**“THE GREEN BUILDING XML EXTENSIONS FOR NREL’s OPENSTUDIO”**

**SUBCONTRACT NO. LHQ-6-52192-01**

**Final Report**

**9/30/2016**

**Executive Summary**

**Goal and Objectives**The goal of this project was to validate the National Renewable Energy Lab’s (NREL’s) OpenStudio software tool to produce valid gbXML and also set the stage to validate other BIM and building analysis software tools in the future. There were three main objectives of this validation work.

1. Demonstrate that the OpenStudio software could pass the gbXML validation procedure
2. Encourage other software vendors to certify their software using the validation procedure as well
3. Work towards a generic validator that could be used on general user models

**Accomplishments**

We were able to accomplish the following tasks in support of the above objectives:

1. Developed OpenStudio models for the validation procedure
2. Validated OpenStudio gbXML export/import so that it became the 1st tool to be officially certified
3. Launched a new gbXML.org website and improved the validation website and documentation
4. Promoted gbXML validation efforts via seminars, webinars, social media, and e-newsletters
5. Developed the requirements for a future generic validator

**Future Work**As a result of the work accomplished above, we determined that there is still much work to do, including:

1. Further Develop Level 3 Compliance: We will be working with Autodesk to develop use cases for gbXML automation within Autodesk Revit and Autodesk 360.

2. Develop a gbXML Conversion Tool: Many software tools are still importing/exporting 0.37 versions of gbXML. While it may take years for these software vendors to update their software to comply with gbXML version 6.01, a conversion tool will easily update these previous versions to the latest version.

3. Develop a Generic Validator and Open-Source Geometry Engine: We still need to develop a “generic” validation software tool that can be used by additional stakeholders including energy modelers, engineers, architects, and others. This tool should be able to validate any generic gbXML file, not just test-case gbXML files.

4. Create a gbXML Portal and Accompanying Web Service: A long-time goal has been to create a gbXML “portal” that allows users to register and upload gbXML files for remote storage, version control, sharing, and much more (see **Future Work** below).

**Overall Goal and Objectives**

The goal of this project was to validate the National Renewable Energy Lab’s (NREL’s) OpenStudio software tool to produce valid gbXML and also set the stage to validate other BIM and building analysis software tools in the future. There were three main objectives of this validation work.

1. The first objective was to demonstrate that the OpenStudio software could pass the gbXML validation procedure.
2. The second objective was to encourage other software vendors to certify their software using the validation procedure as well.
3. The third objective was to work towards a generic validator that could be used on general user models, as opposed to the strict test case models required by the gbXML validation procedure. Delivering a fully working validator was out of scope of this work, therefore requirements were developed to set the stage for future work.

In support of the first objective, we developed OpenStudio models which represented the buildings in the validation procedure test case. These models were generated programmatically using the OpenStudio Ruby Application Programming Interface (API) to reduce maintenance costs and to allow them to be leveraged for other purposes. OpenStudio passed the validation procedures and is now the first software authoring tool to officially become “gbXML certified” (<http://gbxml.org/OpenStudioCertification_Latest> ).

In support of the second objective, we drastically improved the validation website, documentation, and tools required to apply the validation procedure. We publicized the validation procedure and encouraged gbXML software vendors to apply it to their tools. This required contacting other gbXML software vendors directly as well as making public announcements, conducting live webinars, and promoting other ways to generate user interest in the validation efforts. We publicized the state of the OpenStudio validation efforts to encourage other software tools to apply the validation procedure to their own tools. In addition, we simplified and streamlined the validation process to allow the process to be both more responsive and clear to market demands, while also allowing room for future growth of test cases. In fact, Autodesk will soon be validated to Level 2 compliance (see below) and is working with gbXML.org on Level 3 compliance.

In support of the third objective, we developed requirements for a generic validator. User stories were developed and preliminary mockups were made. The most important user story developed was to allow a user to upload their gbXML file to the gbXML.org website and view a 3D representation in their web browser. This would allow the user to identify any problems with their model visually. The second use case was for software to detect certain classes of problems with the user’s model and identify those visually in the 3D display. Finally, the use case of upgrading user models to the current gbXML schema version was identified.

**Tasks**

Here are the tasks that were performed to achieve the above objectives:

**1. Developed OpenStudio Models for Validation Procedure**

We developed OpenStudio models for a number of the buildings in the validation procedure test suite. The test cases were derived from the ASHRAE RP-1468 documentation. The models were programmatically generated using the OpenStudio Ruby API. All scripts and model content developed for this purpose were developed under the Lesser General Public License (LGPL) open source license and are maintained in a public repository on github.com (<https://github.com/GreenBuildingXML/openstudio-gbxml-validation> ).

**2. Validated OpenStudio gbXML Export/Import**

We applied validation procedures to each of the OpenStudio models developed in task 1. We verified that each of the validation test cases imported correctly into the OpenStudio SketchUp plug-in. We also addressed any issues with the validation software that were found in this work. For example, the OpenStudio validation forced gbXML.org to address geometry created by thin-walled geometries, and metric-only software. Previous versions of the validator only addressed thick-walled geometry engines, and supported IP-units. The validation process and OpenStudio were made more robust as a result of this process.

Highlights of improvements/fixes to OpenStudio as a result of this project:

* correctly specifies SlabOnGrade elements
* correctly handles area calculations of sloping floors and ceilings (before calculated area as zero)
* handles most second-level space boundary translations, automatically, for simple geometries
* gbXML export up-to-date with version 6.01 of gbXML

Highlights of improvements/fixes to gbXML validation as a result of this process:

* unit of measure handling
* special procedure development for thin-walled geometry creators, opening up validation to a wider audience
* improved validation website user interface and user experience
* better error-handling and user messaging when validation fails
* improved geometry validation engine

**3. Improved Validation Website and Documentation**

The previous validation website was not hosted on the gbxml.org domain which reduced its visibility and credibility. For this task, we moved the gbxml.org website to a hosting platform that supports the validator software. The gbxml.org website was redesigned to more prominently display the validation procedure and documentation. The website now shows which software tools have been validated or are in the process of being validated. See <http://www.gbxml.org> for more details.

**4. Promoted gbXML Validation Efforts**

This task was performed in parallel with other tasks listed above; the purpose was to keep other software vendors and the public up to date on gbXML validation efforts. We attended the four day SimBuild 2016 Conference in Salt Lake City in August 2016 to promote gbXML validation efforts. In addition, we conducted two live webinars to explain our work.

**5. Generic Validator**

Delivering a fully functional generic validator was out of scope for this work. Therefore, work on this area was focused on developing user stories and requirements for future work. The most important user story developed was to allow a user to upload their gbXML file to the gbXML.org website and view a 3D representation in their web browser. This would allow the user to identify any problems with their model visually. The second use case was for software to detect certain classes of problems with the user’s model and identify those visually in the 3D display. Finally, the use case of upgrading user models to the current gbXML schema version was identified.

Integration with the Autodesk Forge API was investigated as a potential solution for viewing user submitted 3D models. However, this is still a work-in-progress since the Forge API is in beta phase.

**Significant Findings and Issues**

During this project, we decided that 3 levels of gbXML certification, or compliance, were required to provide more clarity to the community of users and vendors.:

1. Level 1 compliance involves validating that a gbXML file is a well formed XML (per the W3C ISO Standard) and also conforms to the [gbXML XSD](http://gbxml.org/Schema_Current_GreenBuildingXML_gbXML) (from gbXML versions 0.37 to 6.01, depending upon which version the software tool currently exports).
2. Level 2 compliance involves validating a gbXML file against 8 to 10 geometric “test-cases” that are based upon ASHRAE Research Project 1468, “Development of a Reference Building Information Model (BIM) for Thermal Model Compliance Testing”. Level 2 requires that second level space boundaries be correct for the simplest test cases, and pass a small subset of translation edge cases.
3. Level 3 compliance has yet to be fully defined. However, Level 3 does involve certain levels of vendor tool automation that goes far above and beyond Levels 1 and 2 compliance. We are currently working with Autodesk to better define Level 3 compliance (See the **Future Work** section for more details).

**List of Issues**OpenStudio passed the following test cases:

1. Test Case 3: Test for proper second level space boundary representation
2. Test Case 6: Test for proper second level space boundary representation
3. Test Case 7: Test for basic pitched roof representation
4. Test Case 8: Test for basic sloped slab on grade representation
5. Test Case 12: Test for proper second level space boundary representation
6. Basic Whole Building Test Case 1: Test for proper second level space boundary representation

To pass these test cases, it did require that NREL make some updates to the OpenStudio SDK responsible for gbXML export since there were a few errors that were pervasive for every test case. Therefore, NREL needed to make some changes to the gbXML export feature of OpenStudio to be compliant with the validation process. gbXML provided NREL with guidance as to how each XML file should look in order to pass, which NREL took and made changes to their code base. In some cases, gbXML decided to relax configurable constraints, to allow the document to pass.

Below is the full list of issues and changes to the OpenStudio code base made as a result of this work effort:

1. There were no BuildingStorey definitions in any export from OpenStudio. This is a required element, but we relaxed this for validation purposes.
2. The Building->Area calculation that was done during export did not meet gbXML specifications for the Building Area calculation. To be fair, gbXML may not have the tightest definition in terms of the criteria for when a Space Area is included in the Building Area, but the point is, plenums and other non-occupiable spaces shouldn’t be included in the building area calculation.
3. Any time there was a floor that was on-grade (industry standard is when z=0, and it seems OpenStudio follows this convention) we would expect the surfaceTypeEnum for that surface to be SlabOnGrade, but OpenStudio defines these surfaces as UndergroundFloor. This needed to be changed.
4. Thin-Shelled geometry challenges: The fact that OpenStudio makes gbXML derived from a thin-walled geometry paradigm consistent with the building energy modeling paradigm as opposed to a BIM model posed a basic philosophical challenge for the validation process. Originally, the validator was designed for BIM-centric tools that have wall thicknesses inherent in the modeling environment. Autodesk Revit, for example, assumes that the wall thicknesses affect the volume and area calculations, even though the wall vertices go to the centerline of the thickness (as if the walls had no thickness at all). When we tried to use these same standard test files as a comparison point for a gbXML created via OpenStudio, we encountered an issue. Whereas we modeled the walls in OpenStudio as if they were on the centerline, with the same coordinates as the standard files, now the volume and area calculations came out differently than the standard files. Of course, if we tried changing things around and modeled the surfaces in OpenStudio at the inner wetted perimeter, we get the volume and area correct, but now the polygon coordinates are in a different location and don’t match the standard file PolyLoop coordinates. So either way, it is not a perfect process. This fact leaves us with a few different options that we could potentially pursue:

* We draw OpenStudio geometry at the centerline and apply a tolerance (set in the validator) on the volume and area comparison that would be the same for any vendor. We make this tolerance sufficient to allow tools like OpenStudio to pass. The upside of this approach is it is simple. The downside is, it adds “slop” to the validation process
* We ask OpenStudio to leverage gbXML’s DocumentHistory element which includes a section for ProgramInfo. The validator would be programmed to look for this, and see it is an OpenStudio document, and a flag would be set in the validator to use different volume and area values than what is in the Standard Test file. The advantage here is we can be more specific to the vendor, and eliminate the slop problem. The downside is, it is slightly more code to maintain and could require some code redesign if there were a day when a lot of vendors had individual use cases that warranted one-off situations.

1. Changes to the validator code base: We made several improvements to the validation engine itself, and also to the standard test case XML files as a result of this work that has already been incorporated into the latest version of the validator. We found two problems with the standard test case files: Occasionally, there were still errors in the files that should no longer persist by the end of this process. The whole building test cases, made at the end of phase two, were particularly problematic because we made them with Honeybee out of Grasshopper. They were not fully-vetted and are quite large and contained several errors we did not pick up when they were first created. Secondly, sometimes the decimal precision was pretty extreme because for all the other test cases we used Revit in US-IP units to create the test cases originally. This meant for sloping surfaces, there was sometimes units like 9’10-11/16” that made for some difficulties when re-creating the test cases. We tightened these up so for a majority of the test cases there was much less guess-work and complexities around the units of measure.

For the validator itself, we found that some of the features put in the first time, most notably the tolerances and the “failure description file” were good additions and served the validation process well. The tolerances allowed us to define the precision required to pass or fail the test case.

There were precision tolerances for:

* length (in feet)
* area (in square feet)
* volume (in cubic feet)

The units were US-IP because the standard files all use these units and the units of measure shown. We’ve considered expanding the tolerances so they can be alternatively defined in metric, or as a percentage of the measurement in question (e.g. - +/- 1% area) but thus far no one has demanded it or asked it as a feature. A second short-coming is that the tolerances are universal for each test case and defined at runtime.

The other feature that still remains is the “failure description file”. This object allows us to programmatically define which unit tests allow are required to “pass” the validator’s test case. This file can be used on a test case-by-test case basis. For example, right now because of the volume and area in OpenStudio not meeting the stringency requirements, we can programmatically say that this test is not crucial, and the validator gives an “overall pass” despite failing these tests. This gives us another little safety valve that is an alternative where we don’t want to change the tolerances but still want to see a “Pass”.

Getting a pass is not an easy task. We found that because the validator relies on a series of individual unit tests (test the building area, test the space areas, test the space ids, etc.) run serially one after the other, there was a flaw in the logic in the validator that would prevent it from continuing if one of the unit tests failed individually. This happened at the surface test level. Therefore, we have removed this restriction, and now the unit tests will all be allowed to run to completion and report on all errors, regardless if one of the unit tests failed.

Some of our unit tests were also just too constricting:

* Tilt and azimuth were expected to match in the standard and uploaded test Files. However, for horizontal surfaces, the azimuths never match. We’ve relaxed the validator to allow any azimuth for horizontal surfaces. Before, the validator would fail when azimuths were not within a certain specified tolerance.
* Height and width are somewhat arbitrary, and completely meaningless for polygons that were not quadrilateral. So now the only requirement is that the product of the height and width (i.e. – the surface area) should be within a specified tolerance. We are looking to improve this further.

A second issue that has been mentioned is the ambiguity between interior floors and ceilings. The validator unequivocally would look for a ceiling definition if the standard file had a ceiling, and look for a floor if the standard file had a floor, and did not treat them as equivalent surfaces. Now, the validator looks for both interior ceilings and floors, and can successfully match. There are special cases included to test the tilts appropriately. Thus far, we have only had to cover instances where the tilt specifies a horizontal surface (i.e. – 0 or 180 degrees). We may include an edge case where the tilt is not 180 or 0 for an interior ceiling or floor, and see what might happen in that case.

**Future Work – Continued Maintenance of gbXML**

Despite all of that was accomplished during this project, there is still much more to do. The following is a list of gbXML maintenance items that we will continue to work on:

**Improving the Documentation**While we’ve drastically expanded upon the gbXML test case documentation online (<http://gbxml.org/TestCases_for_GreenBuildingXML_gbXML> ), there is still work to do improving the documentation, including:

1. Many vendors do not understand what governs the tests. For example, in each test case there are certain individual tests that can fail, yet overall the vendor can still comply despite these errors. Secondly, there are tolerances on each test which can be changed on the fly.
2. Currently the documentation consists of a web page and a written pdf document for each of the tests the vendors must submit. The new documentation must do more than this including educating the reader on the nature of the test:
   * Specifically, what is the test actually “testing”?
     + Is it core functionality that every vendor must achieve to pass?
       - If so, what core functionality is being tested?
     + Is the test a “reach” test that extends the score and overall rating of the vendor, should the vendor pass this test?
       - If so, what extended functionality is being tested?
3. Improve overall clarity:
   * Currently the documentation attempts to clarify common pitfalls and uses some screenshots to attempt to clarify the intent, along with instructions shown with text and sketches to try to convey intent. A review of the documentation has shown that the clarity and intent could be improved.
     + Can the expected outcomes of the test be shown in better detail? Color coded axonometric drawings could reveal what walls should become interior, vs. exterior, for example, or how the walls should be broken up.
4. A reach goal for gbXML.org: 3-D Web visualizations of the test cases that will allow the user to more easily interact with the test cases. This intuitively would allow the user to more easily understand requirements.

**Educating the Audience**As part of this project, we developed a new gbXML.org website that better conveys the gbXML message. In addition, we added a “community” portion to the website that includes forums and a gbXML knowledgebase. Despite these efforts, there’s still a ways to go to help stakeholder’s better understand all facets of gbXML.

**Relaxing the Validation Requirements – Reducing the number of test cases required for certification**Currently, there are 14 test cases that gbXML has presented to industry as requirements for certification. There are actually 30 test cases that are required in theory, but the remaining 16 were deemed too onerous or esoteric for use. Nearly all of these test cases were based upon research done at Stanford’s CIFE and Texas A&M under the leadership of Jeff Haberl, with a few being based upon feedback from the user community who leverages gbXML on a regular basis in their workflows. Much of the work accomplished by these parties covered not only second level space boundary tests, but also special cases where specific, and sometimes non-trivial BIM-to-BEM workflows had to be considered.

Industry has noted that, despite reducing the number of test cases, even those remaining, are beyond their capabilities and budgets to keep up with gbXML requirements. Some of the test cases would require changes to the vendors’ code base that does not fit neatly in their lengthy, market oriented backlogs that are years in the making. gbXML validation, in the reality of this industry environment, must be pared down to

* “essential” test cases for compliance, and
* “reach” goals for compliance.

Many of the vendors should be able, without a huge amount of effort, achieve this simplified “essential” compliance, and then move on to more complex cases, which can be driven internally by their organizations or prioritized through feedback from their users. gbXML believes this will both help industry achieve compliance faster, as well as help to clarify for vendors where they still need to improve, giving the vendors flexibility on how to improve their gbXML production based on market demands.

**Relaxing the Validation Requirements – Relaxing constraints**This is perhaps the most challenging part of the updates to the certification process. As the constraints are relaxed and to make it easier for users to comply, there is a likely commensurate increase in effort to re-code parts of the validation engine. This will be a work in progress and evolve over time.

To describe what will be relaxed, we should first briefly describe the validation process and each of its various pieces. As the pieces are described, we will comment on the current state of the art, and whether or not any relaxations will be put into place.

1. Testing for valid XML: here, we ensure that the xml is “valid xml” meaning there are no malformed tags, incorrect use of characters, and that proper encoding is used, etc. We currently require that all XML follow the W3C Recommendation 26 November 2008 (Extensible Markup Language (XML) 1.0 (Fifth Edition)). We are now calling this Level 1 compliance.
2. Testing for valid gbXML: here, we ensure that the XML that has been submitted adheres to the gbXML XSD Schema. We are now calling this Level 2 compliance.
   1. Change 1 - Currently the validator only accepts one version for validation. The suggestion is to add a drop down menu will to allow the user to text against different XSDs.
3. Individual tests vs. overall tests: the certification process requires that the vendor successfully PASS several test cases (these test cases are shown in table 1 above). Each test case is likewise made up of individual tests, ensuring floor areas match in the vendor-created file and standard gbXML test case file. Currently there is an “overall pass” that indicates whether the vendor has passed the test overall. A table is held in the validator’s database that determines which individual tests are required to pass.
   1. We currently do not intend to change the individual tests that are required for the overall pass
   2. However, it will be more transparent which individual test is required to pass each case, and documentation will make it clear what the individual tests require to pass.
4. Metric vs IP
   1. Currently all of the test cases are written in IP and it is expected that all vendor files are in IP, in the same IP units no less.
      1. In the future, the vendors may submit their test cases in whichever units they most prefer. So long as the units of measure comply with gbXML XSD requirements, any unit is acceptable.

The general idea is, a standard file resides on the gbXML server. It has been scrubbed to ensure a properly defined gbXML file that is in adherence with the schema. The vendor submits a gbXML file that they have drawn on their own. Each of these individual tests is conducted to ensure the vendor has drawn the test case properly and that the vendor tool has translated the geometry successfully to gbXML.

**Future Work – New gbXML Features**

**Further Develop Level 3 Compliance**We will be working with Autodesk to achieve Level 3 certification, which has yet to be fully defined. It does involve certain levels of vendor tool automation that goes far above and beyond Levels 1 and 2 compliance. Autodesk desires to achieve Level 3 compliance for both Autodesk Revit and Insight 360. Some examples of Autodesk-specific Level 3 automation include:

1. Handling ‘real’ architectural models i.e. supporting a very wide variety of modeling practices, coping with natural inaccuracies (small gaps / overlaps) and scaling from concept to detailed design
2. Automatic perimeter / core thermal zoning
3. Automatic identification of elements acting as shading (as opposed to room bounding) without manual definition
4. Handling ‘Sandwich’ conditions i.e. when two or more elements are adjacent or very closely adjacent and often not perfectly parallel but essentially a single gbXML surface
5. Definition of material thermal properties (which get very interesting when combined with sandwich conditions)
6. Working with or with explicit room/space objects and their associate metadata
7. Above / below grade
8. Columns
9. Openings
10. Ceiling voids

Discuss the possibility of “use case” based validation. For example, what fields are required for the energy modeling use case (or maybe the OpenStudio use case)? What fields are required for the HVAC loads use case? See <http://www.hpxmlonline.com/tools-resources/data-selection-tool/> for an example of this.

**Develop a gbXML Conversion Tool**

Both NREL and Autodesk have requested that we develop a simple software tool (web-based) that converts previous versions of gbXML (i.e. – 0.37) to later or the current versions since major tools such as Autodesk Revit continue to support the 0.37 version while validation only supports the latest version. This is not an easy task on the part of the software vendor to update to the latest version since tools like Autodesk Revit have long-term “wish-list to release cycles” (often 1 or more years). Therefore, developing a conversion tool will allow these previous versions of gbXML to update to later versions.

**Develop a Generic Validator and Open-Source Geometry Engine**

While we have successfully developed a test-case validation tool (<http://gbxml.org/validator/Pages/TestPage.aspx> ), we still need to develop a “generic” validation software tool that can be used by more stakeholders including energy modelers, engineers, architects, and others. This tool should be able to validate any generic gbXML, not just test-case gbXML files. We believe we are closer than ever to achieving this vision, however, there is limited time and budget to test such concepts. Developing unit tests, and making incremental improvements to this code base, is a full-time effort that is constantly in need of development efforts.

The software development effort taken thus far to develop vendor certification tools will be leveraged for the generic validator, one that can accept any user model. A web page shall be developed on gbXML.org so a user can upload their gbXML model and the validator shall determine if there are any defects. If there are defects, the website shall provide information to alert the user to any defects that were found. A web based 3D visualization tool may be provided to visualize the uploaded model and to identify the defects in a meaningful way. We plan on using Autodesk’s Forge Viewer API to translate the gbXML geometry into a web-based model.

**Create a gbXML Portal and Accompanying Web Service**Along with the generic validator wish-list item above, another long-time goal has been to create a gbXML “portal” that allows users to register and upload gbXML files for remote storage. We would create a web service (or web API) that could be accessed by authorized software tools to import and/or export gbXML files to and from this portal. This would provide the following benefits:

1. Different “dot” versions of gbXML could automatically be converted to the appropriate “dot” gbXML version that is supported by a consuming software tool.
2. Different versions of the same gbXML file that is produced by a BIM authoring tool could be stored in the portal so that a consuming tool could easily re-import it and update any changed information. Think of it as a sort of “version control” function.
3. If enough gbXML files are uploaded to the portal, we could begin to analyze the data and look for trends that may benefit the industry. For example, icould be analyzedso as