# A Cloud-based BIM Platform for Information Collaboration

**L. Ma[a] and R. Sacks[b]**

[a]National Building Research Institute, Technion – Israel Institute of Technology, Israel
[b]Faculty of Civil and Environmental Engineering., Technion – Israel Institute of Technology, Israel
E-mail: lingma@technion.ac.il, cvsacks@technion.ac.il

**Abstract**

**The Model View Definition (MVD) approach to interoperability of BIM models requires software vendors to write export functions that can map their internal representation to the whole scope of each of many domain-specific target MVDs. However, in everyday practical use of models, a user receiving an exchanged model often requires information that is outside the scope of the exporting application. Thus the need arises to enrich the model for different uses. We propose a cloud-based mode of working in which users can freely query and enrich model objects. The data structure uses the IFC schema but enables addition of dynamically user-defined properties. The approach has been implemented using a NoSQL cloud based database and its use is illustrated with an example of semantic enrichment of a BIM model representing a damaged reinforced concrete beam reconstructed from laser scanning data.**

**Keywords –**

**Building Information Modelling; Model View Definition; Cloud based database; Information collaboration; Semantic enrichment**

## 1 Introduction

The Industry Foundation Classes (IFC) data schema has been developed to solve the information interoperability problem in the implementation of Building Information Modelling (BIM). To support exchange of BIM models across the full spectrum of architecture, engineering and construction (AEC) disciplines, IFC is designed to be generic and comprehensive in order to accommodate many different configurations, industry domains and levels of detail [1]. However, the IFC schema is not tailored to any individual processes within building construction, because in different business processes different users of building data have different information requirements, and authors of the building data will provide detail in different domains. Such discrete processes should be well defined in an Information Delivery Manual (IDM) [2], which maps the actors and information exchange requirements of any activities in the processes. Guided by the IDM, a set of Model View Definition (MVD) concepts is compiled to select a subset of the IFC schema and define allowable values at particular attributes of particular data types [3]. The MVD specifies which entities and attributes are mandatory or optional according to the needs of the specific business process.

The IDM and MVD approach has been used in subdomains like precast concrete construction [3-5], building energy analysis [6], and facility management handover [7]. A number of specialized applications have been developed for checking the compliance of IFC instance files to model view definitions [8, 9].

However, all BIM authoring applications have their own internal schema for building data modelling to suit their specific functional needs and information needs. Since no professional BIM software can support the full range of MVDs in IFC export subroutines, the need arises for tools that can supplement BIM exchanges with necessary information in a dynamic fashion according to the needs of importing applications. In addition, the file-based mode of most IFC exchanges is inadequate for dynamic data supplementation – a server solution is needed, preferably hosted in the cloud.

### 1.1 Semantic Enrichment

To address the need for supplementing information, Belsky et al. [10, 11] developed a rule-based semantic enrichment engine, which aims at upgrading a 'poorly-exported' BIM model to a 'high-standard' model that conforms to the MVD at the import end. The engine identifies undefined objects and missing object relationships in an IFC file based on encapsulated domain experts' (e.g., civil engineers) knowledge of geometrical and topological features of those objects. It automatically supplements the IFC file with information regarding objects' identification and relationships.

However, some information required by certain MVDs cannot be inferred from the geometrical and topological features the semantic enrichment engine depends on [10, 11]. Adding this missing information requires additional data entry, often in a collaborative

fashion. In addition, users may also need to add information items that are beyond the scope of the current IFC schema.

### 1.2 BIM Servers and Cloud Systems

The semantic enrichment engine does not have an interface for users to supplement information, neither within the framework of the target MVD schema nor as part of an extended schema. A similar limitation also exists in current cloud-based BIM applications (such as Autodesk A360 [12] and Trimble Connect [13]) or BIM applications that can be deployed in the cloud (such as BIMServer [14] and GRAPHISOFT BIMcloud [15]). Most of these applications only allow users to query data rather than supplement the data, because they don't expose their local schema. None of them enables extension of the current IFC schema dynamically for enriching information items that are not defined in the schema.

To address these problems, we developed a cloud based BIM platform in which users could freely query, enrich and deliver model information in a working mode coincident with the IDM & MVD concepts. The platform enables both automated rule-based enrichment and user-friendly data entry. The platform's data structure uses the IFC schema but also enables dynamic addition of user-defined properties. The platform was built upon a non-SQL cloud based database. The following sections present the main components of the BIM platform and demonstrate its application.

## 2 The BIM Platform Components

The objective of the platform is to design a *data as a service* system in the cloud for collaboration on a BIM model. A system diagram is shown in Figure 1.

There are two major components in the BIM platform: a database system (MongoDB) for storing the BIM models and an interface for communication between the user (or other BIM applications) and the BIM models for different purposes, including uploading BIM models using IFC instance files, creating models from scratch, or updating a model. The interfaces follow a standard protocol to perform those actions upon the database system using CRUD operations.

The following sections explain why such a database system provides a good alternative for storing a BIM model, and how to work on a BIM model using database operations.

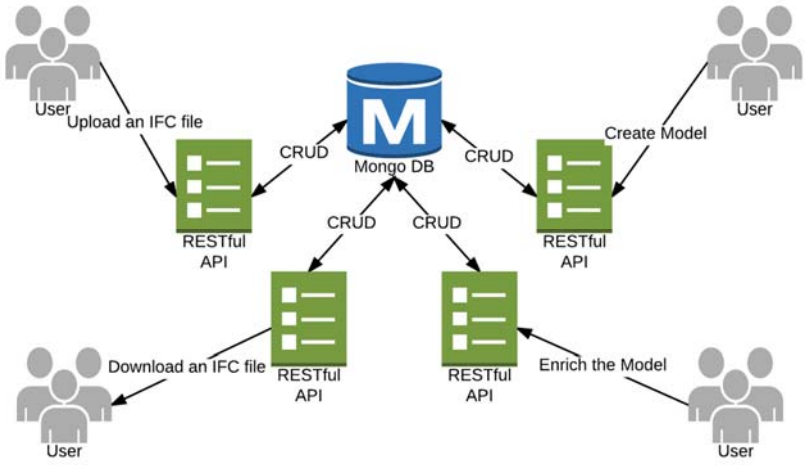



Figure 1 System Diagram of the Cloud Based BIM Platform

### 2.1 NoSQL Cloud Based Database

Database systems generally encapsulate the schema together with the data. IFC instance files, on the other hand, IFC separate the data from the underlying schema, which must be made available to any application that needs to import a BIM model stored in an IFC file. Database systems allow operations such as Create, Read, Update, and Delete (CRUD), which cannot be used directly to model data stored in IFC files.

However, IFC schema is mature in terms of representation of a building design. It can be used to model most of the information items required in subdomains, because it allows creating customized property sets, adding user defined element typing, and many other features. Hence, the IFC schema is a good

starting point for the translation from an IFC instance file to a database storage of the BIM model.

Commonly-used relational databases use predefined tables and tabular relations, and as such are inappropriate for use with the IFC schema, which is an object-oriented schema coded using EXPRESS language. NoSQL is a new data management technology specifically designed to meet the requirements of data variety [16]. It does not require a predefined data schema, which means it not only allows mapping all the data definitions in IFC schema to the database structure, but also enables addition of new entities to the data structure. This conforms to the mode of development of the IFC schema, which is continuously adopting new modelling concepts into the schema in new releases.

One of the most popular NoSQL database is MongoDB, which is used in the proposed BIM platform. It is a document oriented database, and combines the best of relational database with the innovations of NoSQL technology. It stores the data in its featured Binary JavaScript Object Notation (BSON) format. The basic atom in MongoDB is called document. All documents are stored in collections, which are analogous to the data tables in relational databases. A document is an unordered set of field and value pairs, and it always contains a field of "_id" which is a unique index. For example, a document in the database could be like this:

```
{
    name : "concrete beam",
    dimension : [240, 300, 3900],
    _id: "12312366180a4ac15c7fdsdf"
}
```

A beam modelled in an IFC file that conforms to IFC Coordination View 2.0 is shown as follows:

```
#95=IfcBeam('2LyQvT49XF5xaTest6Jq_$',#5,'M_Concrete-Rectangular Beam:300 x 600mm:300 x 600mm:104323','',$,#92,#94,$)
```

A program for serialization of IFC instance file to MongoDB documents was developed. The above data entity of the beam can be mapped to MongoDB using this program and get the document as follow:

```
{
    _id: "56672266180a4ac15c7fb483",
    line_id: 95,
    type: "IfcBeam",
    GlobalId: "2LyQvT49XF5xaTest6Jq_$",
    OwnerHistory: "56672266180a4ac15c7fb429",
    Name: "M_Concrete-Rectangular Beam:300 x 600mm:300 x 600mm:104323",
    Description: "",
    ObjectType: null,
    ObjectPlacement:"56672266180a4ac15c7fb480",
    Representation:"56672266180a4ac15c7fb482",
    Tag: null,
    HasAssociations: [
        "56672266180a4ac15c7fba8e"
        ],
    IsDefinedBy: [
        "56672266180a4ac15c7fb4a4"
        ],
    ContainedInStructure: [
        "56672266180a4ac15c7fba81"
        ]
}
```

Note that the fields like ObjectPlacement and Representation link to other documents by using the latter's _id as their value. In addition, the program can also derive the inverse attributes after completely parsing the IFC file using the IFC schema. So fields like "HasAssociations", "ContainedInStructure" and "IsDefinedBy" are also derived in this example, although they did not exist in the IFC instance file. As a result, the data in the database is richer in terms of semantics and is more human readable.

Another benefit of using MongoDB is that it uses timestamp-based concurrency control, which is a non-locking control mechanism, so that it supports real-time read that will not conflict writes and high-speed logging. It has caching and high scalability, which allows running thousands of machines with distributed data, and is well-suited for deployment in the cloud. Sets of redundant servers called "replica sets" help maintain the system availability and data integrity even if individual cloud instances are taken offline.

As a result, the NoSQL-based MongoDB system serves a very good alternative for BIM application in terms of easy-to-use and concurrent collaboration among different users.

## 2.2 A RESTful API for CRUD Operations on Model Data

An Application Programming Interface (API) is used to allow one software/service talk with the other. Another term, i.e., Hypertext Transfer Protocol (HTTP) is the standard protocol that defines how a web browser and a web server communicate with each other. For example the client sends a request to the server and the server sends back a response. As technology moves forward, network-based software architectures have been widely adopted. Hence an architecture called REpresentational State Transfer (REST) [17] was proposed to define an API that uses HTTP for a series of constrained methods, such as get, put, post, or delete. Such kinds of API are usually called RESTful APIs.

Many RESTful APIs send responses in JSON, which is both human and machine readable. Another advantage of using RESTful APIs is that those HTTP methods can be easily mapped to the CRUD operations in a database like MongoDB.

In MongoDB, a CRUD operation targets a specific collection of documents. Generally CRUD operations can be grouped as queries and data modifications. A query specifies a projection (return which fields from the documents in response), query criteria, e.g., $eq (equal), $gt (greater than), $in (matches any of the values specified in an array), etc., and a result modifier, e.g, sort the data in certain order, limit the number of documents returned. For example the query for getting all the beams in a BIM model is simply:

db.records.find( {type: “IfcBeam”} )

Another group of CRUD operations is data modification, including create, update or delete data. For updating the names of all the beams to “Type A Beam”, one only needs to run:

db.records. updateMany ( {type: “IfcBeam”}, {$set: {name: “Type A Beam”}} )

After associating the MongoDB operations to a customized RESTful API, one can perform the above two operations in a terminal as follows (it is supposed that the server is running on http://127.0.0.1):

$ curl –i http://127.0.0.1?where={“type”: “IfcBeam”}

and

$ curl –X PATCH -i http://127.0.0.1?where={“type”: “IfcBeam”} –d ‘{“name”: “Type A Beam”}’

Note that PATCH is the HTTP method that links to the Update operation in MongoDB. As shown above, by using the RESTful API associated with MongoDB, performing operations on a BIM model is straightforward.

## 3 The BIM Platform Applications

Automated reconstruction of ‘as-built’ building models from remote sensing data, including point cloud data [18] and photogrammetry/videogrammetry data [19, 20] is the focus of much ongoing research around the world. To achieve the goal of creating a semantically rich BIM model from the sensing data, collaboration between experts in computer vision and in civil engineering is essential. The former contribute to reconstruction of 3D primitives from the redundant point cloud data, while the latter help to interpret and classify the 3D primitives using their domain knowledge.

The following sections utilize an example from this area of research as a vehicle to show how the customized cloud BIM platform can facilitate such collaboration. Specifically, we use the example of reconstruction of a reinforced concrete beam model taken from a research effort at the Technion–Israel Institute of Technology to develop a system for reconstruction of BIM models for earthquake damaged buildings [21-24] using terrestrial laser scanning.

### 3.1 3D Reconstruction of Remote Sensing Data

The first application of the platform shows how to create a 3D model from scratch using a computer vision algorithm.

A specimen of a damaged beam, shown in Figure 2 (a), was scanned using a terrestrial laser scanner - Leica ScanStation C10 at the Technion. The point cloud data, shown in Figure 2 (b), was processed using the Random Sample Consensus (RANSAC) algorithm [25]. RANSAC has become one of the most popular algorithms for model fitting from a set of data containing outliers. It is used to divide the point cloud into clusters, each of which contains points attached to a plane. The clustering result of the scan is shown in Figure 2 (c), in which each cluster of points is coloured differently.

In addition, we can define the edges as places where the geometry of the scene changes sharply/abruptly. Such features can be detected as points having high curvature values across a certain cluster of points. Those boundary points form polygons that represent 3D primitives in the scene.

For each polygon, the program sends a PUT request to insert documents for storing the vertices. For example, for a point with coordinates of (1400,120,150), a program will send a request as follows,

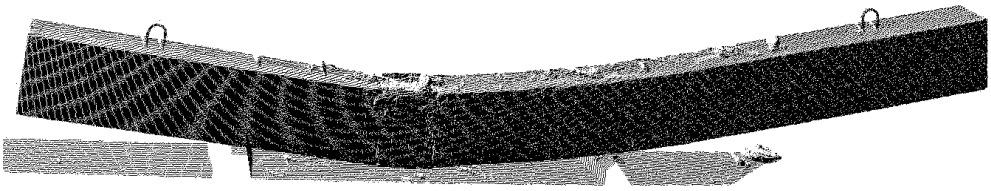
```
$ curl -d ‘[{
        “type”: " IfcCartesianPoint",
        “Coordinates”: [1400,120,150],
        “Dim”: 3,
        “LayerAssignment”:
        “56672266180a4ac15c7fb481”,
        “StyledByItem”: “56672266180a4ac15c7fbdfg”
        }]’
```

Note that, the field of “Dim” is a derived attribute calculated by the length of the array type field of “Coordinates”; “LayerAssignment” and “StyledByItem” are inverse attribute links to other documents for the purposes of grouping shapes and assigning presentation information (e.g., color) respectively. None of them is stored in an IFC instance file, but, for a rich sematic representation, they will be explicitly specified in the database documents by linking to other documents’ object ids, which are accessible after insertion.

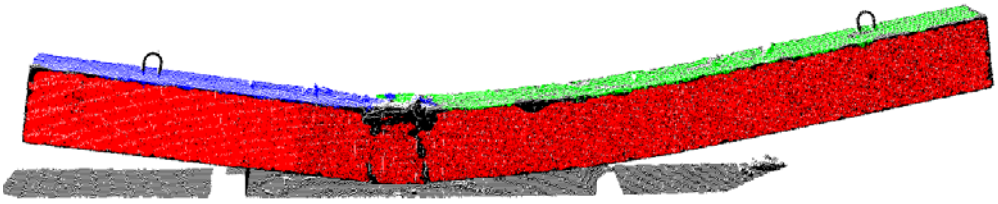
After defining the coordinates of vertices, a document for representing a polygon will be inserted with the object ids (like “56672266180a4ac15c7fb481”) of these vertices using a command such as:

```
$ curl -d '[{
    "type": " IfcPolyLoop",
    "Polygon":  ["56672266180a4ac15c7fb481",
    "56672266180a4ac15c7fb482",
    "56672266180a4ac15c7fb483",
    "56672266180a4ac15c7fb484"],
    "LayerAssignment":
    "56672266180a4ac15c7fb481",
    "StyledByItem": "56672266180a4ac15c7fbdfg"
    }]'
```

(a) The damaged beam



(b) The scan of the beam



(c) The segmentation result of the scan

Figure 2 Scan of a damaged beam and the point cloud segmentation result

Usually, a polygon organizes the vertices in a certain sequence that makes the polygon's normal direction aligned with the face normal direction, which shows where the material is. In this case the value of the orientation of the face is defined as TRUE, otherwise is FALSE meaning that the polygon is reversed when used in the face. Hence the definition of the geometric representation is unambiguous. Suppose that the object id of the polygon is "56672266180a4ac15c7fb4db", then the insertion of the face geometry can be done as follows,

```
$ curl -d '[{
    "type": " IfcFaceOuterBound",
    "Bound": "56672266180a4ac15c7fb4db",
    "Orientation", TRUE,
    "LayerAssignment":
    "56672266180a4ac15c7fb481",
    "StyledByItem": "56672266180a4ac15c7fbdfg"
    }]'
```

In the IFC schema, besides the basic geometric information, a face can also be associated with texture maps through an inverse attribute "HasTextureMaps". As a result, adding a face can be done using the following command,

```
$ curl -d '[{
    "type": " IfcFace",
    "Bound": "56672266180a4ac15c7dsedb",
    "HasTextureMaps", TRUE,
    "LayerAssignment":
    "56672266180a4ac15c7fb481",
    "StyledByItem": "56672266180a4ac15c7fbdfg"
    }]'
```

The 3D reconstruction work can only derive the segmented faces, so only the geometric information of

the faces is added to the model in the database at this stage. Generating a semantically rich BIM model requires the input of the civil engineers, not only computer vision.

### 3.2 Semantic Enrichment of a BIM model

The cloud-based BIM platform enables collaborative semantic enrichment of BIM models. The second application shows how to enrich the BIM model generated in the first application to meet the requirements of modelling damaged building components.

The normal direction of each face can be derived by computing the cross product the vectors of any two of its non-parallel edges, and can be added to the model using the following command:

```
$ curl -d '[{
      “type”: " IfcDirection",
      “Axis”: [1,0,0],
      "Dim", 3
      }]'
```

Thus pairs of perpendicular faces can be identified by the feature that the dot product of their normal directions equals zero. Each pair of perpendicular faces can be used to define a local coordinate system. The axes can be defined using:

```
$ curl -d '[{
      "type": " IfcAxis2Placement3D",
      "Location”: “56672266180a4ac15c7fbtbc”,
      “Dim”: 3,
      "Axis”: "56672266180a4ac15c7dsedb",
      "RefDirection": “56672266180a4ac15c7dsedb”,
      “P”: “56672266180a4ac15c7dsyuj”,
      “LayerAssignment”:
      “56672266180a4ac15c7fb481”,
      “StyledByItem”: “56672266180a4ac15c7fbdfg”
      }]'
```

where “Location” links to an origin (the end point of the face-face intersection line segment), “Axis” and “RefDirection” are the normal axes of the two faces, and “P” is a derived attribute calculated by the cross product of the two axes.

In this domain of concrete beams and columns, most of the building elements can be represented using extruded area solids with a rectangular profile. Hence, after aligning the point cloud to the local coordinate system of a pair of the faces, one can extrude a rectangle along the point cloud segments until all relevant points cover the visible faces of the extruded shape. The result is shown in Figure 3 (a). Insertion of this shape representation of object can be done using,

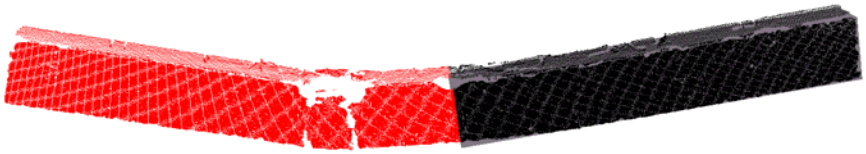
```
$ curl -d '[{
      "type": " IfcExtrudedAreaSolid",
      “Dim”: 3,
      "SweptArea”: “66672266180a4ac15c7fbtbc”,
      "Position”: "76672266180a4ac15c7dsedb",
      "ExtrudedDirection":
      “96672266180a4ac15c7dsedb”,
      “Depth”: “600”,
      “LayerAssignment”:
      “56672266180a4ac15c7fb481”,
      “StyledByItem”: “56672266180a4ac15c7fbdfg”
      }]'
```

where the “SweptArea” links to a profile definition (a rectangle in this case), and the “Position” links to the local coordinates system defined in previous step.

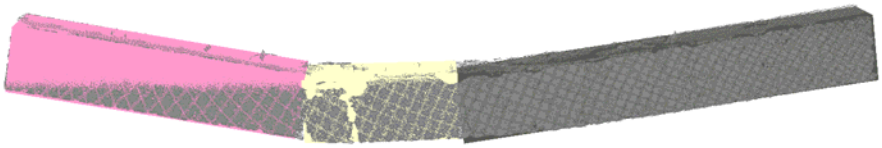
Another two solids can be derived and added to the model in a similar way, and the result is shown in Figure 3 (b). When a site engineer/damage inspector sees the three shapes, they will recognize that these shapes should be the solid segments that belong to a single beam from the original building, whose shape before the damage is as shown in Figure 3 (c). As a result, the correct representation of the beam should aggregate the three extruded area solids into one shape (using IfcProductDefinitionShape) and present this shape as a single beam using the following command:

```
$ curl -d '[{
      "type": " IfcBeam",
      “GlobalId”: “3lB8HeqWX8Tfa2Zdze9mzo”,
      “OwnerHistory”:
      “3lB8HeqWX8Tfa2Zdze9m12”,
      "Name”: “Beam”,
      "Description”: “An earthquake damaged beam”,
      "ObjectType": "Damaged beam”,
      “ObjectPlacement”:
      “76672266180a4ac15c7dsedb”,
      “Representation”:
      “546ds266180a4ac15c7fbdtt”,
      “PredefinedType”: BEAM
      }]'
```

In this way the model was progressively enriched for the purpose of representing a damaged beam in a suitable shape representation. This illustrates the ways in which the cloud-based platform can be used to dynamically compile a building model.



(a) Adjust the 3D model by fitting an extruded area solid



(b) Progressively create three solids in the scene

(c) The original as-built beam model

Figure 3 Enrichment of the Beam Model

## 4 Conclusions

The standard IDM and MVD approach was devised to specify the information exchange model schema for specific business processes in subdomains. However, BIM authoring applications serve a variety of purposes (architectural design, structural analysis, production management, etc.) and have specialized internal data schema. They therefore require extensive mapping functions in order to export customized instance models for export. To address this problem, an alternative cloud based approach to enriching and sharing BIM models in a collaborative way was proposed and tested.

The schema free feature of NoSQL allows users across different domains to share and collaborate on a single but comprehensive BIM model, where they can all contribute to the model enrichment. Instance models prepared according to different MVDs can be merged into this single model.

The major advantages of this platform are:

- Cloud based storage enables information sharing in a comprehensive model.
- The NoSQL databased is not limited to any predefined data schema, as would be required if a relational database was used.
- There is no redundancy of information storage as there is in file systems where new versions of files are saved periodically.
- The RESTful API is a lightweight, standard-based, and platform/language-independent API, which makes execution of operations on BIM model easy.

The cloud-based system developed can be used as the repository for semantic enrichment such as that implemented by the SeeBIM system, but it can also be accessed conveniently by other applications and BIM authoring tools.

## References

[1] BuildingSmart. *Industry Foundation Classes Release 4 (IFC4)*. 2013 2013.

[2] Wix, J. and J. Karlshoej, Information delivery manual: Guide to components and development methods. *BuildingSMART International*, 2010.

[3] Venugopal, M., et al., Semantics of model views for information exchanges using the industry foundation class schema. *Advanced Engineering Informatics*, 2012. 26(2): p. 411-428.

[4] Sacks, R., et al., The Rosewood experiment — Building information modeling and interoperability for architectural precast facades. *Automation in Construction*, 2010. 19(4): p. 419-432.

[5] Eastman, C.M., et al., Exchange Model and Exchange Object Concepts for Implementation of National BIM Standards. *Journal of Computing in Civil Engineering*, 2010. 24(1): p. 25-34.
[6] Jeong, W., et al., Translating building information modeling to building energy modeling using model view definition. *TheScientificWorldJournal*, 2014. 2014: p. 638276-638276.
[7] East, E.W., N. Nisbet, and T. Liebich, Facility Management Handover Model View. *Journal of Computing in Civil Engineering*, 2013. 27(1): p. 61-67.
[8] Oh, M., et al., Integrated system for BIM-based collaborative design. *Automation in Construction*, 2015. 58: p. 196-206.
[9] Zhang, C., J. Beetz, and M. Weise, Interoperable validation for IFC building models using open standards. *Journal of Information Technology in Construction*, 2015. 20: p. 24-39.
[10] Belsky, M., R. Sacks, and I. Brilakis, Semantic Enrichment for Building Information Modeling. *Computer-Aided Civil and Infrastructure Engineering*, 2015. 31(4): p. 261-274.
[11] Belsky, M., R. Sacks, and I. Brilakis. A framework for semantic enrichment of IFC building models. in *Proceedings of the 30th CIB W78 International Conference*. 2013. Beijing, China.
[12] *Autodesk A360*. 2016; Available from: https://a360.autodesk.com/.
[13] *Trimble Connect*. 2016; Available from: http://connect.trimble.com/.
[14] Beetz, J., et al. BIMserver. org–An open source IFC model server. in *Proceedings of the CIP W78 conference*. 2010.
[15] *GRAPHISOFT BIMcloud*. 2016; Available from: http://www.graphisoft.com/bimcloud/overview/.
[16] Han, J., et al. Survey on NoSQL database. in *6th International Conference on Pervasive Computing and Applications (ICPCA)*. 2011. IEEE.
[17] Fielding, R.T., *Architectural styles and the design of network-based software architectures*. 2000, University of California, Irvine.
[18] Tang, P., et al., Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques. *Automation in Construction*, 2010. 19(7): p. 829-843.
[19] Brilakis, I., et al., Toward automated generation of parametric BIMs based on hybrid video and laser scanning data. *Advanced Engineering Informatics*, 2010. 24(4): p. 456-465.
[20] Fathi, H. and I. Brilakis, A videogrammetric as-built data collection method for digital fabrication of sheet metal roof panels. *Advanced Engineering Informatics*, 2013. 27(4): p. 466-476.
[21] Ma, L., et al., A Computational Procedure for Generating Specimens of BIM and Point Cloud Data for Building Change Detection, in *Computing in Civil Engineering 2015*. 2015, American Society of Civil Engineers. p. 684-691.
[22] Ma, L., et al., Preparation of Synthetic As-Damaged Models for Post-Earthquake BIM Reconstruction Research. *Journal of Computing in Civil Engineering*, 2015: p. 04015032.
[23] Ma, L., R. Sacks, and R. Zeibak-Shini, Information modeling of earthquake-damaged reinforced concrete structures. *Advanced Engineering Informatics*, 2015. 21(3): p. 396–407.
[24] Zeibak-Shini, R., L. Ma, and R. Sacks, *Mapping the Structural Frame of a Damaged Reinforced Concrete Building using As-Damaged Scans and As-Built BIM*, in *Proceedings of the 32nd CIB W78 International Conference*. 2015.
[25] Fischler, M.A. and R.C. Bolles, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 1981. 24(6): p. 381-395.