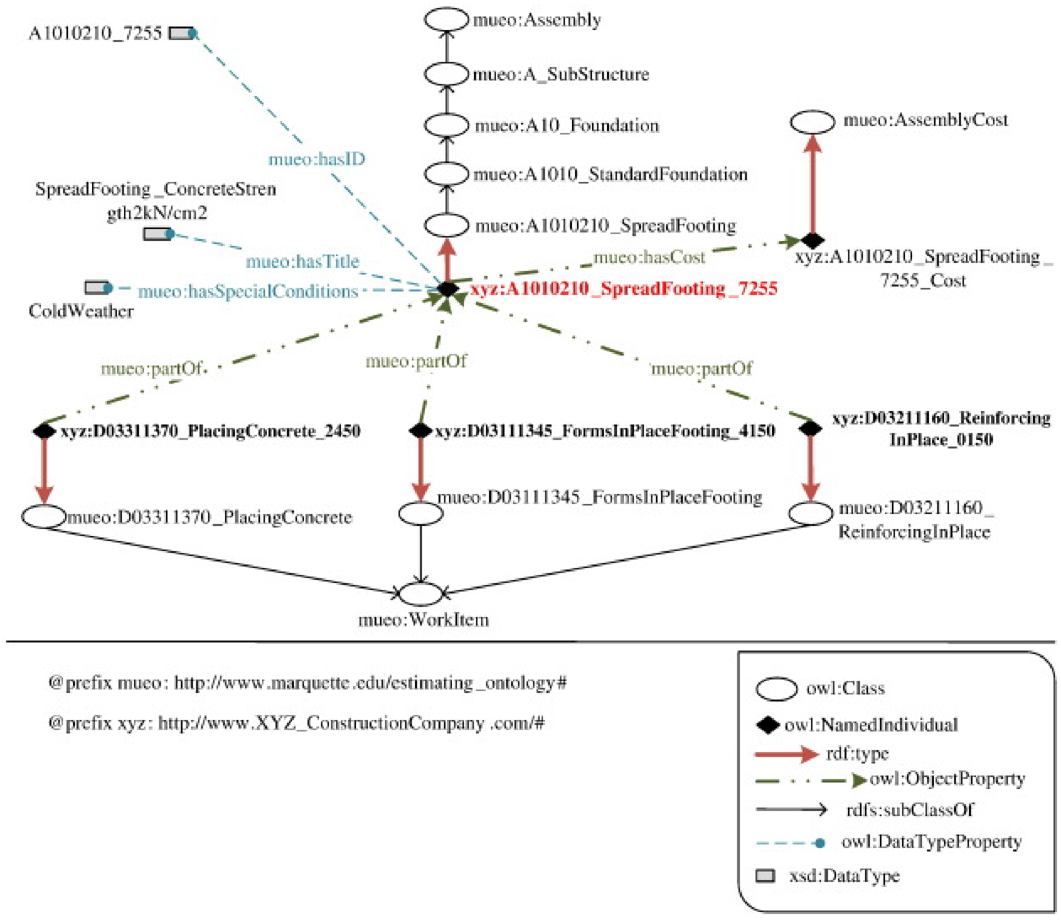

A1010210_7255
mueo:Assembly
mueo:A_SubStructure
mueo:A10_Foundation
mueo:AssemblyCost
mueo:hasID
SpreadFooting_ConcreteStrength2kN/cm2
mueo:A1010_StandardFoundation
mueo:A1010210_SpreadFooting
mueo:hasCost
mueo:hasTitle
xyz:A1010210_SpreadFooting_7255_Cost
mueo:hasSpecialConditions
xyz:A1010210_SpreadFooting_7255
ColdWeather
mueo:partOf
mueo:partOf
mueo:partOf
xyz:D03311370_PlacingConcrete_2450
xyz:D03111345_FormsInPlaceFooting_4150
xyz:D03211160_ReinforcingInPlace_0150
mueo:D03111345_FormsInPlaceFooting
mueo:D03311370_PlacingConcrete
mueo:D03211160_ReinforcingInPlace
mueo:WorkItem
@prefix mueo: http://www.marquette.edu/estimating_ontology#
@prefix xyz: http://www.XYZ_ConstructionCompany.com/#
owl:Class
owl:NamedIndividual
rdf:type
owl:ObjectProperty
rdfs:subClassOf
owl:DataTypeProperty
xsd:DataType

Fig. 6. A spread footing assembly in the estimating knowledge base.

In Fig. 6, “xyz:A1010210_SpreadFooting_7255” is an instance of the “mueo:A1010210_SpreadFooting” class in the estimating ontology. The assembly instance has an ID, a title, a cold weather special condition, cost, and work items “xyz:D03311370_PlacingConcrete_2450”, “xyz:D03111345_FormsInPlaceFooting_4150”, and “xyz:D03211160_ReinforcingInPlace_0150”.

All of the work items that belong to an estimating assembly are represented according to the work item ontology shown in Fig. 5. The semantic representation of "xyz:D03311370_PlacingConcrete_2450" is presented in Fig. 7. In the list of prefixes, "org" is the organization ontology defined by W3C,[35] "fc" represents the Free Class OWL ontology (FC),[32] and "rst" is the URI of the material supplier for the work item. We used multi-valued properties to model quantity, material, crew, productivity, and item cost.

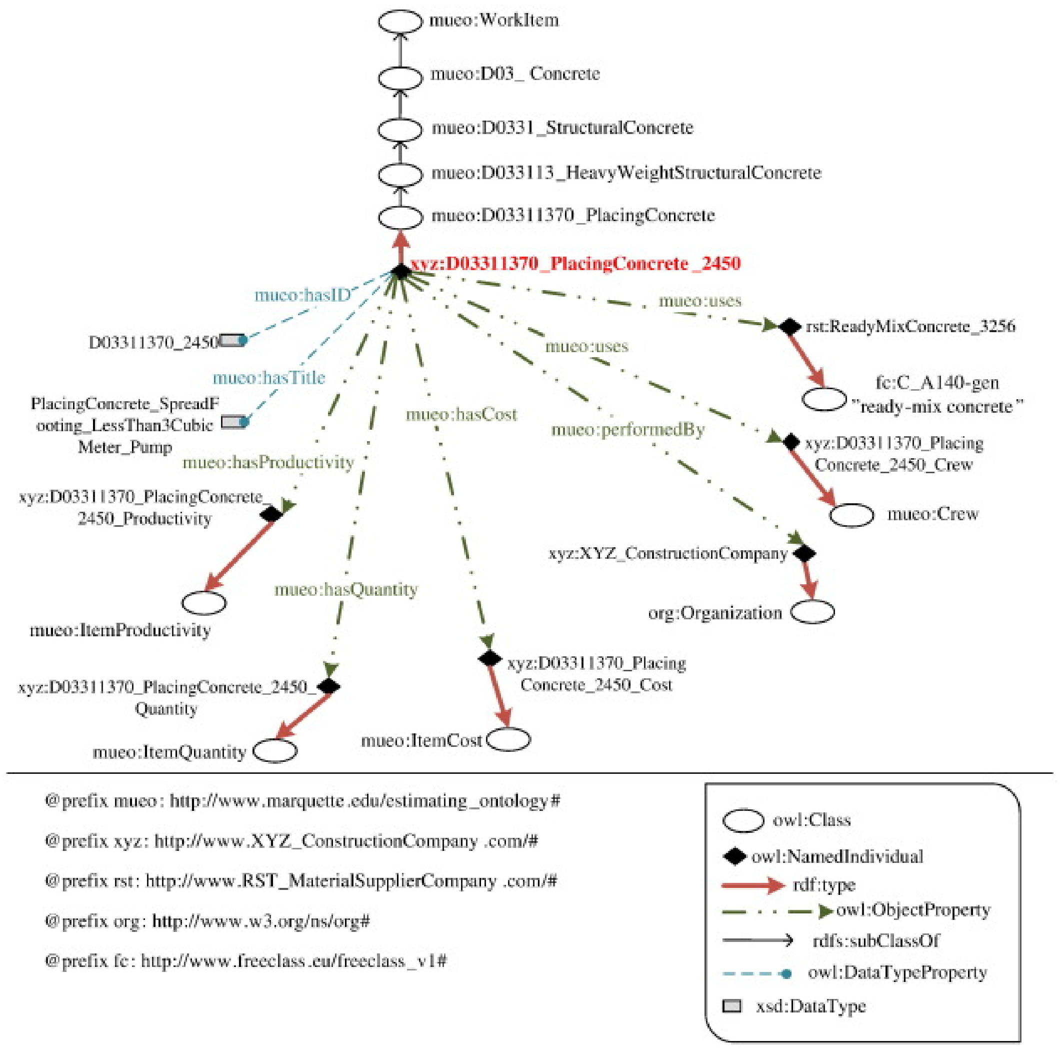

mueo:WorkItem
mueo:D03_ Concrete
mueo:D0331_StructuralConcrete
mueo:D033113_HeavyWeightStructuralConcrete
mueo:D03311370_PlacingConcrete
xyz:D03311370_PlacingConcrete_2450
mueo:hasID
D03311370_2450
mueo:hasTitle
PlacingConcrete_SpreadFooting_LessThan3CubicMeter_Pump
mueo:hasProductivity
xyz:D03311370_PlacingConcrete_2450_Productivity
mueo:ItemProductivity
mueo:hasQuantity
xyz:D03311370_PlacingConcrete_2450_Quantity
mueo:ItemQuantity
mueo:hasCost
xyz:D03311370_PlacingConcrete_2450_Cost
mueo:ItemCost
mueo:uses
rst:ReadyMixConcrete_3256
fc:C_A140-gen "ready-mix concrete"
mueo:uses
xyz:D03311370_PlacingConcrete_2450_Crew
mueo:Crew
mueo:performedBy
xyz:XYZ_ConstructionCompany
org:Organization
@prefix mueo: http://www.marquette.edu/estimating_ontology#
@prefix xyz: http://www.XYZ_ConstructionCompany .com/#
@prefix rst: http://www.RST_MaterialSupplierCompany .com/#
@prefix org: http://www.w3.org/ns/org#
@prefix fc: http://www.freeclass.eu/freeclass_v1#
owl:Class
owl:NamedIndividual
rdf:type
owl:ObjectProperty
rdfs:subClassOf
owl:DataTypeProperty
xsd:DataType

Fig. 7. Placing concrete work item in the estimating knowledge base.

Fig. 8 shows the semantic representation of the quantity property of a work item. We developed a set of SWRL[54] rules that calculate work item quantities from the BIM element properties. SWRL is a language that expresses logic rules which are of the form of an implication between an antecedent (body) and consequent (head): whenever the conditions specified in the antecedent hold (meaning, they are true), then the conditions specified in the consequent must also hold. This would eliminate the need for manual mapping of BIM element properties to estimating assembly properties as is practiced in the current estimating applications.

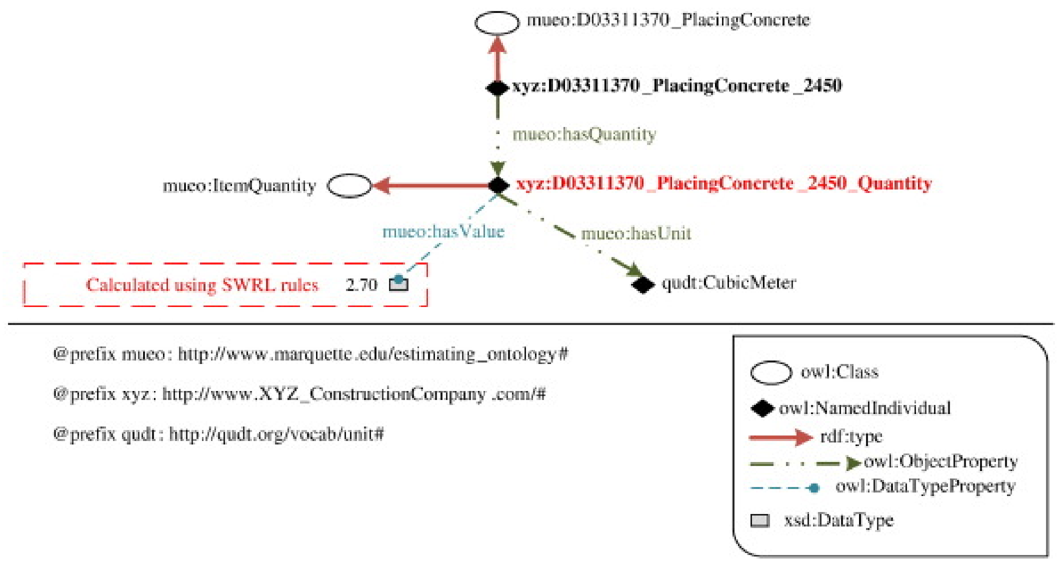

mueo:D03311370_PlacingConcrete
xyz:D03311370_PlacingConcrete_2450
mueo:hasQuantity
mueo:ItemQuantity
xyz:D03311370_PlacingConcrete_2450_Quantity
mueo:hasValue
mueo:hasUnit
Calculated using SWRL rules
2.70
qudt:CubicMeter
@prefix mueo: http://www.marquette.edu/estimating_ontology#
@prefix xyz: http://www.XYZ_ConstructionCompany.com/#
@prefix qudt: http://qudt.org/vocab/unit#
owl:Class
owl:NamedIndividual
rdf:type
owl:ObjectProperty
owl:DataTypeProperty
xsd:DataType

Fig. 8. Work item quantity representation.

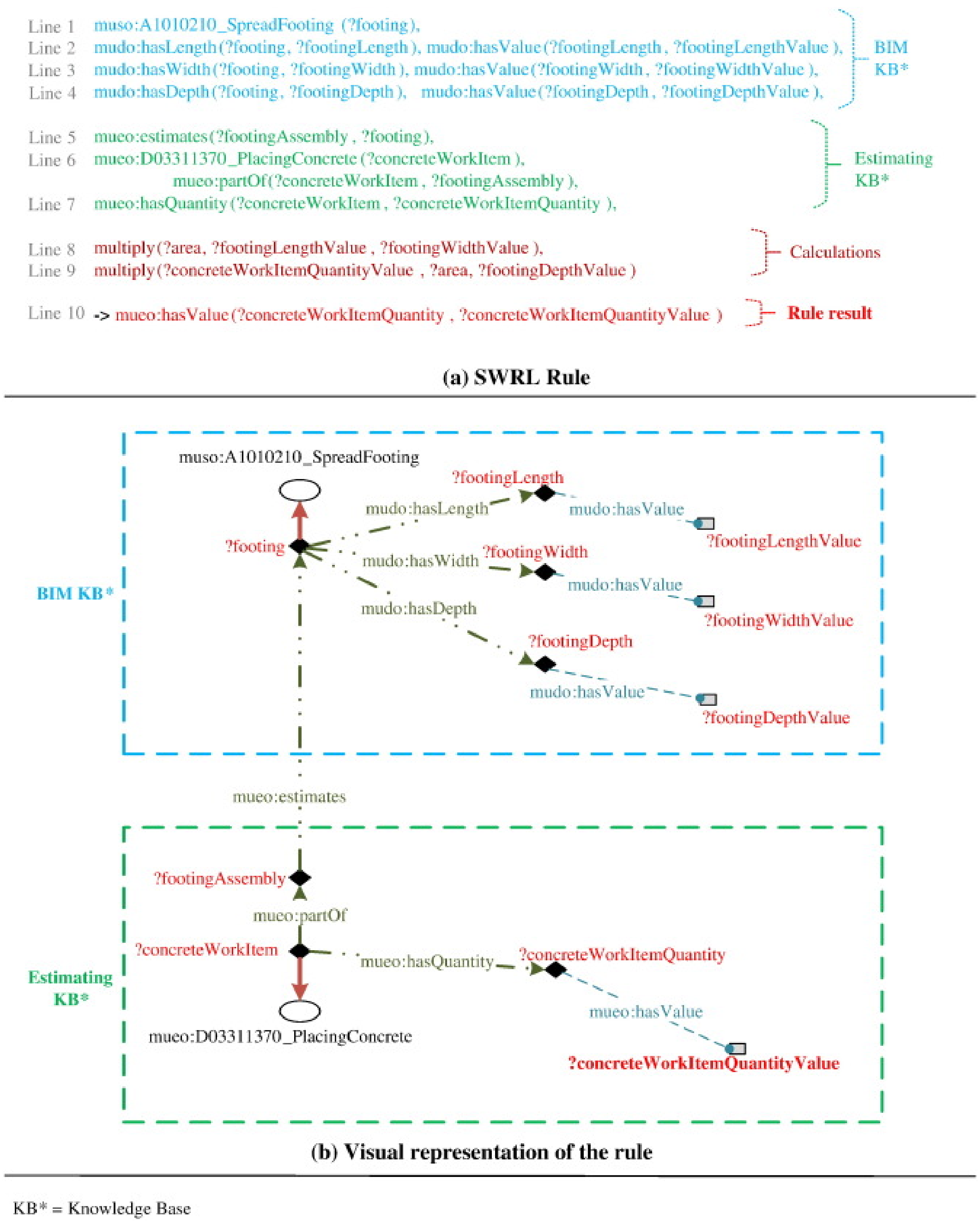
A SWRL rule for calculating the concrete quantity for a concrete-placing work item is shown in Fig. 9a. A graphical representation of the rule is shown in Fig. 9b. The following is a short description of the rule:

Lines 1 to 4: Footing dimensions are retrieved from the BIM knowledge base. In line 1, ?footing refers to an owl individual of type muso:A1010210_SpreadFooting. In lines 2 to 4, the values of the footing's length, width, and depth are retrieved.

Lines 5 to 7: A footing assembly (?footingAssembly) and an assembly work item (?concreteWorkItem) are retrieved from the estimating knowledge base. In line 5, ?footingAssembly is assigned to ?footing element. In line 6, the work item type (mueo:D03311370_PlacingConcrete) is specified. In line 7, the work item quantity (?concreteWorkItemQuantity) that must be calculated is defined.

Lines 8 and 9: The mathematical operations for calculating the work item quantity are defined. In line 8, ?area is calculated by multiplying ?footingLengthValue and ?footingWidthValue. In line 9, ?concreteWorkItemQuantityValue is calculated by multiplying ?area and ?footingDepthValue.

Line 10: Assigns the calculated work item quantity value to ?concreteWorkItemQuantity.



```
muso:A1010210_SpreadFooting (?footing),
mudo:hasLength(?footing, ?footingLength), mudo:hasValue(?footingLength, ?footingLengthValue ),
mudo:hasWidth(?footing, ?footingWidth), mudo:hasValue(?footingWidth, ?footingWidthValue ),
mudo:hasDepth(?footing, ?footingDepth),   mudo:hasValue(?footingDepth, ?footingDepthValue ),

mueo:estimates(?footingAssembly, ?footing),
mueo:D03311370_PlacingConcrete(?concreteWorkItem ),
            mueo:partOf(?concreteWorkItem, ?footingAssembly ),
mueo:hasQuantity(?concreteWorkItem, ?concreteWorkItemQuantity ),

multiply(?area, ?footingLengthValue, ?footingWidthValue ),
multiply(?concreteWorkItemQuantityValue , ?area, ?footingDepthValue )

-> mueo:hasValue(?concreteWorkItemQuantity , ?concreteWorkItemQuantityValue )
```

Fig. 9. SWRL rule to calculate placing concrete work item quantity.

In a similar manner, we developed a set of SWRL rules for calculating quantities of other work items. We used Pellet Reasoner[55] to calculate work item quantities based on the SWRL rules developed. Pellet provides standard reasoning services for OWL ontologies.

The work item shown in Fig. 7 uses a material resource (rst:ReadyMixConcrete_3256). A material resource is defined with properties such as material specifications and unit cost as shown in Fig. 10. We used FC ontology to represent material specifications. For example, cement type is specified using its FC URI (fc:P_E52) and label ("cement cem"). Unit cost property of material is a multi-attribute property that must be specified using a currency, a value, and a unit of measurement.

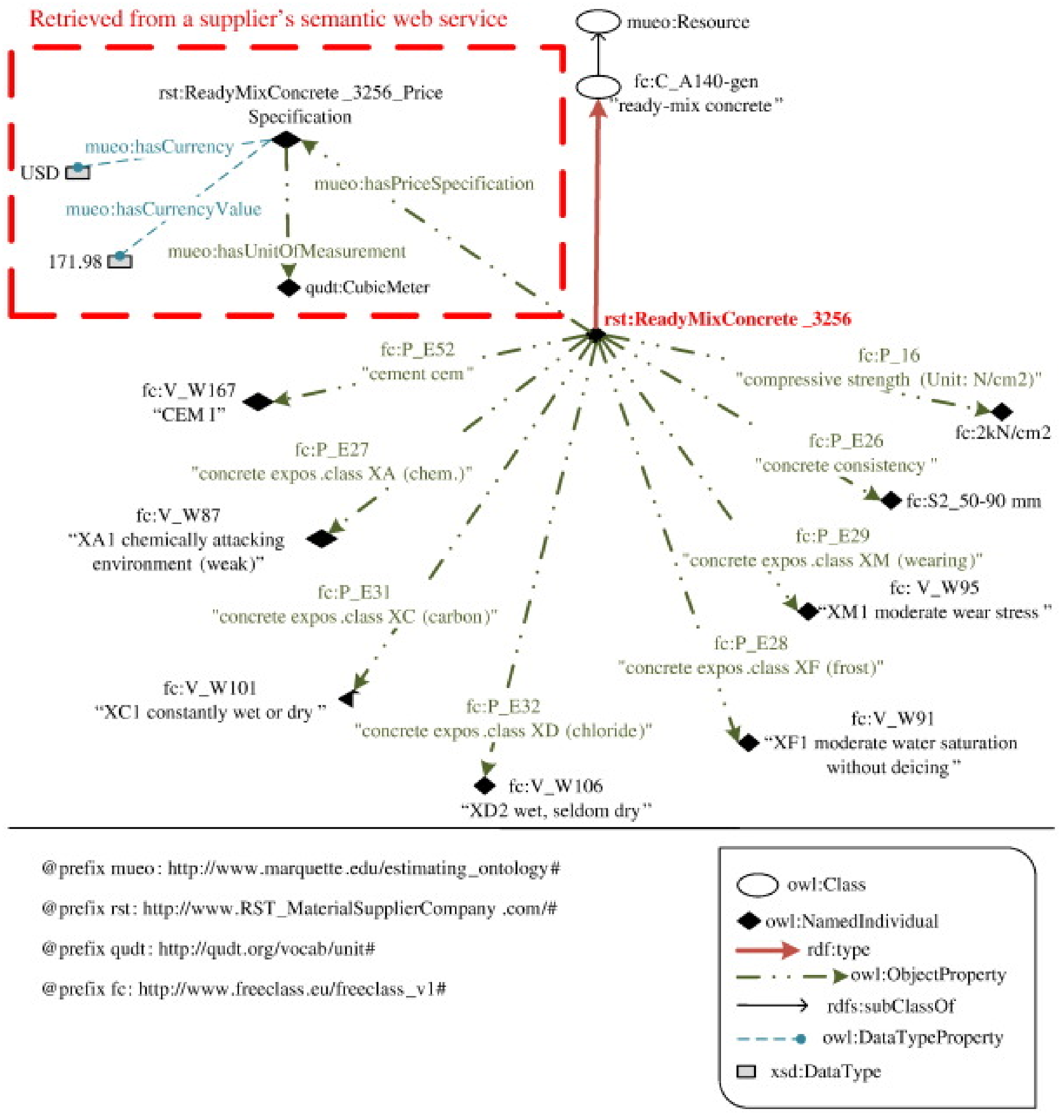

Retrieved from a supplier's semantic web service
mueo:Resource
fc:C_A140-gen
"ready-mix concrete"
rst:ReadyMixConcrete _3256_Price Specification
mueo:hasCurrency
USD
mueo:hasCurrencyValue
171.98
mueo:hasPriceSpecification
mueo:hasUnitOfMeasurement
qudt:CubicMeter
rst:ReadyMixConcrete _3256
fc:P_E52
"cement cem"
fc:V_W167
"CEM I"
fc:P_16
"compressive strength (Unit: N/cm2)"
fc:2kN/cm2
fc:P_E27
"concrete expos .class XA (chem.)"
fc:V_W87
"XA1 chemically attacking environment (weak)"
fc:P_E26
"concrete consistency"
fc:S2_50-90 mm
fc:P_E29
"concrete expos .class XM (wearing)"
fc: V_W95
"XM1 moderate wear stress"
fc:P_E31
"concrete expos .class XC (carbon)"
fc:V_W101
"XC1 constantly wet or dry"
fc:P_E28
"concrete expos .class XF (frost)"
fc:P_E32
"concrete expos .class XD (chloride)"
fc:V_W91
"XF1 moderate water saturation without deicing"
fc:V_W106
"XD2 wet, seldom dry"
@prefix mueo: http://www.marquette.edu/estimating_ontology#
@prefix rst: http://www.RST_MaterialSupplierCompany .com/#
@prefix qudt: http://qudt.org/vocab/unit#
@prefix fc: http://www.freeclass.eu/freeclass_v1#
owl:Class
owl:NamedIndividual
rdf:type
owl:ObjectProperty
rdfs:subClassOf
owl:DataTypeProperty
xsd:DataType

Fig. 10. Concrete material specification and unit cost.

The estimating application developed in this study communicates with material suppliers Semantic Web Services for obtaining the latest material unit costs; it sends material specifications to suppliers' Semantic Web Services and retrieves material unit cost information. More details about the FC ontology is provided in Section 7.3 of the paper.

We used the same methodology to represent crew, productivity, and work item cost information in the estimating knowledge base. We used the above approach to define estimating assemblies and work items for different building elements. We developed several user interfaces to allow an estimator to create, or modify estimating assembly and work item instances. The work items included in an assembly and the work item properties can be edited by an estimator. We created the assembly and work item ontologies in Protégé. We used the Java programming language and Appache Jena[56] library for creating assembly and work item instances and saving the estimating assembly knowledge base (including ontologies and instances) to an OpenRDF Sesame triplestore.[53] Jena is a Java framework for building Semantic Web applications. It includes a collection of tools and Java libraries to support Semantic Web programming; it includes reading, processing, writing, reasoning, storing, and querying RDF and OWL data. It supports SPARQL queries and rule-based inference engines such as Pellet Reasoner.

The following section describes the suppliers' Semantic Web Services developed in this study and how the presented estimating application can retrieve the latest material cost data from suppliers' Semantic Web Services.

7. Material suppliers' Semantic Web Services

The function of a web service is usually to supply information and may also involve exchange or sale of goods or obligations.[57] Web services allow applications distributed over the Internet to communicate with each other by exchanging or producing new information. Enterprise applications are increasingly using service-oriented architecture to communicate with web services provided by different parties.[58]

Although service-oriented architecture can facilitate data integration, it still requires human involvement in discovering and understanding different functions of web services. Web service technologies operate at the syntactic level[59] to support interoperability among application development platforms; however, they do not specify what happens when a service is executed. If an existing service is modified or a new service is created, computer applications that use the service must be revised in order to be able to communicate with the service. This can create difficulties for estimating application developers because of the large number of material suppliers involved in construction projects. Using regular service oriented architecture for cost estimating would require special programming code in the estimating application for each material supplier web service.

E-COGNOS[60] was a European project that developed web services to manage ontologies in the construction domain. They used other classifications such as IFC and ISO 12006-3 to develop construction domain ontologies. E-COGNOS used aspects of Semantic Web to formally document and update organizational knowledge. They utilized a set of web service related technologies such as Simple Object Access Protocol (SOAP), Universal Discovery Description and Integration (UDDI), Web Services Description Language (WSDL), and XML. Their web services are intended to enable users to use ontologies for specific tasks and is not intended to enable computer applications find and execute web services provided by other organizations.

Ontologies can be used to semantically describe web services, which can be referred to as “Semantic Web Services.”[61] Semantic descriptions of web services can automate service discovery, composition, and execution.[62] In this study, we used ontologies to semantically define material suppliers' web services and map them to estimating resource ontologies. This allows our estimating application to facilitate material suppliers' web service discovery, composition, and execution.

The estimating approach presented in Fig. 2 requires material suppliers to publish their data as Semantic Web Services. We developed a number of material suppliers' Semantic Web Services on a server in our computer lab to test the presented estimating approach. To retrieve a resource cost data, the estimating application directly communicates with a supplier's Semantic Web Service. Fig. 11 shows the ontologies used and the messages exchanged between the estimating application and a supplier's Semantic Web Service. The estimating application developed in this study uses a Semantic Web Service communication module to communicate with a supplier's Semantic Web Service. The function of this module is to send product specifications to the supplier's Semantic Web Service and to receive supplier's product offering information; this type of communication is performed in Simple Object Access Protocol (SOAP) format.

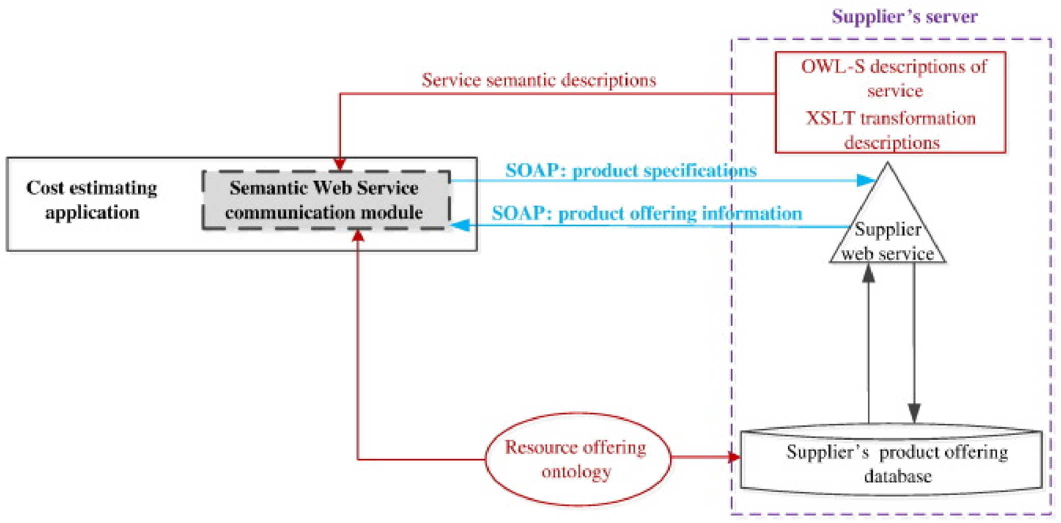

Supplier's server
Service semantic descriptions
OWL-S descriptions of service
XSLT transformation descriptions
Cost estimating application
Semantic Web Service communication module
SOAP: product specifications
SOAP: product offering information
Supplier web service
Resource offering ontology
Supplier's product offering database

Fig. 11. Accessing a supplier's Semantic Web Service.

The suppliers' Semantic Web Services developed in this study provide programming interfaces to access the suppliers' product offering databases that can be stored in RDF/OWL, relational or xml databases. We modeled suppliers' product offering databases using relational databases. The supplier web service converts the SOAP messages received from the estimating application to an SQL query for retrieving the requested data from a product offering database. To create a SOAP message, the estimating application's web service communication module accesses web service semantic descriptions (OWL-S and XSLT descriptions) and resource offering ontologies over the internet. These ontologies are discussed below.

7.1. Service semantic descriptions

We created semantic descriptions of suppliers' web services to enable our estimating application to identify which services must be accessed, prepare input messages for those services, execute web services, and integrate data returned from the web services into the estimating knowledge bases. In this study, we used OWL-S[16] to provide Semantic Web Service descriptions. OWL-S is a framework to provide computer-interpretable descriptions of web services and the means by which they are accessed.[63] Our estimating application uses these descriptions to learn how to use services. OWL-S provides semantic description for a web service by defining three sub-ontologies: service profile ontology, process model ontology, and grounding ontology.

1. OWL-S service profile ontology provides semantic description of what a service does and the limitations, quality, and requirements of the service. The estimating application developed in this study uses service profile descriptions to find an appropriate service.
2. OWL-S process model ontology describes how to use and request a service, and what happens when that service is requested. The process model ontology is also used to compose and coordinate different web services.
3. OWL-S service grounding ontology specifies communication protocols, message formats, port numbers, and other details that describe how a supplier service can be accessed.

We also used XSLT[64] transformation descriptions for converting RDF data to SOAP and vice-versa. In this study, the presented estimating application uses OWL-S, XSLT transformations, and resource offering ontology to communicate with suppliers' Semantic Web Services.

7.2. Resource offering ontology

The estimating application presented in this study retrieves resource costs from material suppliers' Semantic Web Services. Construction material suppliers provide resource offerings that include resource specifications and costs. In order to semantically define construction suppliers' offerings, a resource offering ontology is required. We used Good Relations (GR)[33] for this purpose. GR is developed to allow businesses to semantically define their product offerings and publish them on the web.[34] GR provides a conceptual model for general concepts such as company, store location, offer, product descriptions, price, payment, shipment, and warranty information.[65] GR has been adopted and is widely used in web-based eCommerce.[10,66,67] For example, Best Buy and overstock.com, two of the largest retail chains for consumer products have used GR to publish their product data. Search engines such as Google use GR to enhance the information they provide about a product in response to a search query. In this research, we used GR to represent semantic descriptions of construction suppliers' offerings.

Fig. 12 shows the ontology for a business entity offering a product or service. GR allows business entities to offer their products using three classes: BusinessEntity, Offering, and ProductOrService. These classes are connected to each other using two object properties: offers and includes. A business entity offers its offerings, and each offering includes a product or service. As Fig. 12 shows, a product or a service is described with qualitative and quantitative properties. A unit price specification describes the currency, currency value, and unit of measurement properties of an offer. GR allows business entities to provide more details about their offers such as available delivery methods, accepted payment methods, and advanced booking requirements.

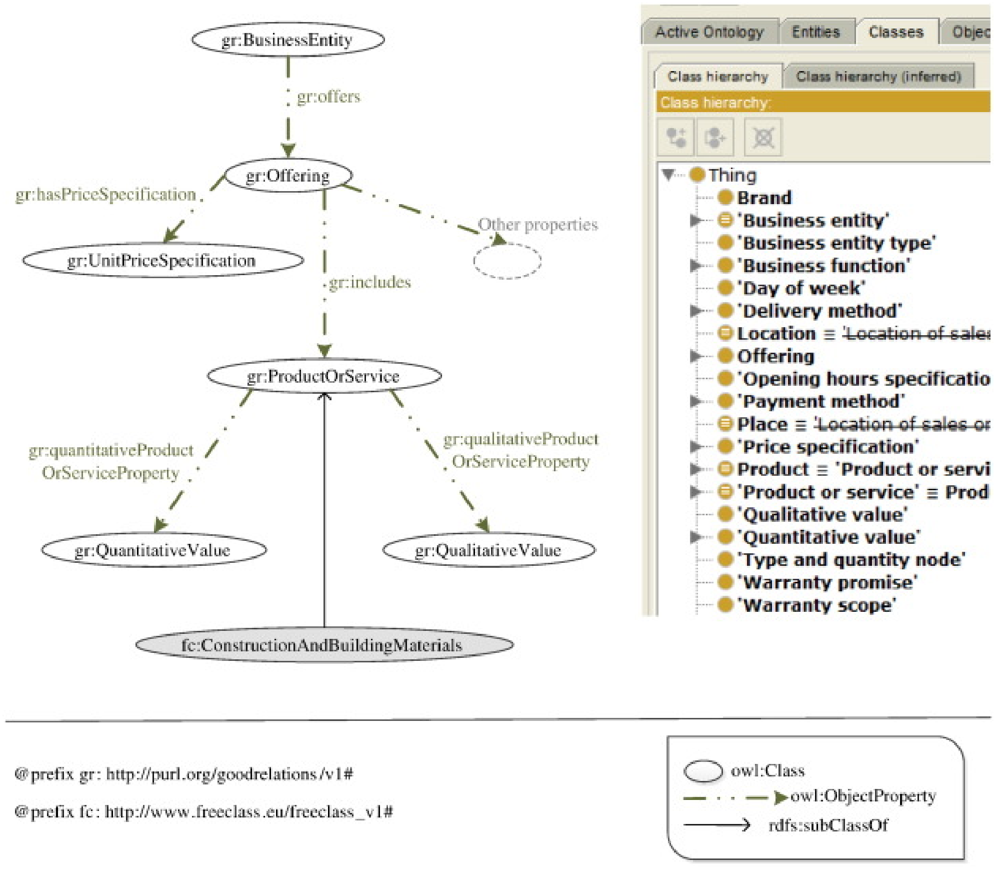
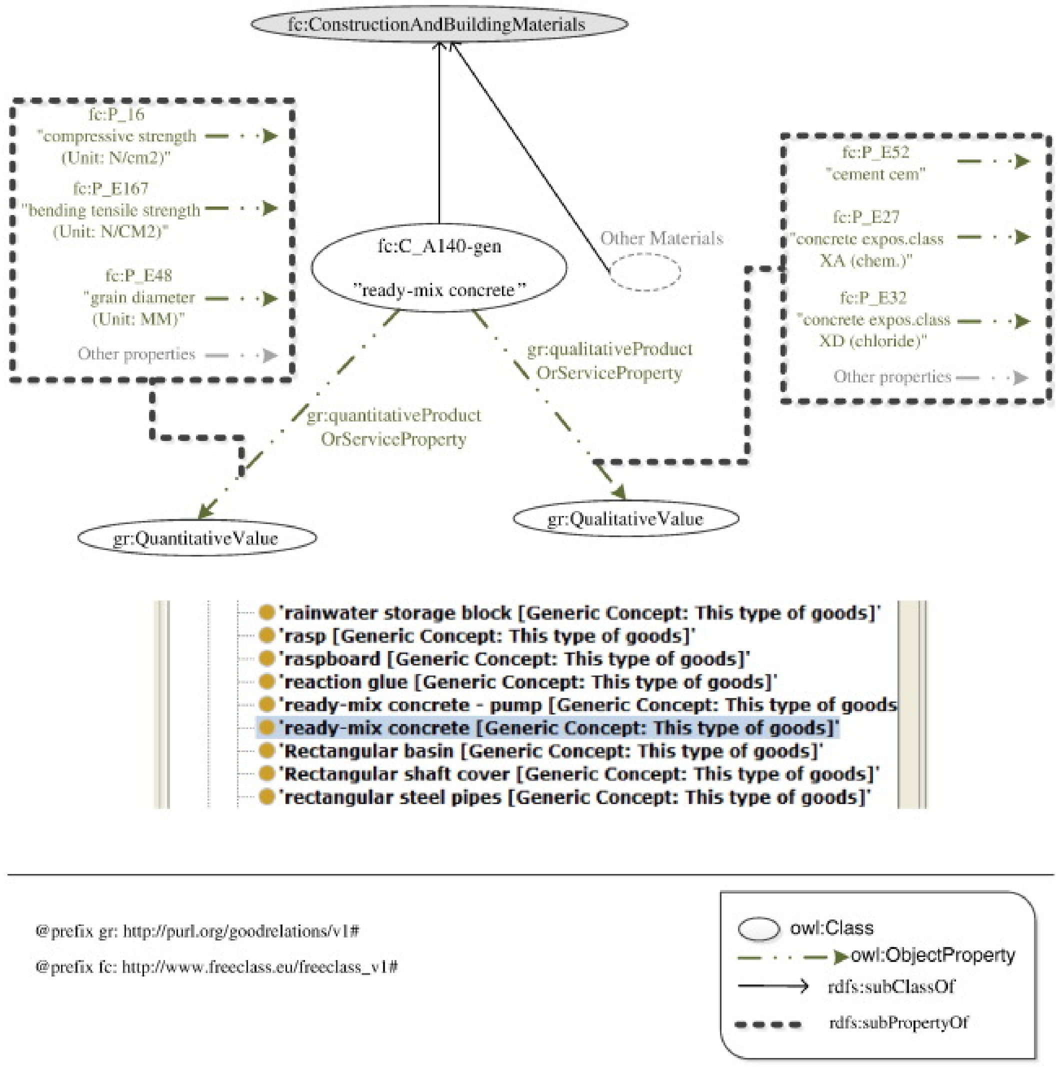


Fig. 12. Part of resource offering ontology.

7.3. Material ontology

The Free Class (FC) OWL ontology[32] is developed by the European Building and Construction Materials Database to describe construction materials and services. FC has over 88 million triples of real business data to represent construction material and services domain. FC is a W3C and GR compliant ontology. FC and GR vocabularies allow construction material suppliers to semantically define and publish their construction product offerings. Fig. 13 shows the ready-mix concrete material class in the FC ontology. Every material in FC is described with quantitative and qualitative properties. Several quantitative properties of concrete such as compressive strength, bending tensile strength, and grain diameter are shown on the left side of Fig. 13. These FC properties are sub-properties of the GR property quantitativeProductOrServiceProperty. Other FC quantitative properties are included in the ontology in a similar manner. On the right hand of Fig. 13, some of the qualitative properties defined in FC for concrete are presented. These properties are sub-properties of GR property qualitativeProductOrServiceProperty. Other FC qualitative properties are defined in the ontology in a similar manner.


fc:ConstructionAndBuildingMaterials
fc:P_16 "compressive strength (Unit: N/cm2)"
fc:P_E167 "bending tensile strength (Unit: N/CM2)"
fc:P_E48 "grain diameter (Unit: MM)"
Other properties
fc:C_A140-gen "ready-mix concrete"
Other Materials
fc:P_E52 "cement cem"
fc:P_E27 "concrete expos.class XA (chem.)"
fc:P_E32 "concrete expos.class XD (chloride)"
Other properties
gr:qualitativeProduct OrServiceProperty
gr:quantitativeProduct OrServiceProperty
gr:QuantitativeValue
gr:QualitativeValue
@prefix gr: http://purl.org/goodrelations/v1#
@prefix fc: http://www.freeclass.eu/freeclass_v1#
owl:Class
owl:ObjectProperty
rdfs:subClassOf
rdfs:subPropertyOf

'rainwater storage block [Generic Concept: This type of goods]'
'rasp [Generic Concept: This type of goods]'
'raspboard [Generic Concept: This type of goods]'
'reaction glue [Generic Concept: This type of goods]'
'ready-mix concrete - pump [Generic Concept: This type of goods
'ready-mix concrete [Generic Concept: This type of goods]'
'Rectangular basin [Generic Concept: This type of goods]'
'Rectangular shaft cover [Generic Concept: This type of goods]'
'rectangular steel pipes [Generic Concept: This type of goods]'

Fig. 13. Ready-mix concrete class in FC ontology.

In this study, we used GR and FC to develop suppliers' product offering Semantic Web Services. We defined the inputs and outputs of suppliers' Semantic Web Services according to GR and FC. The estimating application developed in this study uses GR and FC to send product specifications to suppliers' Semantic Web Services and retrieve the latest material unit costs for updating the assembly and work item knowledge bases.

Fig. 14 shows a section of a ready mixed concrete supplier knowledge base we developed in this study. In the list of prefixes, “gr” and “fc” represent Good Relations and Free Class OWL ontologies. “rst” is the URI of a ready mix concrete material supplier. “rst:Offering_1823” defines the specifications and unit cost for a ready mixed concrete product. The GR vocabulary enables suppliers to describe their offerings with information such as price, store location, warranty information, available delivery methods, accepted payment methods, and advanced booking requirements. Fig. 14 shows the price specification that is described with currency, currency value, and unit of measurement. Similarly other aspects of each offer are included in the supplier knowledge base using the GR vocabulary.

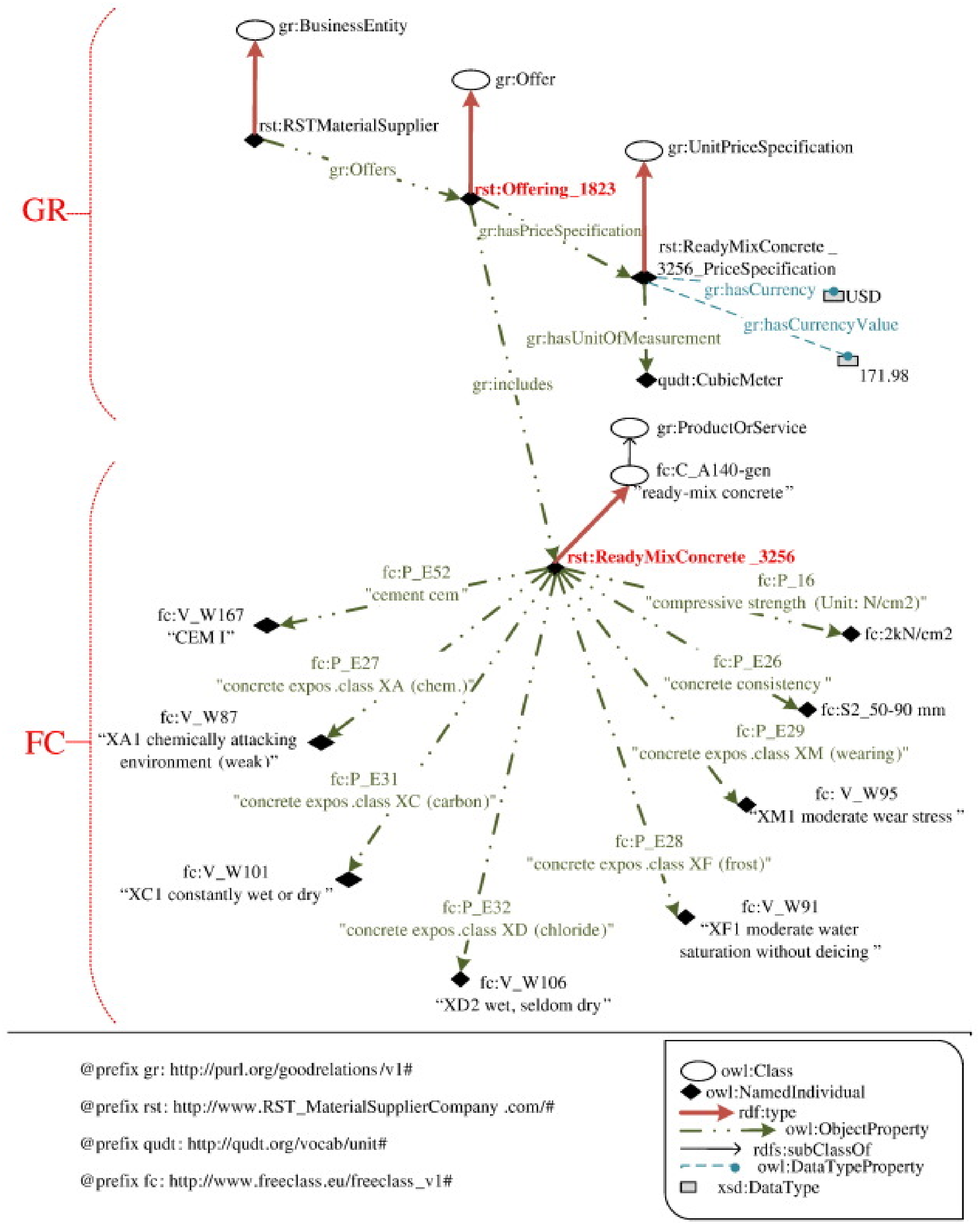

GR
FC
gr:BusinessEntity
gr:Offer
rst:RSTMaterialSupplier
gr:Offers
gr:UnitPriceSpecification
rst:Offering_1823
gr:hasPriceSpecification
rst:ReadyMixConcrete_3256_PriceSpecification
gr:hasCurrency
USD
gr:hasCurrencyValue
171.98
gr:hasUnitOfMeasurement
qudt:CubicMeter
gr:includes
gr:ProductOrService
fc:C_A140-gen "ready-mix concrete"
rst:ReadyMixConcrete_3256
fc:P_E52 "cement cem"
fc:V_W167 "CEM I"
fc:P_16 "compressive strength (Unit: N/cm2)"
fc:2kN/cm2
fc:P_E27 "concrete expos.class XA (chem.)"
fc:V_W87 "XA1 chemically attacking environment (weak)"
fc:P_E26 "concrete consistency"
fc:S2_50-90 mm
fc:P_E29 "concrete expos.class XM (wearing)"
fc:P_E31 "concrete expos.class XC (carbon)"
fc: V_W95 "XM1 moderate wear stress"
fc:V_W101 "XC1 constantly wet or dry"
fc:P_E28 "concrete expos.class XF (frost)"
fc:P_E32 "concrete expos.class XD (chloride)"
fc:V_W91 "XF1 moderate water saturation without deicing"
fc:V_W106 "XD2 wet, seldom dry"
@prefix gr: http://purl.org/goodrelations/v1#
@prefix rst: http://www.RST_MaterialSupplierCompany.com/#
@prefix qudt: http://qudt.org/vocab/unit#
@prefix fc: http://www.freeclass.eu/freeclass_v1#
owl:Class
owl:NamedIndividual
rdf:type
owl:ObjectProperty
rdfs:subClassOf
owl:DataTypeProperty
xsd:DataType

Fig. 14. Part of a material supplier knowledge base.

We created the supplier Semantic Web Services prototypes for this study using Java programming language. We defined the inputs and outputs of these web services according to GR and FC ontologies. We created OWL-S descriptions of the web services using the OWL-S API.[68,69] The OWL-S API provides a Java API for programmatic access to create, read, write, and execute Semantic Web Services. We created Suppliers' product offering databases in Microsoft SQL server.

Using the new estimating approach requires construction material suppliers to semantically define and publish their product specifications and cost data on the web as Semantic Web Services. Currently, none of the construction material suppliers use ontologies to describe their products. However, several large consumer companies have started to use ontologies to semantically define their products and services for web commerce including Best Buy and Overstock.com. These companies have successfully implemented semantically defined product offering ontologies such as Good Relations. In essence, the technology is available if industry decides to use it.

The following section describes a prototype estimating application developed in this study to test the new approach. The estimating application developed in this study is able to find and execute material suppliers' Semantic Web Services using the above mentioned ontologies to retrieve resource cost data.

8. Prototype estimating application

This section describes a prototype estimating application developed in this study based on the approach presented in Fig. 2. In this approach, an estimating application uses a BIM knowledge base, an estimating assembly and work item knowledge base, and various suppliers' Semantic Web Services to create an estimate. We developed a cost estimating application using Java programming language. The BIM and estimating assembly and work item knowledge bases are stored in OpenRDF Sesame triplestore. We used Jena[56] framework to bring the capabilities of Pellet reasoner to the estimating application. We created a Semantic Web Service communication module as part of the estimating application using the OWL-S API.[68,69]

Fig. 15 shows the use cases we developed in our prototype. Our prototype provides the most basic functionality for an estimating application. As Fig. 15 shows, an estimator can select a building model, prepare a list of material suppliers, define estimating assemblies and work items, map BIM elements to estimating assemblies, and perform an estimate. These activities require access to a BIM knowledge base, an estimating assembly and work item knowledge base, and material suppliers' Semantic Web Services. The estimating application interactions with knowledge bases and Semantic Web Services are developed as asynchronous transactions so that the estimating application would be responsive in case a supplier's Semantic Web Service is slow.

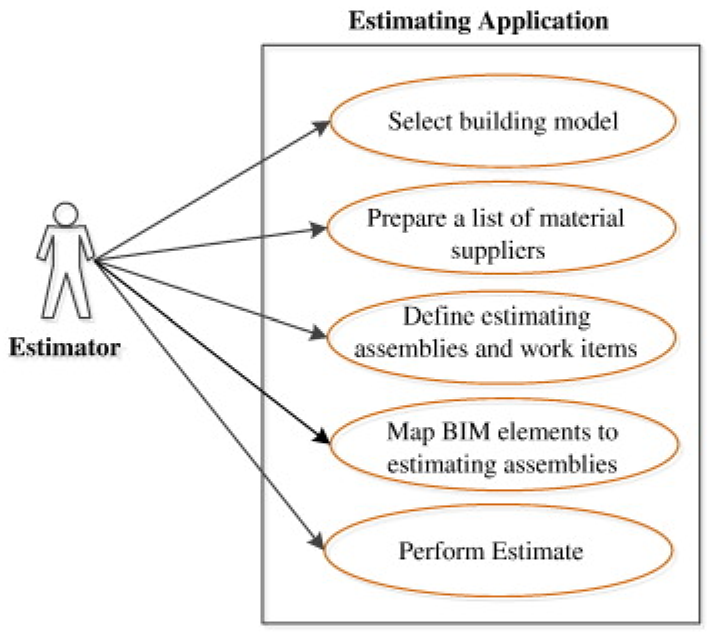

Estimating Application
Select building model
Prepare a list of material suppliers
Define estimating assemblies and work items
Map BIM elements to estimating assemblies
Perform Estimate
Estimator

Fig. 15. Developed estimator use cases.

A brief description for each use case is presented below:

1. Select building model. The estimating application user interface allows a user to select a building model from a Sesame BIM triple-store server. To do this, the user enters the URI of the building to be estimated. For example, the URI of the building project used in this study is abc:EngineeringHall as shown in Fig. 3. The estimating application uses the building project URI to establish a connection with the BIM knowledge base of the building project stored in the building designer's Sesame server.
2. Prepare a list of material suppliers. The user creates a list of material suppliers to be used for the project. The supplier list includes information such as supplier name and URI.
3. Define estimating assemblies and work items. An estimating application requires predefined assemblies and work items as discussed in section 6 of the paper. We developed interfaces that allow a user to create work items and assemblies and store them in a Sesame knowledge base. The work item user interface allows a user to create a new work item and enter the properties discussed in Section 6 for the work item. The user also selects the material supplier for the work item. An assembly user interface allows the user to define a new assembly and assign a number of predefined work items to the assembly.
4. Map BIM elements to estimating assemblies. An assembly is designed to estimate the cost of a building element. An element mapping user interface provides the user with two lists: the first list contains the names of predefined estimating assemblies from the estimating knowledge base and the second list includes building element types retrieved from the project BIM knowledge base. The user maps each model element type to the assembly designed for estimating the element.
5. Perform estimate. Perform estimate is a command button that executes a number of program processes. These processes perform the following tasks:
   - Iterate through BIM knowledge base elements.
   - For each BIM element, retrieve the corresponding assembly from the estimating knowledge base.
   - Retrieve the assembly work items.
   - For each work item, calculate work item quantity.
   - For each work item, submit material specifications to the selected material supplier and receive material unit cost.
   - Calculate work item cost.
   - Calculate assembly cost by adding assembly work item costs.
   - Calculate project cost by adding all project assembly costs.
   - Return project cost.

9. Validation

The purpose of this validation is to measure the time it takes to create a cost estimate using our semantics-based approach compared with a commercial computer estimating application. We used WinEst[2] estimating software for this purpose. WinEst is a popular commercial estimating software. It was readily available for this study because it is also used by the second author in a construction cost estimating course that he teaches at Marquette University.

Currently, to prepare a BIM-based cost estimate for a project, an estimator must complete the following steps:

1. Edit assemblies of work items in the estimating application that represent BIM elements to be estimated.
2. Map BIM elements to their corresponding estimating assemblies. For example, map a spread footing element to a spread footing assembly in the estimating software.

3. Map BIM element properties to the estimating assembly properties to calculate work item quantities. For the spread footing example, this step involves mapping the dimensions of a BIM spread footing element to the corresponding estimating assembly dimensions.
4. Update material resource unit costs in the estimating application database.

Currently, estimators manually perform the above mentioned steps. Our semantics-based approach modifies how steps 3 and 4 are performed. To calculate work item quantities in step 3, our estimating application eliminates the need for manually mapping element properties to the corresponding assembly properties by semantically defining the BIM elements and the estimating assemblies. For updating the estimating application material resource unit cost database (step 4), our estimating application retrieves resource unit costs directly from suppliers' Semantic Web Services as explained in Section 7 of this paper. By eliminating the manual activities from steps 3 and 4, our approach improves estimator efficiency.

A building project includes a large number of elements and resources. We limited the validation of our estimating approach to structural concrete elements in a 3 story educational building project named Engineering Hall. Our validation included a total of 40 elements including footings, columns, beams, and slabs. For each element we mapped length, width, and depth of each BIM element to its corresponding estimating assembly dimensions. Validation included only concrete material and concrete pouring operations.

9.1. Mapping BIM element properties to estimating assembly properties

We investigated the efficiency of our semantics-based approach relative to WinEst's method for calculating concrete quantities for the above-mentioned 40 structural elements in Engineering Hall. We used 5 estimators familiar with WinEst software; the same estimators were taught the interfaces to our software and how to use the software for estimating purposes. The five estimators were asked to:

1. Prepare the necessary estimating assemblies for concrete footings, columns, beams and slabs.
2. Map BIM elements to their corresponding estimating assemblies.
3. Map the dimensions of each BIM element to their corresponding estimating assembly properties to calculate the concrete quantity for the BIM element.

The time required to complete the first two steps are almost the same in both WinEst and our approach. In Step 3, WinEst requires an estimator to manually map element dimensions to their corresponding assembly dimensions. When using WinEst, the mapping required in step 3 took an average of 53 s per concrete element. The same estimators also used our estimating application to estimate the costs of the same building elements. Since in our approach, BIM elements and estimating assemblies are semantically defined, the mapping required in step 3 is machine processable and does not require estimator involvement. The total time saved for estimating 40 concrete elements was about 35 min and 20 s. Considering the fact that a construction project includes a large number of elements, eliminating step 3 substantially improves estimator efficiency.

9.2. Updating estimating application's material resource unit cost database

In a computer lab, we developed two web servers representing two ready-mixed concrete material suppliers. Each web server published concrete material specifications and their respective unit costs in two formats: (1) in a table format for human access and (2) as Semantic Web Services for computer access. In an experiment, we asked the same 5 estimators to update a material unit cost database using the tabulated data published on suppliers' web sites. The estimators were asked to obtain the minimum unit costs for the concrete mixes needed for Engineering Hall from the supplier web sites and update a WinEst material database. It took the estimators on the average 1 min and 58 s to update the unit cost for each of the four concrete mix specifications used in the 40 building elements investigated. Our estimating application directly accessed suppliers' Semantic Web Services, submitted the specifications for the required concrete mixes, obtained suppliers' material unit costs, and used the minimum material cost to update the estimating software's material database. The total time saved in updating unit costs for four concrete specifications was about 7 min and 52 s. The results of this experiment demonstrates the potential for reducing the time required for updating material cost databases when a large number of material unit costs must be updated.

10. Limitations and directions for further research

The estimating approach discussed in this paper requires a number of knowledge bases. To allow knowledge sharing, the required knowledge bases must be based on standard ontologies that are not available yet. The prototype knowledge bases used in this study were created based on ontologies that we developed.

The architecture and engineering industry must develop a standard ontology that can be used for creating BIM knowledge bases. There have been efforts to use EXPRESS-to-OWL conversion procedures for developing an ifcOWL ontology.[48–51] However, none of the ifcOWL ontologies have become industry standards yet.

Standard ontologies are also needed for creating estimating knowledge bases that represent estimating assemblies, work items and construction resources. The construction estimating ontologies presented in this paper can serve as the starting point for further research towards development of industry standard ontologies.

As discussed in the paper, to make updating of material cost databases machine processable, construction material suppliers must provide material information as knowledge bases that can be accessed over the Internet. The Good Relations ontology[33] discussed in this paper is used by a number of businesses such as Best Buy, Overstock.com, Yahoo!, and Google[10] for providing product data or for product search purposes. Future development of a standard material ontology would allow construction material suppliers to provide material information as knowledge bases that can be searched and queried by remote computers.

Semantically defined construction knowledge can lead to other innovations that would improve the AEC industry efficiency. Niknam and Karshenas[11] have used BIM and construction knowledge bases to create a project social networking website that improves communication among project participants and allows adding new knowledge to project knowledge bases. Incorporating social networking and construction project knowledge can initiate new and innovative approaches to construction cost estimating.

11. Summary and conclusion

Construction cost estimating requires access to several sources of data. These sources include BIM created by designers, estimating assemblies and work items created and maintained by contractors, and resource unit cost data provided by material suppliers. In this paper, we investigated the application of Semantic Web and Semantic Web Service technologies to facilitate finding, accessing, and combining the information necessary for construction cost estimating. One of the objectives of this paper was to present the information needed for cost estimating in a semantics-based format. We used ontologies to create (1) a BIM knowledge base, (2) an estimating assembly and work item knowledge base, and (3) suppliers' Semantic Web Services. We developed a prototype cost estimating application that can access these knowledge bases and Semantic Web Services to perform cost estimating.

To validate our estimating application, we compared its performance with a commercial estimating program called WinEst. We asked 5 estimators to perform quantity takeoffs for 40 structural concrete elements in a 3 story building. The building elements investigated included footings, columns, beams, and slabs. The WinEst estimating software requires an estimator to manually map element dimensions to their corresponding estimating assembly dimensions. The average time an estimator spent to map a concrete element's length, width and depth properties to its corresponding WinEst assembly properties was about 53 s. In our approach, BIM elements and estimating assemblies are semantically defined which makes the mapping machine processable and eliminates the need for estimator involvement in the element property mapping process. The total time saved for mapping 40 concrete elements was about 35 min and 20 s. Since a construction project includes a large number of element property mappings, our approach can substantially improve estimators' efficiency.

Updating material resource unit cost databases is another time consuming task for estimators. Currently, estimators manually update material cost databases. We presented an approach that requires material specifications and costs available as Semantic Web Services. In our approach, a semantics-based cost estimating application can submit material resource specifications to suppliers' Semantic Web Services and retrieve material unit cost data for updating material cost databases. In a lab experiment, we compared our approach to updating estimating material cost databases to the current manual methods. We observed that on the average it takes 1 min and 58 s less time for updating a concrete mix unit cost in a resource cost database if material suppliers' product data are available as Semantic Web Services. The total time saved in updating unit costs for four concrete specifications used in 40 structural concrete elements was about 7 min and 52 s. Considering the fact that estimating databases include a large number of material resources and estimating material cost databases must be updated before a new estimate, our approach has the potential to substantially improve estimating efficiency.

The purpose of this paper was to show the potential of using Semantic Web technology in construction cost estimating. Although Semantic Web technology is being used in web commerce, its application to construction cost estimating requires that the construction industry further defines and standardizes the ontologies, knowledge bases, and Semantic Web Services discussed in this paper.

References

[1] S. Aram, C. Eastman, J. Beetz. **Qualitative and quantitative cost estimation.** *Methodol. Anal.* (2014), pp. 381-389

[2] WinEst. http://www.meridiansystems.com/products/winest/overview/ (2015)

[3] **Autodesk Revit.** http://www.autodesk.com/education/free-software/revit (2015)

[4] M. Niknam, S. Karshenas. **A semantic web service approach to construction cost estimating.** *Comput. Civ. Eng.,* 2013 (2013), pp. 484-491

[5] **W3C Standards: Semantic Web.** http://www.w3.org/standards/semanticweb/ (2015)

[6] **W3C Standards: Ontologies.** http://www.w3.org/standards/semanticweb/ontology (2015)

[7] T.R. Gruber. **A translation approach to portable ontology specifications.** *Knowl. Acquis.*, 5 (1993), pp. 199-220

[8] **Introduction to Ontologies and Semantic Web.** http://obitko.com/tutorials/ontologies-semantic-web/ (2015)

[9] W3C Semantic Web Best Practices and Deployment Working Group. http://www.w3.org/2001/sw/BestPractices/ (2015)

[10] D. Allemang, J. Hendler. **Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL.** Elsevier (2011)

[11] M. Niknam, S. Karshenas. **A social networking website for AEC projects.** *Comput. Civil Build. Eng.,* 2208–2215 (2014)

[12]P. Hitzler, M. Krotzsch, S. Rudolph. **Foundations of Semantic Web Technologies.** Chapman and Hall/CRC (2011)

[13]H. Knublauch, D. Oberle, P. Tetlow, E. Wallace, J. Pan, M. Uschold. **A semantic web primer for object-oriented software developers.** W3C Working Group Note, W3C (2006)

[14]N.F. Noy, D.L. McGuinness. **Ontology Development 101: A Guide to Creating Your First Ontology.** (2001)

[15]E. Prud'Hommeaux, A. Seaborne. **SPARQL query language for RDF.** W3C Recommendation, 15 (2008)

[16]D. Martin, M. Burstein, J. Hobbs, O. Lassila, D. McDermott, S. McIlraith, *et al.* **OWL-S: semantic markup for web services.** W3C Member Submission, 22 (2004). (2007-2004)

[17]M. Uschold, M. King. **Towards a Methodology for Building Ontologies, Citeseer.** (1995)

[18]T.R. Gruber. **Toward principles for the design of ontologies used for knowledge sharing?** *Int. J. Hum. Comput. Stud.,* 43 (1995), pp. 907-928

[19]M. Fernández-López, A. Gómez-Pérez, N. Juristo. **Methontology: From Ontological Art Towards Ontological Engineering.** (1997)

[20]Y. Sure, S. Staab, R. Studer. **On-To-Knowledge Methodology (OTKM).** *Handbook on Ontologies*, Springer (2004), pp. 117-132

[21]H.S. Pinto, S. Staab, C. Tempich. **DILIGENT: towards a fine-grained methodology for Distributed.** *Loosely-controlled and evolvInG Engineering of oNTologies*, 16 (2004), p. 393

[22]M.C. Suárez-Figueroa. **NeOn Methodology for building ontology networks: specification, scheduling and reuse.** Dissertations in Artificial Intelligence, Vol. 338, IOS Press (2012). (ISBN 978-1-61499-115-1 Available at: http://oa.upm.es/3879/)

[23]M.C. Suárez-Figueroa, A. Gómez-Pérez, E. Motta, A. Gangemi. **Ontology Engineering in a Networked World.** Springer Science & Business Media (2012)

[24]M.C. Suárez-Figueroa, R. García-Castro, B. Villazón-Terrazas, A. Gómez-Pérez. **Essentials in Ontology Engineering: Methodologies, Languages, and Tools.** (2011)

[25]**Resource Description Framework (RDF).** http://www.w3.org/RDF/ (2015)

[26]**W3C Standards: OWL Web Ontology Language Current Status.** http://www.w3.org/standards/techs/owl#w3c_all (2015)

[27]**protégé.** http://protege.stanford.edu (2015)

[28]**RSMeans Construction Cost Data.** http://www.rsmeans.com/ (2015)

[29]**UNIFORMAT II Classification System.** http://www.astm.org/Standards/E1557.htm (2015)

[30]**MasterFormat System of Classification.** http://www.csinet.org/masterformat (2015)

[31]R Hodgson, PJ Keller, J Hodges, J Spivak. QUDT–Quantities, Units, Dimensions and Data Types Ontologies. 2013, URL http://qudt.org. (2011).

[32]BauDataWeb. **The European Building and Construction Materials Database for the Semantic.** (2015). (http://semantic.eurobau.com/>)

[33]**Good Relations Ontology.** http://www.heppnetz.de/projects/goodrelations/ (2015)

[34]M. Hepp. **Goodrelations: an ontology for describing products and services offers on the web.** *Knowledge Engineering: Practice and Patterns*, Springer (2008), pp. 329-346

[35]World Wide Web Consortium. **The Organization Ontology.** (2014)

[36]J. Hebeler, M. Fisher, R. Blace, A. Perez-Lopez. **Semantic Web Programming.** John Wiley & Sons (2011)

[37]T. Segaran, C. Evans, J. Taylor. **Programming the Semantic Web.** O'Reilly Media, Inc. (2009)

[38]C. Lagoze, J. Hunter. **"The ABC ontology and model."** *Journal of Digital Information* 2, no. 2 (2006)

[39]J.R. Hobbs, F. Pan. **Time ontology in OWL.** W3C Working Draft, 27 (2006), p. 133

[40]S. Staub-French, M. Fischer, J. Kunz, B. Paulson. **A generic feature-driven activity-based cost estimation process.** *Adv. Eng. Inform.*, 17 (2003), pp. 23-39

[41]M.P. Nepal, S. Staub-French, R. Pottinger, J. Zhang. **Ontology-based feature modeling for construction information extraction from a building information model.** *J. Comput. Civ. Eng.,* 27 (2012), pp. 555-569

[42]IFC overview. http://www.buildingsmart-tech.org/specifications/ifc-overview (2015)

[43]**RDF vs. XML.** https://www.cambridgesemantics.com/semantic-university/rdf-vs-xml (2015)

[44]P. Pauwels, D. Van Deursen, J. De Roo, T. Van Ackere, R. De Meyer, R. Van de Walle, *et al.* **Three-dimensional information exchange over the semantic web for the domain of architecture, engineering, and construction, artificial intelligence for engineering design.** Anal. Manuf., 25 (2011), pp. 317-332
[45]S. Aram, C. Eastman, R. Sacks. **A Knowledge-Based Framework for Quantity Takeoff and Cost Estimation in the AEC Industry Using BIM.** (2014)
[46]M. Maghiar, L. Livingston, A. Wiezel. **Technology Ontology and BIM-Enabled Estimating for Owners and Contractors.** (2014), pp. 2200-2207
[47]S. Lee, K. Kim, J. Yu. **BIM and ontology-based approach for building cost estimation.** *Autom. Constr.*, 41 (2014), pp. 96-105
[48]J. Beetz, J. Van Leeuwen, B. De Vries. **IfcOWL: a case of transforming EXPRESS schemas into ontologies, artificial intelligence for engineering design.** *Anal. Manuf.,* 23 (2009), pp. 89-101
[49]E. Karan, J. Irizarry, J. Haymaker. **Generating IFC models from heterogeneous data using semantic web.** *Construct. Innov.,* 15 (2015)
[50]E.P. Karan, J. Irizarry. **Extending BIM interoperability to preconstruction operations using geospatial analyses and semantic web services.** *Autom. Constr.,* 53 (2015), pp. 1-12
[51]P Pauwels, W Terkaj, ifcOWL ontology < www.w3.org/community/lbd/ifcOWL > 2015 (2014).
[52]S. Karshenas, M. Niknam. **Ontology-based building information modeling.** *Comput. Civil Eng.,* 2013 (2013), pp. 476-483
[53]**OpenRDF Sesame Triplestore.** http://rdf4j.org/ (2015)
[54]I. Horrocks, P.F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, M. Dean. **SWRL: a semantic web rule language combining OWL and RuleML.** W3C Member Submission, 21 (2004), p. 79
[55]**Pellet Reasoner.** http://clarkparsia.com/pellet/ (2015)
[56]Appache Jena. http://jena.apache.org/ (2015)
[57]D. Martin, M. Burstein, D. Mcdermott, S. Mcilraith, M. Paolucci, K. Sycara, *et al.* **Bringing semantics to web services with OWL-S.** World Wide Web, 10 (2007), pp. 243-277
[58]G. Li, V. Muthusamy, H.A. Jacobsen. **A distributed service-oriented architecture for business process execution.** *ACM Trans.* Web, 4 (2010), p. 2
[59]D. Roman, U. Keller, H. Lausen, J. de Bruijn, R. Lara, M. Stollberg, *et al.* **Web service modeling ontology.** *Appl. Ontol*., 1 (2005), pp. 77-106
[60]C. Lima, T. El-Diraby, J. Stephens. **Ontology-based optimization of knowledge management in e-construction.** *J. IT Constr*., 10 (2005), pp. 305-327
[61]B. Grasic, V. Podgorelec. **Automating ontology based information integration using service orientation.** WSEAS *Trans. Comput*., 9 (2010), pp. 547-556
[62]D. Fensel, F.M. Facca, E. Simperl, I. Toma. **Web service modeling ontology.** Semantic Web Services (2011), pp. 107-129
[63]D. Martin, M. Burstein, J. Hobbs, O. Lassila, D. McDermott, S. McIlraith, *et al.* **OWL-S: semantic markup for web services.** W3C Member submission, 22 (2004). 2007-2004
[64]J. Clark. **Xsl Transformations (xslt), World Wide Web Consortium (W3C).** http://www.w3.org/TR/xslt (1999)
[65]M. Hepp, A. Radinger, A. Wechselberger, A. Stolz, D. Bingel, T. Irmscher, *et al.* **GoodRelations Tools and Applications.** (2009)
[66]J. Ashraf, R. Cyganiak, S. O'Riain, M. Hadzic. **Open eBusiness Ontology Usage: Investigating Community Implementation of GoodRelations.** (2011)
[67]C. Fürber, M. Hepp. **Using SPARQL and SPIN for Data Quality Management on the Semantic Web.** (2010), pp. 35-46
[68]**OWL-S API.** https://code.google.com/p/owl-s/ (2015)
[69]**OWL-S API.** https://github.com/nicholasdelrio/visko/tree/master/visko/owls-api-3.0 (2015)