



# An Object-Oriented Database for Managing Building Modeling Components and Metadata

## Preprint

N. Long, K. Fleming, and L. Brackney

*Presented at Building Simulation 2011  
Sydney, Australia  
November 14-16, 2011*

**NREL is a national laboratory of the U.S. Department of Energy, Office of Energy Efficiency & Renewable Energy, operated by the Alliance for Sustainable Energy, LLC.**

**Conference Paper**  
NREL/CP-5500-51835  
December 2011

Contract No. DE-AC36-08GO28308

## NOTICE

The submitted manuscript has been offered by an employee of the Alliance for Sustainable Energy, LLC (Alliance), a contractor of the US Government under Contract No. DE-AC36-08GO28308. Accordingly, the US Government and Alliance retain a nonexclusive royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for US Government purposes.

This report was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or any agency thereof.

Available electronically at <http://www.osti.gov/bridge>

Available for a processing fee to U.S. Department of Energy and its contractors, in paper, from:

U.S. Department of Energy  
Office of Scientific and Technical Information

P.O. Box 62  
Oak Ridge, TN 37831-0062  
phone: 865.576.8401  
fax: 865.576.5728  
email: <mailto:reports@adonis.osti.gov>

Available for sale to the public, in paper, from:

U.S. Department of Commerce  
National Technical Information Service  
5285 Port Royal Road  
Springfield, VA 22161  
phone: 800.553.6847  
fax: 703.605.6900  
email: [orders@ntis.fedworld.gov](mailto:orders@ntis.fedworld.gov)  
online ordering: <http://www.ntis.gov/help/ordermethods.aspx>

Cover Photos: (left to right) PIX 16416, PIX 17423, PIX 16560, PIX 17613, PIX 17436, PIX 17721



Printed on paper containing at least 50% wastepaper, including 10% post consumer waste.

# AN OBJECT-ORIENTED DATABASE FOR MANAGING BUILDING MODELING COMPONENTS AND METADATA

Nicholas Long, Katherine Fleming, and Larry Brackney  
National Renewable Energy Laboratory, Golden, CO USA

## ABSTRACT

Building simulation enables users to explore and evaluate multiple building designs. When tools for optimization, parametrics, and uncertainty analysis are combined with analysis engines, the sheer number of discrete simulation datasets makes it difficult to keep track of the inputs. The integrity of the input data is critical to designers, engineers, and researchers for code compliance, validation, and building commissioning long after the simulations are finished. This paper discusses an application that stores inputs needed for building energy modeling in a searchable, indexable, flexible, and scalable database to help address the problem of managing simulation input data.

## INTRODUCTION

Over the past decade, traditional relational database management systems (RDBMS) have shown their limitations in a Web-based world where scalability and redundancy across multiple servers are of interest. These simulation databases will be easier to implement, maintain, and update if they are designed with a flexible object-oriented database model that stores data in a single collection rather than across multiple tables and rows. Nonrelational database structures are more agile; however, they lack the well-known table-join structure of the conventional relational databases. Although not ideal for all solutions (such as heavily transaction-dependent accounting applications where the timing and completion of data manipulation are critical), nonrelational database systems are well-suited for handling the size and complex relationships associated with building data. We discuss the benefits and difficulties of implementing a nonrelational database management system to index several hundred thousand simulation components in a Web-based framework.

The Building Component Library (BCL) is a new Web resource and database being developed at the National Renewable Energy Laboratory. It addresses issues associated with reliable storage and retrieval of data required for building energy modeling (BEM) and will be an online repository for building component metadata and associated raw data that are generally required to build and execute building energy models.

The library will include window and wall constructions, heating, ventilation, and air-conditioning (HVAC) components, plug loads, weather data, code-compliant components, utility rate data, energy conservation measures, complete subsystems, and even whole buildings. BCL metadata include data conformant to a component taxonomy that enables searching and a flexible and extensible set of attributes. These attributes include length, width, weight, cost, U-factor, data provenance, and supporting video or images, along with other attributes that further define components. Model input can be stored in multiple formats, including OpenStudio (NREL 2011), EnergyPlus (Crawley et al. 2008), DOE-2 (York and Cappiello 1981) input, and various weather file formats.

## BACKGROUND

For years, organizations, researchers, and companies have sought a solution for storing all the data needed to design, model, construct, operate, and evaluate a building. This concept is termed *building information modeling* (BIM) and is the digital representation of the building in the form of building objects along with their attributes, properties, and relationships with other building objects (Eastment 2009).

The United States spends an estimated \$15.8 billion per year on issues related to interoperability (Gallagher et al. 2004). Several companies have implemented the BIM concept in different ways but with essentially the same goal (unifying building data into a single model). These entities have started developing online data repositories to store individual components and objects for their BIM offerings.

Autodesk released Autodesk Seek as a beta application in 2008 (Donn et al. 2009), which continues to be extended with product specifications and geometric representations for various computer-aided design (CAD) packages. The components on Seek contain several files, including the CAD file, and occasionally performance data (such as the IES file for lighting) (Autodesk 2011).

SmartBIM was created in 2010 by Reed Construction Data and Source2, which recently merged with ecoScorecard (SmartBIM 2011). SmartBIM's online

repositories contain building components for various needs. The SmartBIM objects typically include the product specifications and typically do not contain performance data suitable for BEM.

Google created the 3D Warehouse in 2005 (Google 2011), which allows Google SketchUp users to upload pieces or entire models into the warehouse to share. If an entire model is uploaded (and geolocated), the model may also be added to Google Earth. Google's 3D Warehouse does not contain detailed product data or performance data that are suitable for BEM.

Autodesk, buildingSMART, and Google create online repositories for building components. In general they focus on the architectural, geometric, and mechanical aspects, as the goal is to integrate the objects into CAD tools but not to directly target BEM.

There is no standard file format for defining the data needed for the BIM components. Two of the most prominent interoperability file format offerings are the Industry Foundation Classes (IFC) and Green Building eXtensible Markup Language (gbXML) (gbXML 2008).

The International Alliance for Interoperability branded the buildingSMART Alliance (IAI-bSa 2011) to better integrate the various formats of building data. The Industry Foundation Classes format is helpful for characterizing building architectural elements; however, it lacks a large amount of HVAC and other data needed for general BEM.

The gbXML format is based on the popular XML. In general, gbXML defines the geometric and architectural elements well, including a high-level HVAC definition that could be translated with various assumptions into BEM.

The BIM data format is further complicated when BEM is the goal, as the file formats depend on the particular information needed by different simulation engines. The BCL target is to provide data to the energy modeler. It also contains information related to the operational aspects of the building (schedules, water mains temperatures, etc.), and includes representative components that may not be available from a manufacturer. The components are typically used for codes and standards types of analyses.

## THE CASE FOR OBJECT-ORIENTED DATABASES IN ENERGY MODELING

The data used to describe and analyze buildings vary in type and dimensionality. Building characteristics data such as location, size, and the more complex envelope constructions must be stored alongside time series information such as performance and weather data at annual, monthly, daily, hourly, and even minute recording intervals. Moreover, aggregated data, either

of a single variable over time or of a sum or average of related variables, often need to be stored.

Relational database management systems (RDBMSs) have been the go-to model for database enterprise and Web solutions. These types of databases have multiple tables with columns representing different fields and rows representing individual records. Data associations are established through keys, which create relationships between records in different tables (Bowman et al. 1996).

Relational databases are widespread, well known, and well supported. Examples of open-source and proprietary RDBMSs include MySQL, PostgreSQL, Microsoft SQL Server, ORACLE, and SQLite. Their schema (the database structure representing each table, its fields, and its relationships to other tables and fields) must be defined and implemented before the database can be used. The SQL query language is used to insert, update, and return information from the database to the user or application (Bowman et al. 1996). When data are queried by an application, they are returned in a specific format, which must then be converted to a format (such as an array) that can be used by the application language, before they can be manipulated. The converse is also true: data stored in an array by the application must then be converted to SQL format before they can be inserted in the database.

Relational databases such as EnergyPlus's SQLite output (UIUC, LBNL 2010) have been used quite successfully to store building characteristic metadata and performance data. They are, however, rigid, bulky, and difficult to scale across several servers. Multiple tables are needed to represent complex relationships between variables, which makes querying the database complicated and error prone. Issues arise when new data need to be stored: additional fields must be incorporated into the database schema, which may result in new tables, fields, and relationships. These new fields must then be added to all previously stored records, which may result in many empty fields and much wasted space.

Time-varying building performance data present a storage challenge for any type of database, but relational databases have particular difficulty because of the large number of records, querying, and aggregation requirements these data entail. Relational databases usually scale vertically, which means processing power, memory, and space are added to a single "super" server to increase performance. This is more expensive (in maintenance and costs) than horizontal scaling, now that server resources have been commoditized. Horizontal scaling involves adding multiple modestly equipped servers to increase overall system performance. This scaling method has the added benefit of increased reliability through

redundancy. Relational databases can sometimes be run on multiple servers to increase performance, but these solutions are expensive and difficult to implement (Plugge et al. 2010).

The NREL Commercial Buildings group has had experience with two building-related Web applications built on relational databases: the 179D Easy Calculator and the High Performance Buildings Database (HPBD). The HPBD was originally developed in 2002 and substantially updated in 2011 (as discussed later). The HPBD contains many exemplary building case studies that are displayed on a dozen portals, including the American Institute of Architects and the U.S. Green Building Council websites, in addition to the main HPBD portal. Naturally, each portal's requirements differ slightly and change over time to increase its functionality. No matter how well-designed the original schema, tables and fields often need to be added to satisfy evolving needs. This is a costly and time-consuming task. Changes made for one portal must be propagated to the others, and mappings from the database fields to the code must be implemented. The result is an explosion of tables and join-statements, which makes the implementation challenging to maintain and the data difficult to reuse and repurpose.

179D estimates a building owner's potential tax deduction based on comparisons of lighting, envelope, and HVAC parameters to baseline simulation models. More than 200,000 simulations were run and the results were used to extrapolate equations that calculate estimated savings. The simulation results and aggregated results were stored in comma separated value (CSV) files, which were then loaded into a relational database that is specific to the 179D application and does not house the raw simulation results. Having a generic, centralized method of housing simulation data that may vary slightly from one record to another would facilitate its repurposing for other research studies and applications.

MongoDB is a document-oriented database that is optimized for speed and scalability (Plugge et al. 2010). It is a schemaless database, is not made of tables with rows of data, and does not use the SQL language for querying. Instead, data are stored in structured "documents" that can contain all the information related to an entity. All data are stored in one location, and no relationships are necessary between documents so MongoDB can easily scale to many servers. Scalability is essential in Web 2.0 applications, where users not only access information, but also generate content.

A user can store complex data in a single document, instead of using multiple tables to store and join statements to query the information in a relational database. Storing and retrieving data are easy and no

time is unnecessarily spent designing and updating schemas. Each document can be completely different from the others and structure does not need to be specified upfront. The mappings between the application code and the database are easier to implement and use: the data are stored in a binary format and are returned in typed arrays that can be used directly by the code, making the database faster to access, develop, and maintain.

MongoDB allows a special indexing technique called *geospatial indexing*, so geography-dependent information such as climate and other weather-related data can be queried by proximity to a given location. This is relevant to storing weather station locations and geolocated building data such as location.

The new HPBD implementation (DOE 2011) also uses the MongoDB database to store building data. As with our other data-intensive Web applications, the HPBD site is implemented in Drupal (a content management platform) (Drupal 2011) and uses a hybrid solution to store data. MySQL (MySQL 2011), a popular relational database, is used to store the Drupal content needed to render the site; MongoDB is used to store all building-related information.

In the HPBD, each MongoDB record, or document, corresponds to a single building's complete set of metadata and performance data. The documents are stored in a single collection, MongoDB's equivalent of a relational table. No schema needs to be defined ahead of time, and data can be easily structured in the documents. Embedded documents in the main document are used to structure multiple annual water and energy performance datasets as well as monthly utility bill data.

Most documents currently in the database are similarly organized, because they were created as part of the same application; however, there is no restriction on the number and type of fields each document can contain or on document organization. Null fields are simply not added to documents. This saves space, and a user can add new fields on the fly without updating stored records.

All data pertaining to a building are stored in a single MongoDB collection and all extraneous data related to the specific application are stored in a separate database. This facilitates data sharing between multiple applications and application program interface (API) development. Building data at various levels of detail—and captured through various means—could be stored in the same database as the HPBD data to create a centralized repository for all building-related information. This would facilitate data analysis and visualization.

## BUILDING COMPONENT LIBRARY IMPLEMENTATION

The BCL was designed to separate the building data from the website content, even though this required two databases. This separation enables greater future flexibility, the building of additional front ends on the building component data, and the performance of data mining routines without accessing user accounts or website content. A document database stores building data; a relational database stores website-specific data.

We chose the technologies used to implement the website based on several criteria: open source licensing, functionality, compatibility with server architecture, security, and community engagement related to the development. Multiple Web servers are used to facilitate expansion and handle the potentially large number of users. MongoDB was of particular interest because it scales horizontally and allows for splitting the database across several servers as well as replication. The replications create backups of data on other servers and can provide automatic failover.

We used agile principles to design the site because the technologies were uncertain, and revisited and enhanced many steps throughout the design. The steps used in the design were: (1) develop the site map, which included high-level design mockups and understanding how users would access the data; (2) design the component taxonomy structure (see next section); and (3) generate and upload components.

A beta release version of the BCL site is publicly available.

## TAXONOMY, METADATA, AND DATA

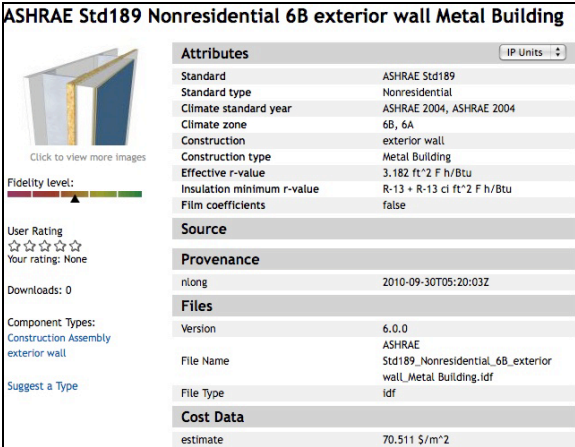
The vast number of building design parameters makes characterizing these in a single cohesive taxonomy a formidable, yet critical, task. During the taxonomy design process, we decided to keep the taxonomy as flat as possible to allow for future flexibility. Rigid taxonomies will not last more than a couple of iterations, so, based on user feedback, the taxonomy was allowed to morph.

For flexibility, we subdivided the taxonomy into a few principal categories: building design, construction assemblies, building loads (including miscellaneous electrical loads), HVAC systems, schedules, and location-dependent components. One category of interest is the miscellaneous electrical loads (Frank et al. 2011), which includes hundreds of products such as computers and appliances that are uploaded into the BCL.

The taxonomy describes the component types in the BCL; each has an unnumbered set of attributes. Currently, the attributes are broken up into a few data types consisting of strings, integers, and doubles. Each

attribute type is expressed differently in the search facets functionality described in the next section.

Figure 1 shows metadata that are available for a typical BCL component. These can vary by component type. Many components presently contain images and component-specific attributes, information about the source or manufacturer, a description of the data provenance, cost, and a summary of attached files available for download. Attached files can be retrieved from the component page via a download link, which provides a zip file containing images, videos, and model input file snippets as appropriate. The file snippets are not restricted to any particular modeling software.



ASHRAE Std189 Nonresidential 6B exterior wall Metal Building	
<b>Attributes</b>	
Standard	ASHRAE Std189
Standard type	Nonresidential
Climate standard year	ASHRAE 2004, ASHRAE 2004
Climate zone	6B, 6A
Construction	exterior wall
Construction type	Metal Building
Effective r-value	3.182 ft <sup>2</sup> F h/Btu
Insulation minimum r-value	R-13 + R-13 ci ft <sup>2</sup> F h/Btu
Film coefficients	false
<b>Source</b>	
<b>Provenance</b>	
nlong	2010-09-30T05:20:03Z
<b>Files</b>	
Version	6.0.0
File Name	ASHRAE Std189_Nonresidential_6B_exterior wall_Metal Building.idf
File Type	idf
<b>Cost Data</b>	
estimate	70.511 \$/m <sup>2</sup>

Credit: Building Component Library / NREL (<http://bcl.nrel.gov>)

Figure 1 A BCL Component

Attribute metadata are currently used to filter search results, and are described in the next section. More metadata and associated dynamic calculations will be added to enable high-level analysis external to BEM engines. For example, life cycle cost data and site/source transport calculations may be easily added to BCL. Users can employ a comma separated value export facility coupled with BCL's search and sort functionality to quickly perform life cycle analysis of a prospective component.

Each BCL component is versioned and assigned a unique identifier that can be accessed via an associated uniform resource locator (URL). This will facilitate citation and enable BEM to be both transparent and repeatable. This will increase BEM credibility. Specifications, publications, and entire models can contain references to specific versions of components by URL.

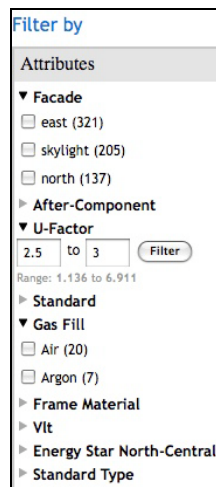
## FACETED SEARCH FUNCTIONALITY

The BCL can perform multifaceted searching, enabling users to quickly identify relevant components that meet specific constraints. This will become a key feature for



efficient access to relevant data, as the BCL will certainly grow to contain millions of components. Apache Solr was identified as a capable, open source search engine that could integrate with the Drupal content management and MongoDB framework used for this project. It provides rapid full text searching, faceted search, database integration, and other enabling features that could quickly be leveraged for BCL (Apache 2011).

In practice, user search queries return a list of potential components along with a filter sidebar that enables users to efficiently refine searches by constraining attributes, component types, and file formats. For example, the search term *window* returns a list of all window-related components that may be sorted by relevance, user rating, or upload date. A contextually appropriate list of relevant attributes that may be used to quickly refine search results appears in the sidebar. A partial list of appropriate attributes automatically returned in the sidebar of a window search is shown in Figure 2.



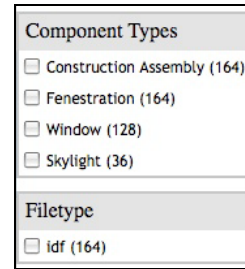
Credit: Building Component Library / NREL (<http://bcl.nrel.gov>)

Figure 2 BCL Filter by Attributes

In this example, the initial search returned 1,412 components associated with the search term *window*. A specified U-factor range reduced the results to 164. Multifaceted searching also further reduces options, and the list of available attributes changes dynamically based on the components remaining in the filtered search.

Users may employ the filter by component type or file type to further refine their searches (see Figure 3). These are also dynamically constructed and appear in the filter sidebar of a search. In this case, the user might specify a skylight component, which would reduce the search to only 36 components. Each search result has an associated EnergyPlus .idf file snippet. Search

results producing alternate data formats such as .epw files would also appear as search options.



Credit: Building Component Library / NREL (<http://bcl.nrel.gov>)

Figure 3 Component and File Type Filters

## INTEROPERABILITY

An API for accessing the search engine and returning specific BCL components is provided for external applications, and is critical for transparent data access and integration. The API documented on the BCL website includes application examples for returning programmatic queries in various formats, including YAML, XML, and JSON. The API is freely accessible to registered users.

Three methods are exposed in the API:

- Search. The user can concatenate filters for each component attribute.
- View. The user can see the details of each component of interest.
- Download. The user can send a list of identifiers to download the components.

An example of a search query resulting in two records that represent Denver follows. The API key is needed to verify the user's identity.

```
http://bcl.domain.name/api/search/denver.xml?show_rows=02&oauth_consumer_key=#{apikey}
```

As components are uploaded to the BCL, additional attributes will be created and periodically added to the faceted search index and made available through the API.

The OpenStudio suite and software development kit (SDK) are currently being extended to use BCL data. The SDK includes an object-oriented abstraction to BEM (presently emphasizing EnergyPlus) that is being closely aligned with BCL content. OpenStudio reverse translators (translating from EnergyPlus to the OpenStudio native format) can use raw .idf snippets presently stored in the BCL; however, we envision extending BCL data with OpenStudio component (.osc) models as the format matures. The .osc format contains information required for the EnergyPlus simulation engine. Raw snippets in other software formats, such as

Radiance, can also be stored along with universal metadata native to the BCL (Ward 1994).

In practice, OpenStudio will provide a mechanism for managing a local database of user-specified components. Users would draw from the local library to create a project-specific library containing constructions, schedules, weather input, and other relevant data for that specific energy modeling project. Data synchronization and revision management with the energy model will be transparent to users who interact with the BCL through the proposed integrated ProjectManager user interface built with OpenStudio (see Figure 4).



Credit: Marjorie Schott / NREL

Figure 4 Mockup of an OpenStudio Local Component Library

## SOCIALIZING DATA

The potential scope of a fully populated BCL is substantial, and the cost to populate and manage millions of components is almost certainly more than a single organization could bear. To address these challenges, the BCL was designed with social functionality.

Social engagement is offered through a component rating system whereby a user may assign a qualitative measure of 1 to 5 “stars” to assess the overall quality of a component’s metadata and data. The score reflects a combination of community-perceived accuracy and utility, and is used as a sorting criterion by the search engine. A second method of social interaction is the facility for a user to comment on components. This provides a means for more detailed feedback than does the simple voting mechanism, and can enable a user to identify specific deficiencies in metadata or data.

Components are ultimately envisioned to be publicly submitted and revised through a user-generated content module developed for the BCL. This module will also provide Atom or Really Simple Syndication (RSS) news feeds for information about component updates

and recently added comments that are useful for component owners and users. Publicly submitted components will be distinguished from those submitted by standards organizations or other “trusted sources,” and faceted searching will enable end users to identify appropriately vetted components based on their needs.

## CONCLUSION

BEM penetration has been limited across the sector because analysis tools are complex and gathering appropriate and trusted input data is difficult. BCL is intended to address the latter issue by simplifying the task of gathering model inputs. Providing mechanisms to cite and repeat analyses will also provide substantial benefit to the sector through easier replication of design and transparency for those who are required to inspect the underlying assumptions of a given analysis. BCL data are socialized to distribute the effort of populating and maintaining such a large database, and mechanisms have been created to help users identify components of high quality and utility.

## ACKNOWLEDGEMENTS

The authors would like to acknowledge the support of the U.S. Department of Energy’s Building Technologies Program including former Buildings Program Manager, Drury Crawley, and current Buildings Analysis Tool Program Manager, Amir Roth. The authors would also like to thank Oliver Davis, Matt Brown, Allan Wintersieck, and Mike Chin from concept3D for their development efforts.

This work was supported by the U.S. Department of Energy under Contract No. DE-AC36-08-GO28308 with the National Renewable Energy Laboratory.

## REFERENCES

- Autodesk. 2011. Autodesk Seek. <http://seek.autodesk.com> (accessed August 11, 2011).
- Bowman, J.S., Emerson, S.L., Darnovsky, M. 1996. *The Practical SQL Handbook: Using Structured Query Language*. Addison Wesley, Third Edition.
- Crawley, D.B., J.W. Hand, M. Kummert, and B.T. Griffith. 2008. “Contrasting the Capabilities of Building Energy Performance Simulation Programs.” *Building and Environment* 43 (4): 661–673.
- DOE. 2011. High Performance Building Database (HPBD). Washington, D.C.: U.S. Department of Energy. <http://buildingdata.energy.gov>.
- Donn, M. Crawley, D. Hand, J. Marsh, A. 2009. “The Provenances of Your Simulation Data.” *Glasgow, Scotland: Building Simulation 2009*. 1405–1412.
- Drupal. 2011. <http://drupal.org> (accessed August 11, 2011).



- Eastment, C. 2009. "What is BIM?" Article Last updated August 2009. <http://bim.arch.gatech.edu/?id=402> (accessed August 13, 2011).
- Frank, S., L.G. Polese, E. Rader, M. Sheppy, and J. Smith. 2011. "Extracting Operating Modes from Building Electrical Load Data." Baton Rouge, A: Proceedings of the 2011 IEEE Green Technologies Conference 2011, pp. 1–6.
- Gallaher, M.P., A.C. O'Conner, J.L. Dettbarn, and L.T. Gilday. 2004. "Cost Analysis of Inadequate Interoperability in the U.S. Capital Facilities Industry." Gaithersburg, MD: National Institute of Standards and Technology.
- gbXML. 2008. Green Building XML Version 0.37. <http://www.gbxml.org>. (accessed August 13, 2011).
- Google. 2011. Google 3D Warehouse. <http://sketchup.google.com/3dwarehouse/> (accessed August 11, 2011).
- Apache. 2011. Apache Solr. Last published July 2011. <http://lucene.apache.org/solr/> (accessed March 31, 2011).
- IAI-bSa. 2011. BuildingSMART Alliance. <http://www.buildingsmartalliance.org/> (accessed August 11, 2011).
- MySQL. 2011. <http://www.mysql.com/> (accessed August 11, 2011).
- NREL. 2011. <http://openstudio.nrel.gov> (accessed May 20, 2011).
- Plugge, E., Membrey, P., Hawkins, T. 2010. The Definitive Guide to MongoDB. Apress, First Edition.
- SmartBIM. 2011. <http://smartbim.com> (accessed August 11, 2011).
- UIUC, LBNL. 2010. EnergyPlus Documentation: Output Details and Examples. U.S. Department of Energy.
- Ward, G.J. 1994. The RADIANCE Lighting Simulation and Rendering System. Proc. SIGGRAPH.
- York, D. and C. Cappiello. eds. 1981. DOE-2 Reference Manual (Version 2.1A). Berkeley, CA: Lawrence Berkeley National Laboratory.