A Cloud based Information Repository for Bridge Monitoring Applications

Seongwoon Jeong*[a], Yilan Zhang[b], Rui Hou[b], Jerome P. Lynch[b], Hoon Sohn[c], Kincho H. Law[a]
[a]Dept. of Civil & Environmental Engineering, Stanford University, Stanford, CA, USA 94305;
[b]Dept. of Civil & Environmental Engineering, University of Michigan, Ann Arbor, MI, USA 48109;
[c]Dept. of Civil & Environmental Engineering, KAIST, Daejeon 305-701, Republic of Korea;

ABSTRACT

This paper describes an information repository to support bridge monitoring applications on a cloud computing platform. Bridge monitoring, with instrumentation of sensors in particular, collects significant amount of data. In addition to sensor data, a wide variety of information such as bridge geometry, analysis model and sensor description need to be stored. Data management plays an important role to facilitate data utilization and data sharing. While bridge information modeling (BrIM) technologies and standards have been proposed and they provide a means to enable integration and facilitate interoperability, current BrIM standards support mostly the information about bridge geometry. In this study, we extend the BrIM schema to include analysis models and sensor information. Specifically, using the OpenBrIM standards as the base, we draw on CSI Bridge, a commercial software widely used for bridge analysis and design, and SensorML, a standard schema for sensor definition, to define the data entities necessary for bridge monitoring applications. NoSQL database systems are employed for data repository. Cloud service infrastructure is deployed to enhance scalability, flexibility and accessibility of the data management system. The data model and systems are tested using the bridge model and the sensor data collected at the Telegraph Road Bridge, Monroe, Michigan.

Keywords: Bridge information modeling, OpenBrIM, SensorML, cloud service, NoSQL database

1. INTRODUCTION

Sensor and monitoring technologies have been deployed to support bridge management. Sensors are instrumented to collect structural response data (such as acceleration, strain, and displacement) which are then used to supplement manual inspections. Such quantitative measurements provide valuable information to assess the structural condition of a bridge. The emergence of sensor technologies has led to broad implementation of bridge monitoring systems world wide [1-4]. While sensor data can provide valuable information about the bridge condition, such data by itself is not sufficient for bridge monitoring applications. Additional information, such as bridge geometry, material properties and analysis models, is needed to enhance the use of the sensor data for long term bridge management purposes. An information management system that allows the integration of a wide variety of data and supports data sharing among a broad spectrum of bridge monitoring and management applications is a critical component of a bridge monitoring system.

It has been well recognized that information models and interoperability standards play an important role in integrated projects that involve diverse project participants, system and practices [5-7]. In the building industry, building information modeling (BIM) technologies have emerged as the de facto standards to facilitate information sharing and are widely supported by many analysis and design software [8]. The success of BIM has led to efforts towards developing open standards for bridge information modeling (BrIM) [9,10]. Similar to BIM, the goal of BrIM standards is to establish an integrated schema for bridge structure to capture information throughout the project lifecycle including planning, designing, construction and management [11]. However, current BrIM standards include very few data entities that are mostly about the geometry of a bridge, and they are not sufficient for bridge monitoring applications.

The recent advancements of cloud computing infrastructure can facilitate effective management and use of data [12]. For cost effectiveness, cloud computing services alleviate installation and maintenance of expensive servers. Users can dynamically and incrementally acquire computing resources on an as-needed basis. Computational resources and storages can be scaled according to usage, which can be important when dealing with continuously increase of information. Research efforts have begun to utilize cloud computing platform to enable integration using building

* e-mail: swjeong3@stanford.edu

Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2016, edited by Jerome P. Lynch,
Proc. of SPIE Vol. 9803, 980313 · © 2016 SPIE · CCC code: 0277-786X/16/$18 · doi: 10.1117/12.2219064

information model [13]. Cloud computing services can also have great potential to support bridge monitoring and management applications where a significant amount of data needs to be managed.

In this study, we investigate a cloud based information repository for bridge monitoring and management. The repository is designed to store not only the geometry data, but also analysis models and sensor data. We use the data model of OpenBrIM standards as the base. We extend the current OpenBrIM schema to capture entities needed to represent analysis models (usable by tools such as CSI Bridge [14]) and to employ standards (such as SensorML) for representing sensor description and sensor data. Cloud computing service provided by Microsoft AZURE and Apache Cassandra database system are employed to facilitate data storage management and retrieval of the bridge information. The bridge model and sensor data collected from the Telegraph Road Bridge located at Monroe, Michigan are employed to assess the effectiveness of the cloud-based information repository system.

2. BRIDGE INFOMRATION MODELING FOR SHM

This section briefly reviews the current development of OpenBrIM (version 3.0) [10]. OpenBrIM (version 3.0) uses an object-oriented parametric markup language, ParamML, as the basic syntax to facilitate interoperability as well as to support parametric design [10]. For the OpenBrIM standards, a bridge structure can be described as a collection of hierarchical objects and a set of parameters corresponding to each object. An object (represented by an "*O*" tag in ParamML) describes either a physical or a conceptual entity, while a parameter (represented by a "*P*" tag in ParamML) describes an attribute of an object [15]. For example, a rectangular beam entity can be defined as an object, while its width, height, and length are defined as parameters. OpenBrIM defines a collection of object types with particular emphasis on the geometric representation of a bridge. For example, object types such as *Point*, *Surface*, *Volume* and *Line* are defined to describe the three-dimensional geometry of a bridge element. OpenBrIM also includes conceptual object entities such as *Project* (as the root object for the bridge model), *Group* (for aggregation of physical or conceptual objects), and *Design Code* (for defining criteria for the *Project*). In addition, an object can inherit the attributes and data of another object by using "*Extends*" attributes. Detailed definitions of the objects and their parameters currently specified in the OpenBrIM standards can be found in [11,15].

As noted, the data model of OpenBrIM currently focuses on the geometric definition of bridge elements. Although OpenBrIM includes a few object types for analysis purpose (see Table 1), they are insufficient for defining a finite element model. To create a finite element model, commercial FEA software tools often involve a complex collection of entities. For an example of the Telegraph Road Bridge (a steel girder bridge in Monroe, MI) as shown in Figure 1, the finite element model created using CSI Bridge, one of the most widely used FEA software for bridge analysis and design, consists of more than fifty tables as shown in Figure 2. Each table contains many attributes or parameters that are needed for analysis purpose. To further extend OpenBrIM for monitoring application, additional object types are needed to describe the sensor and the collected sensor information.

Figure 1. Side view of Telegraph Road Bridge, Monroe, MI

Table 1. Types of FEA object defined in OpenBrIM [15]

FEA object type	Parameters
FEA Node	X-Coordinate, Y-Coordinate, Z-Coordinate, X-dir Translational Fixity, Y-dir Translational Fixity, Z-dir Translational Fixity, X-axis Rotational Fixity, Y-axis Rotational Fixity, Z-axis Rotational Fixity,
FEA Line	Node1, Node2, Beta Angle, Section
FEA Surface	Node1, Node2, Node3, Node4, Thickness, Material
FEA Analysis Case	Weight Factor
FEA Nodal Load	Node, Analysis Case, Fx, Fy, Fz, Mx, My, Mz
FEA Result Combination	-
FEA Result Combination Item	Case, Factor

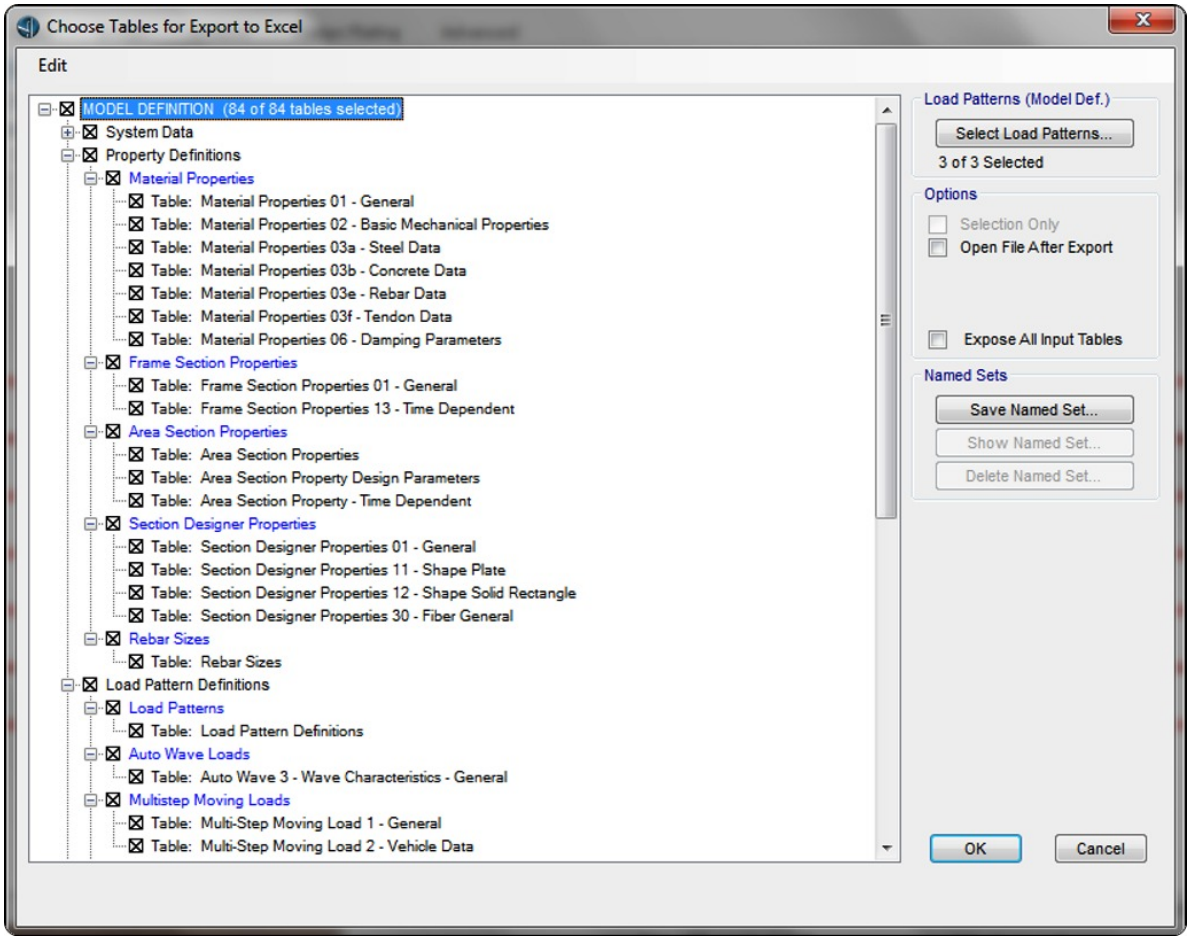


Figure 2. A partial list of CSI Bridge data table representing the finite element model of the Telegraph Road Bridge [14]

2.1 A bridge analysis information model

In this section, we describe the basic data schema that is needed for describing a practical finite element analysis (FEA) model for a bridge structure. We use the Telegraph Road Bridge shown in Figure 1 as the target structure. Furthermore, we examine the object entities that are needed by a finite element tool, CSI Bridge (Computer & Structure, Inc.), to conduct an analysis [14].

Since OpenBrIM uses eXtensible Markup Language (XML) based syntax, the current data model can be easily extended and enriched with new entities and additional information by adding user-defined object types and parameters. As for the analysis entities that are already defined in the OpenBrIM standards along with sufficient parameter definitions, we use the entities without changes. For example, CSI Bridge's data table named "Joint Load – Forces" can be added to the *FEA Nodal Load* object defined in OpenBrIM as shown in Figure 3.

For the analysis entities that are defined in the OpenBrIM standards with limited parameter definitions, we add user-defined parameters to describe all the attributes that are needed for representing the analysis model of CSI Bridge. As shown in Figure 4, although the OpenBrIM includes *Material* object describing material properties, the current object definition of *Material* object includes insufficient parameters. To enrich the current *Material* object, we add parameters such as *SymType* (directional symmetry type), *TempDepend* (temperature dependency) and other attributes which are required to describe a CSI Bridge's analysis model.

For the data entities that are required for CSI Bridge, but are not included in OpenBrIM standards, we define new object types along with necessary parameters. As an example, the vehicle lane definition needs to be defined to conduct moving load analysis, while OpenBrIM include no data entity to define the lane definition. To add the vehicle lane data entity to the current OpenBrIM standards, an additional object type named *Lane* is defined together with its parameters such as *Station*, *Width*, and *Offset* as shown in Figure 5. The summary of the data entities that are added to the current OpenBrIM is shown in Figure 6.

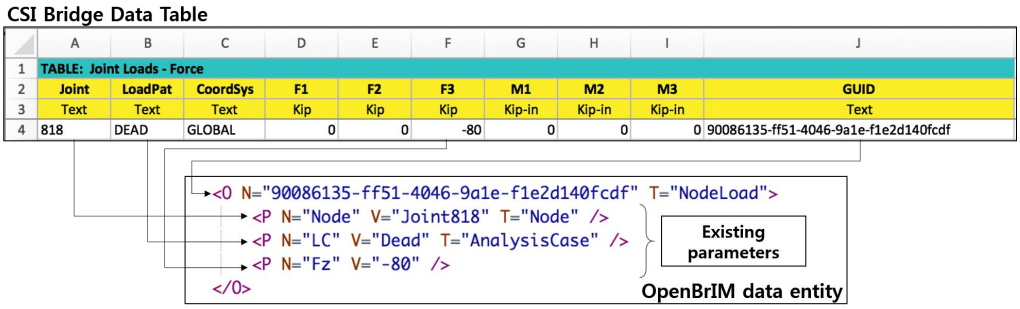


Figure 3. OpenBrIM data entity: existing object and parameters

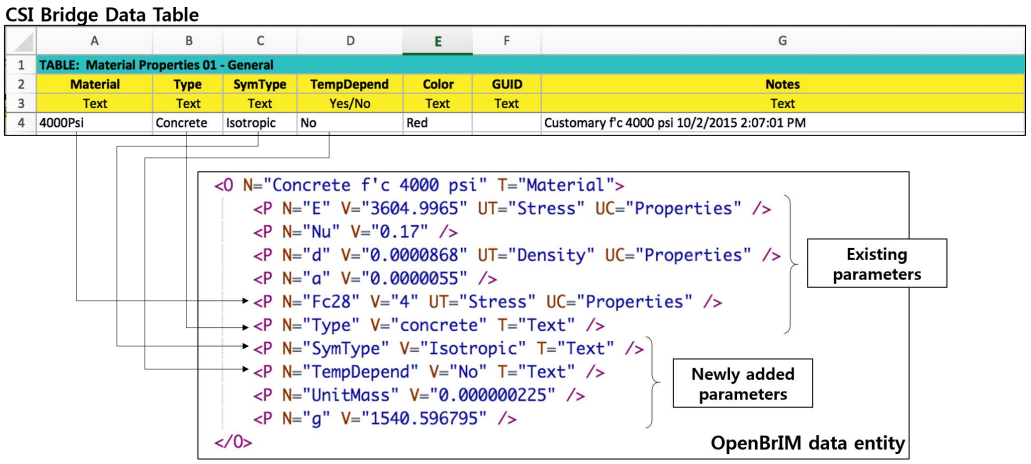


Figure 4. Enriched OpenBrIM data entity: existing object and newly added parameters

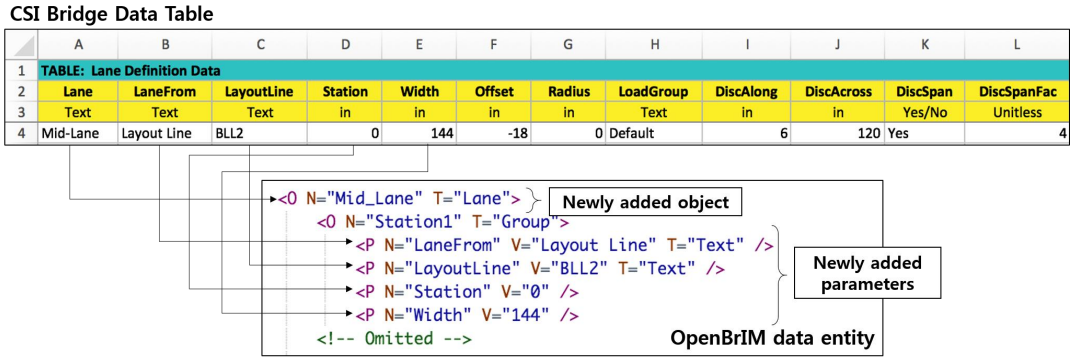


Figure 5. Enriched OpenBrIM data entity: newly added object and parameters

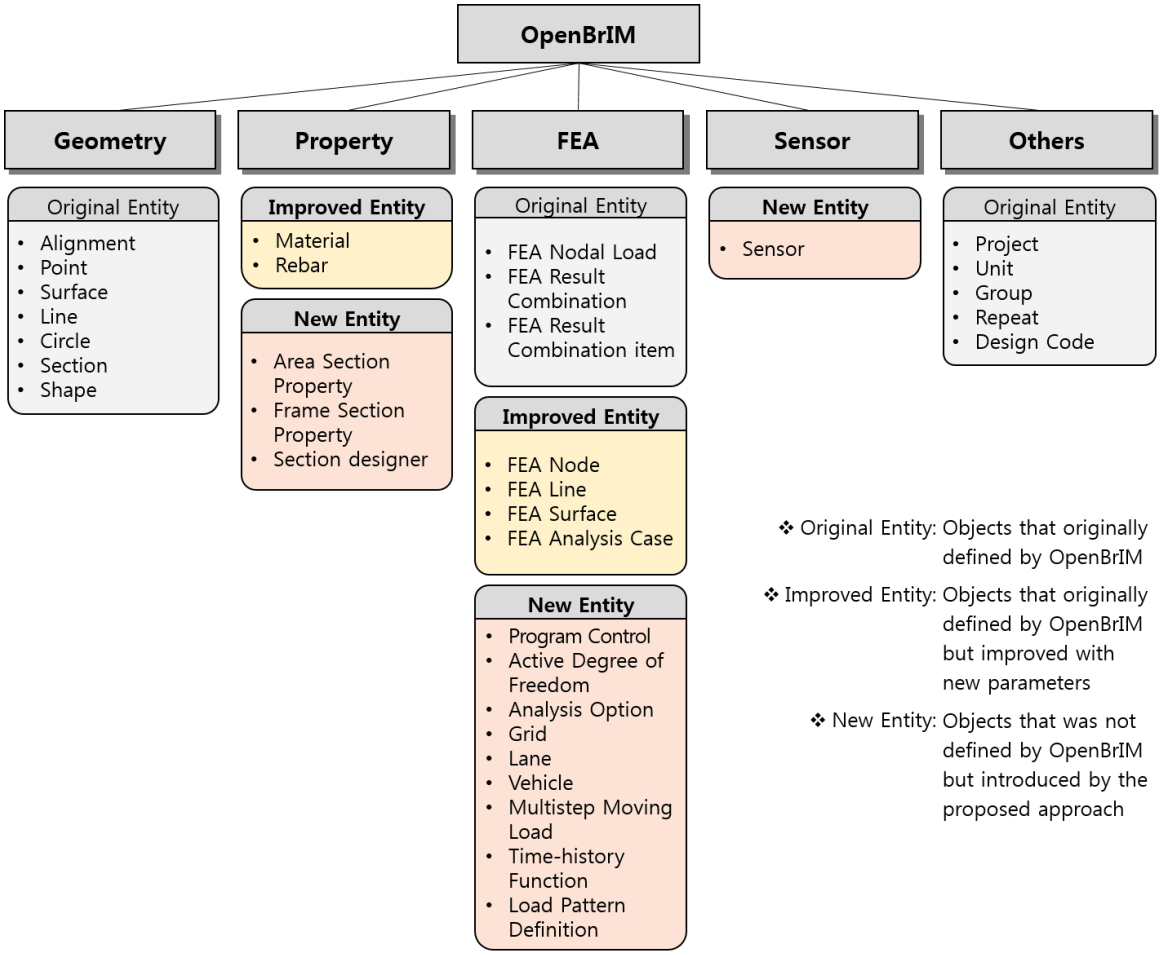


Figure 6. Enriched OpenBrIM data model with new entities representing finite element model and sensor entities

2.2 Sensor Integration model

There are several standards that have been developed to support identification, discovery, access, utilization, and data exchange within and across systems [16,17]. Sensor Web Enablement (SWE) suite provided by the Open Geospatial Consortium (OGC) is one of the most widely used suites by the sensor web community [16]. The SWE includes relevant standards to describe metadata about the sensors as well as to model the services provided by the sensors [18]. For

example, OGC's Sensor Model Language (SensorML) provides XML-based standards data format to describe sensor metadata as well as the processes and processing components associated with the sensors [19]. OGC's Observations and Measurements (O&M) provides XML-based model to describe phenomenon observed by the sensors and data measured by the sensors [20]. OGC's Sensor Observation Service (SOS) provides a Web service interface to support retrieval of sensor metadata and observation data [21].

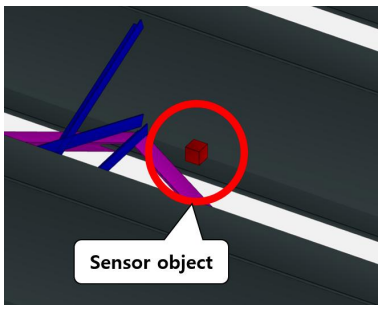
The metadata provided by SensorML includes many attributes that can be utilized to describe the configuration of the sensor in a bridge monitoring system. For structural monitoring applications, we take advantage of the sensor metadata provided by SensorML. A new object definition "*Sensor*" is defined along with its parameters describing the attributes of the sensor object. The attributes that we add to the current BrIM schema for structural monitoring applications are summarized as shown in Table 2 [19,22]. Figure 7(a) shows a snippet of an OpenBrIM document containing a sensor object *u43ch0*, which includes attributes such as *output* ("Strain"), *install* ("2014-01-01 00:00:00"), and *remove* ("None"). The geometry of sensor is defined by a child *Volume* object and described in the 3-dimensional workspace of OpenBrIM Viewer as shown in Figure 7(b). In addition, the hierarchical data structure of OpenBrIM can describe the relationship between the sensor object *u43ch0* and its parent object *girder2* (i.e. the sensor *u43ch0* belongs to *girder2* object), which enables meaningful query such as "finding all sensor objects belong to *girder2*".

Table 2. Attributes of sensor object [19,22]

Category	Attributes
System description	sensor ID, sensor name, description, keywords, group
Identifiers	long name, short name, model number, manufacturer
Classifiers	intended application, sensor type
Constraints	document valid time, security constraints, legal constraints
Input & output	input, output, unit of measurement, quantity definition
Location	location(text), location(section), location(coordinate)
Parameter	sampling rate
Physical properties	weight, weight unit, length, length unit, width, width unit, height, height, unit, casing material
Electrical requirements	voltage, current type, amp range, sensing range
Capabilities	sensing range, sensitivity, sample period, measurement output time
Contacts	responsible party, telephone, address
Documentation	manual
Data	data link

```
<O N="Girder2" T="Group">                                    Parent object
    <O N="Girder2_geometry" T="Group" Extends="Girder1" X="64*1" Y="-99*1" />
    <O N="u43ch0" T="Sensor" X="-100">                       Sensor object
        <P N="output" V="Strain" T="Text" />
        <P N="install" V="2014-01-01 00:00:00" T="Text" />
        <P N="remove" V="None" T="Text" />                   Sensor parameters
        <O N="u43ch0_geometry" T="Volume"
           X="64+EndSpanLength+22*6+ExpansionLength+MiddleSpanLength/2"
           Y="-95" Z="-25">                                  Sensor geometry
        <!-- omitted -->
```

(a) Snippet of OpenBrIM document containing a sensor object named *u43ch0*



(b) Sensor object described in 3-dimensional work space of the OpenBrIM Viewer

Figure 7. Sensor object described in OpenBrIM document and OpenBrIM Viewer

3. CLOUD BASED DATA MANAGEMENT SYSTEM

In this study, we employ Microsoft AZURE, which is one of the most widely utilized cloud computing services, to facilitate management and access of bridge information [23]. To set up a cloud service, we first create a Linux-based virtual machine platform (Ubuntu Server 14.04 LTS, Standard A1 Tier: 1 core, 1.75GB memory) on AZURE. Figure 8 shows a summary of the virtual machine named "eig-cloud2", which can be remotely accessed via Secure Shell (SSH) network protocol.

Although AZURE supports several data storage services such as MySQL Database (a relational data management system), MongoLab (a document-oriented database) and Redis Cache (an in-memory data structure store), Apache Cassandra database system is employed for structural monitoring in this study to ensure good performance for the management of time series data [24]. The distributed structure of Cassandra based on a ring topology supports automatic partitioning and replication, which enables scalability and availability of the database system [7,24]. Figure 9 shows a running instance of Apache Cassandra database system on the Linux virtual machine "eig-cloud2".

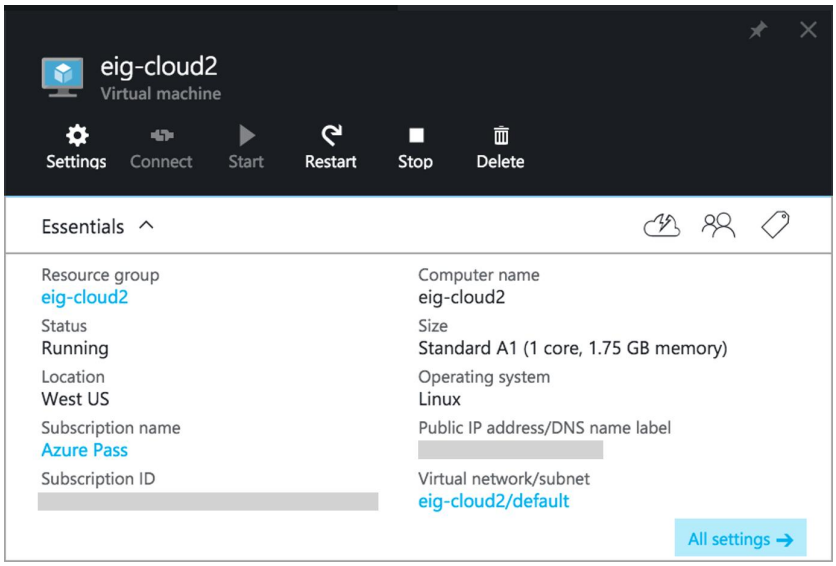


Figure 8. Virtual machine "eig-cloud2" created on the Microsoft AZURE cloud platform

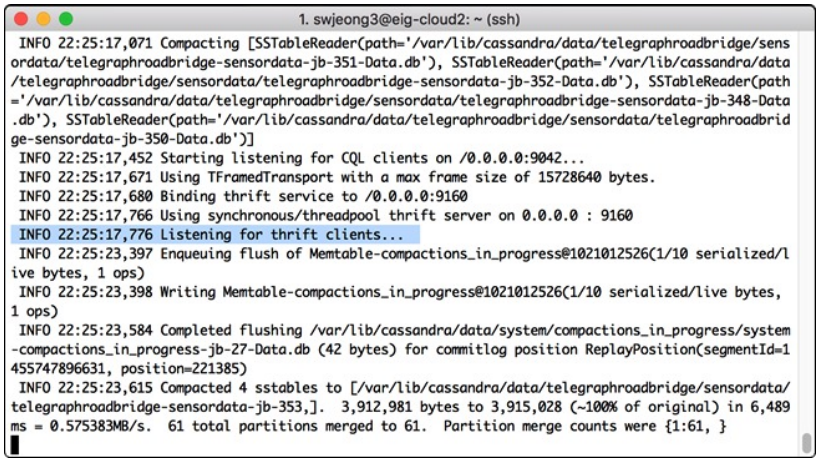


Figure 9. An instance of Cassandra database system running on the virtual machine "eig-cloud2" on the AZURE cloud

The fundamental data structure in Apache Cassandra consists of key space, column family, row key, column name, and column value [24]. The key space can be defined for the naming of a specific project or a specific bridge structure. The column family can be used to create a cluster (family) of data entities belonging to a particular group of information. For instance, the column families named *bridgeinformationmodel* and *sensordata* are defined respectively for the bridge information model and the sensor data. Each row in the *bridgeinformationmodel* family stores the data corresponding to a single object in the bridge information model. As shown in Figure 10, the column *element_id* stores the unique ID for the object, the columns *name* and *type* store the the name and the type of the object, the column *extends* stores the name of the template object, and the column *contents* stores an XML snippet describing the parameters and attributes. Since the hierarchical data structure of XML is not supported by the column-oriented data structure of Apache Cassandra, we define additional columns to record parent and children objects to represent the hierarchical relationship of the bridge information model expressed in the XML structure (such as the OpenBrIM). Figure 11 shows an example of the rows that store group objects *girder2* and its children objects *girder2geometry* and *u43ch0*, which are previously described in Figure 7(a). For the rows described in Figure 11, we use the same value for the columns *element_id* and *name* since the name of each object is uniquely defined in the information model. Although each row contains different numbers and types of data because of the complex data structure of OpenBrIM, the flexible data structure of Apache Cassandra can elegantly handle the heterogeneous data sets by allowing each row to have different collections of columns.

The column family *sensordata* is defined to store time series data measured by sensors installed on a bridge structure. Although the hashing algorithm and partitioning strategy used by Apache Cassandra is useful to handle big data with distributed computing nodes, the strategy can deteriorate the range query performance due to the distribution of sequential data. To alleviate this problem, we implement a time series data modeling scheme as shown in Figure 12 [25]. In this scheme, the partitioning key is defined according to the sensor ID and the year and month of data acquisition event, so that the sequential time series data measured by a single sensor in a single month can be stored sequentially within a disk, followed by minimal seek time for the range query [25]. To link the sensor object in the bridge information model to the corresponding sensor data, we assign the column *sensor_id* in *sensordata* column family to have the same unique ID of the corresponding sensor object in the column family *bridgeinformationmodel*. Figure 13 shows an instance of the rows in *sensordata* column family where the sensor data collected by the same sensor in the same month are stored in consecutive columns in a sorted order.

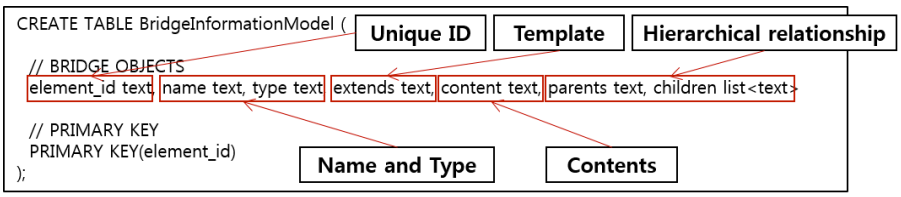


Figure 10. Data schema of *bridgeinformationmodel* column family

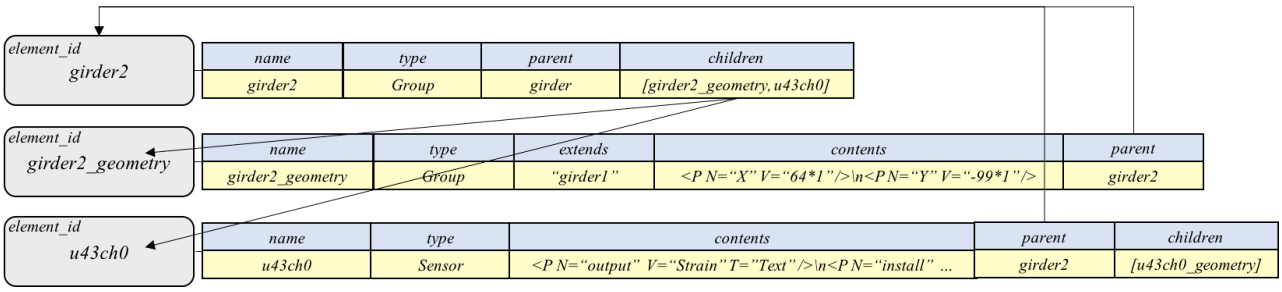


Figure 11. Example of rows stored in the *bridgeinformationmodel* column family

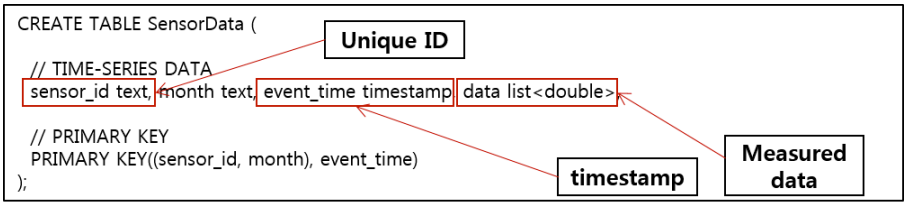


Figure 12. Data schema of *sensordata* column family

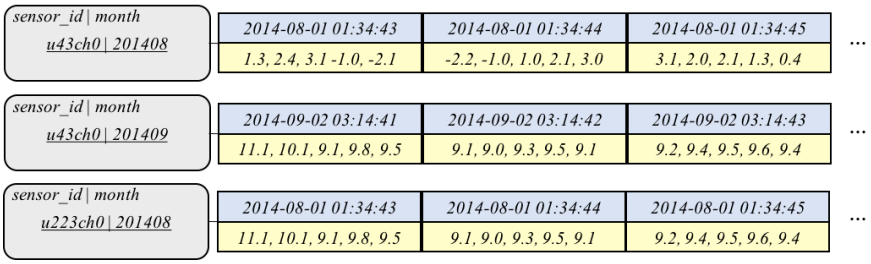


Figure 13. Example of rows stored in the *sensordata* column family

4. UTILIZATION OF THE CLOUD BASED INFORMATION REPOSITORY

In this section, the utilization of the cloud-based information repository is discussed using the data collected from the Telegraph Road Bridge (TRB), located in Monroe, Michigan. TRB is a steel girder highway bridge instrumented with a wireless sensor network [26,27]. A bridge information model is created based on a set of engineering drawings, the finite element model of the bridge, and the layout of the sensor network. The sensor data are collected from 14 accelerometers, 40 strain gauges and 6 thermistors during the period from August 2014 to February 2015. The sensor data are acquired for a one-minute time duration on every 2-hour interval. The sampling rate of the accelerometers, the strain gauges, and the thermistors are set at 200Hz, 100Hz, and 100Hz, respectively [26,27].

4.1 Bridge model retrieval

The schema for the bridge information model includes information about the geometry of the bridge and their components, the structural elements, the material properties, and the load conditions. To enable the retrieval of the finite element model, a prototype script is written in Python to extract the entities such as node, area and material properties required for the analysis. The structural data are parsed and restructured in Excel spreadsheet format, which is then imported to CSI Bridge. As an example, Figure 14 shows the Python snippet that generates a table named "Connectivity - Area" by retrieving the list of *FESurface* objects as well as the *Node* objects of the *FESurface* objects. Figure 15 shows the spreadsheet for "Connectivity - Area" information and the finite element model of the TRB. As another example, Figure 16 shows the geometry model retrieved from the database, restructured as a ParamML data document, and displayed on the OpenBrIM Viewer [10].

```
def writeSheet2(sheet, name, prop):
    itemlist = downloadSingleTuple(name, "group", "children")
    i = 4
    for item in itemlist:
        data = downloadSingleTuple(item, prop, "children")
        writeSingleTuple(sheet, i, item, data)
        i = i + 1
*** (Omitted) ***
if __name__ == '__main__':
    cluster  = Cluster(contact_points=[cloudPlatformIP])
    session  = cluster.connect(keySpaceName)
    XLSXFILE = "trb_template.xlsx"
    book = openpyxl.load_workbook(XLSXFILE)
*** (Omitted) ***
    writeSheet2(book.get_sheet_by_name("Connectivity - Area"), 'FESurfaces', 'FESurface')
```

③ Retrieve the list of "FESurface" entities

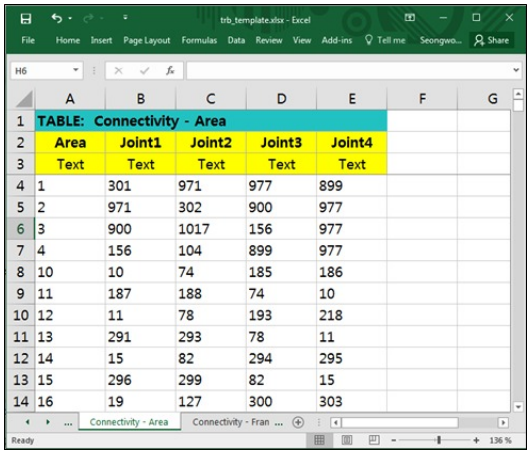
④ Retrieve "Node" information

⑤ Parse the data to a table format

① Connect to the Cassandra database in the cloud platform

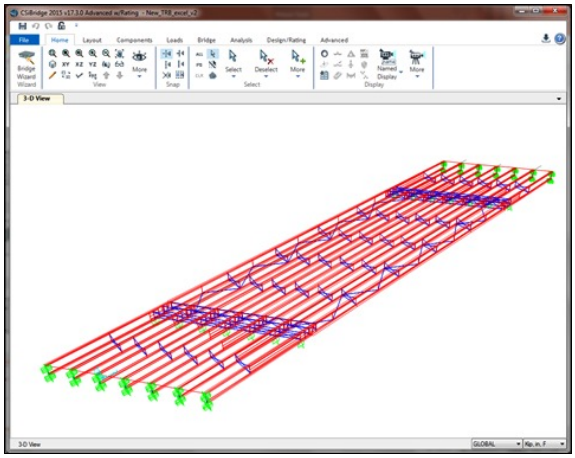
② Call the data retrieval function

Figure 14. A snippet of finite element model retrieving script



	A	B	C	D	E	F	G
1	TABLE: Connectivity - Area						
2	Area	Joint1	Joint2	Joint3	Joint4		
3	Text	Text	Text	Text	Text		
4	1	301	971	977	899		
5	2	971	302	900	977		
6	3	900	1017	156	977		
7	4	156	104	899	977		
8	10	10	74	185	186		
9	11	187	188	74	10		
10	12	11	78	193	218		
11	13	291	293	78	11		
12	14	15	82	294	295		
13	15	296	299	82	15		
14	16	19	127	300	303		

(a) Retrieved spread sheet for "Connectivity – Area"

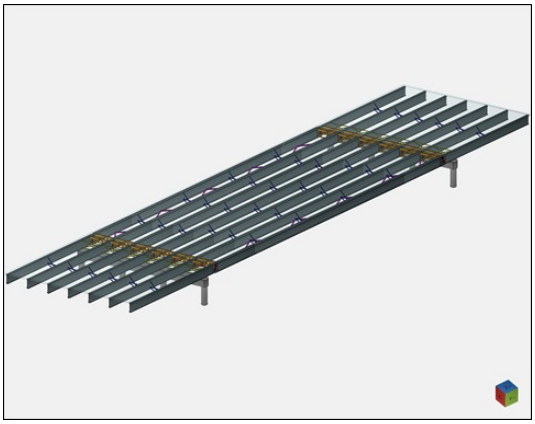


(b) Regenerated finite element model

Figure 15. Retrieved Excel spread sheets and finite element model (CSI Bridge)

```
<O N="SuperStructureTemplate" T="Group">
  <P N="ObjTypeDef" V="1" />
  <O N="IBeamType1" T="Group">
    <P N="ObjTypeDef" V="1" />
    <O N="IBeamType Param" T="Group">
      <P N="Material" V="STEEL" T="Material" D="Material" />
      <P N="Length" V="EndSpanLength-ExpansionLength/2" />
      <P N="w" V="12" />
      <P N="h" V="54" />
      <P N="TopFlangeThickness" V="0.5" />
      <P N="WebThickness" V="0.5" />
      <P N="BottomFlangeThickness" V="0.625" />
    </O>
    <O N="TopFlange" T="Surface">
      <P N="Thickness" V="TopFlangeThickness" />
      <O T="Point" X="0" Y="-w/2" Z="h/2-TopFlangeThickness/2" />
      <O T="Point" X="Length" Y="-w/2" Z="h/2-TopFlangeThickness/2" />
      <O T="Point" X="Length" Y="w/2" Z="h/2-TopFlangeThickness/2" />
      <O T="Point" X="0" Y="w/2" Z="h/2-TopFlangeThickness/2" />
    </O>
```

(a) Restructured ParamML data document



(b) Regenerated geometry model shown by OpenBrIM Viewer

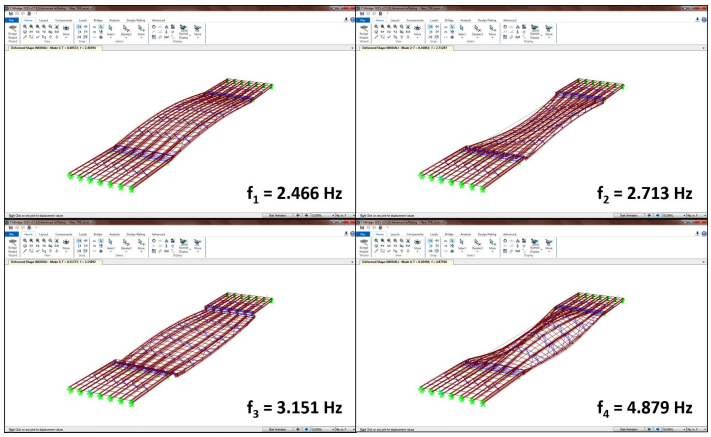
Figure 16. Part of ParamML data document and Retrieved geometry model (OpenBrIM Viewer)

4.2 Supports for structural analysis and monitoring applications

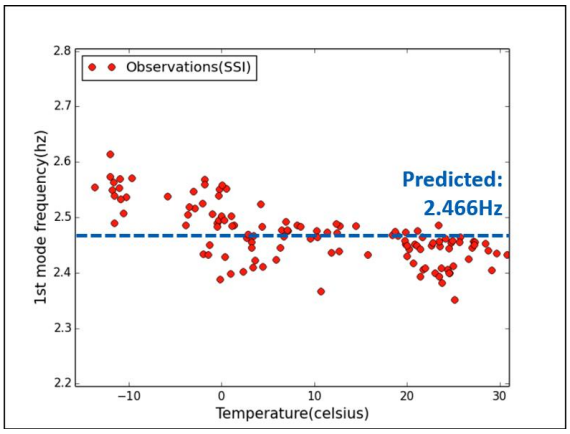
Since the bridge information model contains both the comprehensive structural analysis model and the sensor information, the data repository can be employed to support bridge analysis and monitoring functions. For demonstration purpose, modal analysis is performed using the FE model and experimental modal synthesis is conducted using the sensor data. As described in the previous example, the FE model can be retrieved as input to the CSI Bridge software. Once the FE model is retrieved and imported to CSI Bridge, a modal analysis can be performed. Figure 17(a) shows the first four natural frequencies and the corresponding mode shapes obtained from the finite element model. Furthermore, time series data, such as acceleration, can be retrieved and can be used to calculate the modal properties using Subspace Identification module (available as a MATLAB module) [28]. Figure 17(b) shows the first modal frequencies obtained using the Subspace Identification module along with the temperature measurements. The results can be used to observe the variation of measured natural frequencies with temperature changes. For instance, the measured natural frequencies of the bridge tend to be higher for cold temperature.

4.3 Information retrieval with mobile device

The data stored in the cloud-based data management system can also be easily accessed with mobile devices. The application is developed using Swift 2.0, a programming language for developing iOS application. Several swift APIs such as UIViewController, UIButton, UITableView, UIMapView and networking API are employed. On the cloud server, a HTTP server is installed to handle user request for querying sensor data from the Cassandra database. Figure 18 shows the screenshots of the interface for the mobile device and the retrieved strain data. For example, when a user enters a sensor ID and the sampling period as shown in Figure 18(b), the application sends a data retrieval request to the HTTP server. The HTTP Server then parses and sends the request to the Apache Cassandra database on the cloud platform. Once retrieved from the Cassandra database, the HTTP server sends the query results back to the mobile device as illustrated in Figure 18(c).



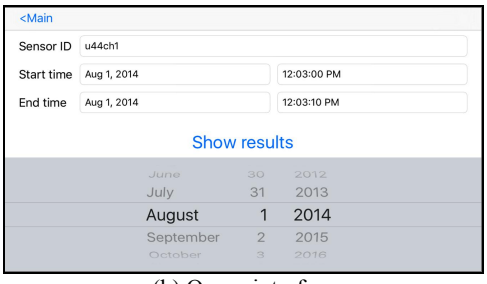
(a) Predicted natural frequencies based on FE model



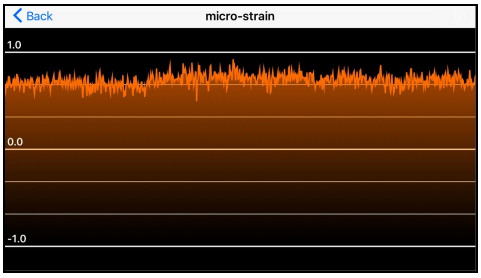
(b) Measured natural frequencies

Figure 17. Utilization of bridge information model for structural modal analysis

(a) Main view



(b) Query interface



(c) Retrieved sensor data

Figure 18. Screenshots of prototype sensor data retrieval application

5. SUMMARY AND CONCLUSION

In this study, a cloud-based bridge information repository designed for structural monitoring and management is developed using the OpenBrIM standards as the base. Additional entities that are needed to describe a finite element model are defined. Specifically, the structure of the CSI Bridge software is investigated to define the needed entities for structural analysis. Additionally, a data schema for sensors is defined to capture the sensor data. Specifically, SensorML standard is investigated to define the attributes for the sensor objects. Microsoft's AZURE cloud computing platform is employed for hosting the virtual machine supporting the bridge information repository. Apache Cassandra database is employed on the virtual machine to efficiently manage the bridge information as well as the sensor data. The cloud-based bridge information repository is demonstrated using the data collect from the Telegraph Road Bridge, located at Monroe, Michigan. The results show that the cloud-based service can effectively support integrated engineering service and enhance data sharing for bridge monitoring applications.

ACKNOWLEDGMENTS

This research is supported by a Grant No. 13SCIPA01 from Smart Civil Infrastructure Research Program funded by the Ministry of Land, Infrastructure and Transport (MOLIT) of the Korea government and the Korea Agency for Infrastructure Technology Advancement (KAIA). The research is also partially supported by the US National Science Foundation (NSF), Grant No. ECCS-1446330 to Stanford University and Grant No. CMMI-1362513 and ECCS-1446521 to the University of Michigan. The authors would like to thank the Michigan Department of Transportation (MDOT) for access to the Telegraph Road Bridge and for offering support during the installation of the wireless monitoring system. Any opinions, findings, conclusions or recommendations expressed in this paper are solely those of the authors and do not necessarily reflect the views of NSF, MOLIT, KAIA or any other organizations and collaborators.

REFERENCES

[1] Pines, D. and Aktan, A. E., "Status of structural health monitoring of long-span bridges in the United States," Progress in Structural Engineering and materials, 4(4), 372-380 (2002).
[2] Koh, H., Park, W. and Kim, H., "Recent activities on operational monitoring of long-span bridges in Korea," Proceedings of the 6th International Conference on Structural Health Monitoring of Intelligent Infrastructure, 66-82 (2013).
[3] Sumitoro, S., Matsui, Y., Kono, M., Okamoto, T. and Fujii, K., "Long span bridge health monitoring system in Japan," In 6th Annual International Symposium on NDE for Health Monitoring and Diagnostics, International Society for Optics and Photonics, 517-524 (2001).
[4] Ou, J. and Li, H., "Structural health monitoring in mainland China: review and future trends," Structural Health Monitoring, 9(3), 219-231 (2010).
[5] Cheng, J.C.P., Law, K.H., Bjornsson, H., Jones, A. and Sriram, R., "A service oriented framework for construction supply chain integration," Automation in Construction, 19(2), 245-260 (2010).
[6] Ray, S., "Interoperability standards in the semantic web," Journal of Computing in Information Science and Engineering, ASME, 2(1), 65-69 (2002).
[7] Das, M., Cheng, J. C. and Law, K. H. "An ontology-based web service framework for construction supply chain collaboration and management," Engineering, Construction and Architectural Management, 22(5), 551-572 (2015).
[8] Bernstein, H. M., Jones, S. A. and Russo, M. A., [The business value of BIM in North America: multi-year trend analysis and user ratings (2007-2012)], In Smart Market Report, McGraw-Hill Construction, Bedford, MA, 9-14 (2012).
[9] Yabuki, N., Lebegue, E., Gual, J., Shitani, T. and Zhantao, L., "International collaboration for developing the bridge product model IFC-Bridge," Proceedings of the Joint International Conference on Computing and Decision Making in Civil and Building Engineering, 1927-1936 (2006).
[10] "OpenBrIM V3," <http://openbrim.org/> (accessed 4 February 2016).
[11] Shirole, A. M., Chen, S. S. and Puckett, J. A., "Bridge information modeling for the life cycle: progress and challenges," Proceedings of 10th International Conference on Bridge and Structure Management, 313-323 (2008).
[12] Law, K.H., Cheng, J.C.P., Fruchter, R. and Sriram, R., [Cloud applications in engineering], In Encyclopedia of Cloud Computing, Murugesan, S. and Bojanova, I. (Eds.), Wiley. (in press) (2016).
[13] Kumar, B. and Cheng J. C. P., "Cloud computing and its implications for construction IT," Proceedings of 13th International Conference on Computing in Civil and Building Engineering, 30, 315 (2010).
[14] "Structural Bridge Design Software | CSI Bridge," < https://www.csiamerica.com/products/csibridge> (accessed 1 March 2016).
[15] "ParamML Author's Guide," <https://sites.google.com/a/redeqn.com/paramml-author-s-guide/> (accessed 4 February 2016).
[16] Lee, K., "IEEE 1451: A standard in support of smart transducer networking," Proceedings of IEEE Instrumentation and Measurement Technology Conference, 2, 525-528 (2000).
[17] Pschorr, J., Henson, C. A., Patni, H. K. and Sheth, A. P. "Sensor Discovery on Linked Data," 2010, <http://corescholar.libraries.wright.edu/knoesis/780> (accessed 2 March 2, 2016).
[18] Jirka, S., Bröring, A. and Foerster, T., "Handling the semantics of sensor observables within SWE discovery solutions," 2010 International Symposium on Collaborative Technologies and Systems, Art. No. 5478495, 322-329 (2010).

[19] Open Geospatial Consortium "OGC® SensorML: Model and XML Encoding Standard," 2014, <http://www.opengeospatial.org/standards/sensorml> (accessed 4 February 2016).
[20] Open Geospatial Consortium, "Observations and Measurements - XML Implementation," 2011, <http://www.opengeospatial.org/standards/om> (accessed 4 February 2016).
[21] Open Geospatial Consortium, "OGC® Sensor Observation Service Interface Standard," 2012, <http://www.opengeospatial.org/standards/sos> (accessed 4 February 2016).
[22] Jeong, S., Byun, J., Kim, D., Sohn, H., Bae, I. H. and Law, K. H. "A data management infrastructure for bridge monitoring," Proceedings of the SPIE Smart Structures/NDE Conference, Art. No. 94350P (2015).
[23] "Microsoft Azure," <https://azure.microsoft.com/> (accessed 18 February 2016).
[24] Hewitt, E., [Cassandra: the definitive guide], O'Reilly Media, Inc., 2011, <http://proquest.safaribooksonline.com/9781449399764> (accessed 4 February 2016).
[25] McFadin, P., "Getting Started with Time Series Data Modeling," 2015, <https://academy.datastax.com/demos/getting-started-time-series-data-modeling> (accessed 4 February 2016).
[26] O'Connor, S. M., Zhang, Y., Lynch, J. P., Ettouney, M. and van der Linden, G. "Automated analysis of long-term bridge behavior and health using a cyber-enabled wireless monitoring system," Proceedings of the SPIE Smart Structures/NDE Conference, Art. No. 90630Y (2014).
[27] Zhang, Y., O'Connor, S. M., van der Linden, G., Prakash, A. and Lynch, J. P. "SenStore: A Scalable Cyberinfrastructure Platform for Implementation of Data-to-Decision Frameworks for Infrastructure Health Management," Journal of Computing in Civil Engineering, 10.1061/(ASCE)CP.1943-5487.0000560, 04016012 (2016).
[28] Overschee, P. V., "Subspace Identification for Linear Systems," In MATLAB, 2002, <http://www.mathworks.com/matlabcentral/fileexchange/2290-subspace-identification-for-linear-systems> (accessed 16 December 2015)