ASHRAE and IBPSA-USA SimBuild 2016
Building Performance Modeling Conference
Salt Lake City, UT
August 8-12, 2016

AUTOMATED METHODS FOR IMPROVING ENERGY MODEL QUALITY ASSURANCE AND QUALITY CONTROL

Chris Balbach, PE, CMVP[1] and David Bosworth[2],
[1]Performance Systems Development of NY, Ithaca, NY
[2]BUILDlab, Dryden, NY

ABSTRACT

While industry standard definitions of acceptable limits of energy model accuracy remain elusive, there are renewed interests in improving the overall accuracy and reliability of the products of energy modeling tools - predictions. While recent advancements in open-source energy modeling frameworks related to Quality Assurance and Quality Control provide users with powerful and flexible means for automating these activities, key gaps remain. This paper will discuss advances in both Quality Assurance and Quality Control, provide examples of their use in current practice, and propose a logic and framework that could support the broad use of these techniques in everyday building energy modeling practice.

INTRODUCTION

We begin by defining distinctions between a product and the process used to generate the product. In the context of energy modeling, we will consider our product of interest to be a prediction derived from a model. The scope of a prediction can vary, from focusing on a single component, a collection of components or a collection of systems. While many simulation use-cases focus on energy usage as the prediction variable, other metrics such as energy cost or environmental impacts may be useful in supporting decision making. To simplify our discussion, this paper will consider products to be produced by simulation at different hierarchical levels of an energy model. In the context of energy modeling, we then define processes as the series of actions or steps taken in order to generate the products.

This is an important distinction, as we propose model Quality Assurance (QA) activities to relate to improvements in process, while model Quality Control (QC) activities relate to improvements in product. Thus, QA activities can and should occur during energy model development - but do not require executing an energy simulation (i.e. generating a product). Model QA activities can also vary in scope - for example, a QA activity might be to generate a report describing the quantity of defaulted input parameters that are present in the current state of a model. Another example of QA would be range checking a model component input value. A key continuous improvement principle involves ongoing searches for process improvements, such as improved procedural actions taken to document the composition of an evolving energy model. Aside from a technical description, model QA activities also define roles and responsibilities for procedural execution of the activity (frequency, etc) and how the results are to be interpreted. Well-defined model QA activities also generate sufficient documentation to support an audit process - acting as a management layer to document the proper application of QA activities. A well defined system of Energy Model QA along with documentation of QA results and actions can further increase trust in overall product quality.

An example of an object level model QA test might be the inspection of the target lighting setpoint of a daylighting control object. Regardless of the control strategy (open loop, closed loop, stepped, continuous, on-off, etc). Since there are space types such as hospital examination rooms or educational computer labs that require high or low lighting levels, a QA warning identifying these possible scenarios should be triggered if the target setpoint is less than 275 lux or greater than 550 lux. A QA error should be triggered if the target illuminance is less that 30 lux or greater than 1000 lux. We distinguish a QA warning with a possible but unlikely value, while an error represents a value that is physically impossible or is non-compliant with levels defined in codes or standards.

An example of a system level QA test might be the inspection of the minimum loop temperature of a chilled water loop. Unless an ice storage system is attached to the loop, a QA warning should be triggered if the specified minimum chilled water loop temperature is less than 41°F. A QA error should be triggered if the minimum loop temperature is less than 36°F. If attached ice storage systems are detected, the warning limit should be reduced to 25°F and the error limit should be reduced to 20°F.

Energy model QC actions, however, require simulations to be executed, since QC is applied to a product (or results in our case). Put simply, model QC efforts focus on identifying defects in products. QC systems require defects (and defect tolerances) to be pre-defined. Differing energy model use-cases will require different defect tolerances, and creators of energy models need to contextually understand customer expectations of the specific use-case to adequately set defect tolerances. A basic example of this would be unmet load hours, whose allowable range is specified by ASHRAE 90.1 Appendix G. A model QC action might be to generate a warning message after the simulation which classifies if unmet hours tolerance has surpassed the defect tolerance. Model QC activities should clearly identify the defect and, when possible, encapsulate best practices and mechanisms for defect elimination.

An example of a specific energy end-use QC test might be a check of the sum of energy consumed by all heating and cooling pump motors as a fraction of the whole building electric energy consumed. The check would only be applicable to models having both central chilled and hot water systems, and the check would need to validate that an annual run period was simulated. A climate specific pump energy` fraction QC warning level might be set at a level of $(0.0000141*HDD_{65}) + (0.0000284*CDD_{50})$. A climate specific pump energy fraction QC error level $(0.0000159*HDD_{65}) + (0.0000326*CDD_{50})$.

In March of 2011 Rocky Mountain Institute (RMI) hosted the Building Energy Modeling Innovation Summit in which key stakeholders in the Building Energy Modeling (BEM) community were invited to come together to work toward breaking down perceived key barriers to wider adoption of BEM practices (RMI, 2011). The participants were divided into five breakout groups to brainstorm around specific topics. Of the five groups, four mentioned Quality Control or Quality Assurance as a specific goal that is currently lacking in the industry, and the Support and Resources group went so far as to develop a few key points toward a Quality Control Framework that would increase quality in individual models and the perception of quality from outside observers and stakeholders. The primary thrusts of their recommendations included development of reporting standards for model inputs and outputs, exposure of modeling assumptions and default values, and comparison of inputs to statistical standards to identify suspicious values. The summit participants repeatedly pointed to the prospect for advances in QA and QC activities to lessen the perceived and proven lack of prediction repeatability between individual practitioners. Despite this forward looking summit, significant advances in a standardized use of QA and QC processes in energy modeling are still forthcoming.

USE OF QA/QC IN OTHER DISCIPLINES

Quality Assurance and Quality Control activities have a long history in disciplines such as manufacturing and software development. One reason American auto manufacturers were losing market shares in the 1980 s was a lack of focus on efficient production of quality vehicles (Huffman, 2010). The purpose of QA/QC for automotive assembly is to improve manufacturing efficiency while providing customer satisfaction and preventing costly vehicle recalls. A basic automotive QA/QC process begins with a design team determining a set of specifications and tolerances for the vehicle including key performance and cost targets. Next, designers iterate to develop a prototype vehicle that may meet the specifications. A set of prototype vehicles are then constructed. These vehicles are heavily instrumented and vehicle performance is determined (tested) under various controlled or simulated extreme conditions, such as differing road surfaces, differing exterior environments, wind tunnel and loading conditions. Once all specifications are equitably achieved, tooling and assembly methods are configured for efficiently building production quantity automobiles. During automotive production, QC tests are often applied to samples of a batch of products - Individual assemblies, entire subassemblies, or the finished vehicle. These samples may receive rigorous levels of QC testing ensuring the assembly or sub-assemblies are defect free before allowing the sampled population to proceed to the next step of the assembly process. For example, after initial engine block machining, spot inspections are made of the bored cylinder wall dimensions of randomly selected engine blocks. A bore diameter found out of tolerance represents a defect, and if a defect is found, it is dealt with before the engine block receives additional machining. Some automotive QC tests such as frame weld testing, shaft balancing and dimensional tolerance testing are automated, while others still involve human engagement, involving visual inspection and interpretation. In both cases, it is important to note that the QC tests are not subjective - both the measurement system and criteria (tolerance) for a defect are pre-defined. The frequency and amplitude of QC test failures are analyzed by dedicated QA staff to cost effectively improve assembly procedures. For example, observed patterns in cylinder wall diameter defects may lead to increasing the number of passes of the carbide cutting tool, increasing the frequency of changing the cutting

tool edge, or other process improvements. The focus in manufacturing is to eliminate product defects requiring re-work , such that every product is produced in a linear and efficient manner.

Software development practices have been evolving an integrated approach to QA and QC. Unlike the material constraints of the automotive production line, software development is a fluid and constantly changing practice. As such, quality assurance in software development is focused on function rather than mechanics. There are many techniques, tools, and paradigms that are applied to software development QA, but one of the most common and most applicable to BEM processes is test-driven development (TDD) backed by a version control system and an automated testing mechanism. TDD is a design process in which, before any code is written for a new feature or process, a set of automated tests are developed that detail how it should behave. The automated testing system will declare that all of these tests fail, at first, as no code has been written to make them work correctly. Then, the software is developed until all of the tests attain a passing status. These tests can function at several levels - from unit tests that test the simplest and core functions individually, to integration tests that test that the pieces of code are working together correctly, to acceptance testing that checks that the product is working at the highest level as a user expects it to work. Finally, a system of regression testing is implemented that repeats all of these tests whenever a change is made to any part of the code, to ensure that the changes did not break an unexpected but related block of code elsewhere. These constant automated testing procedures are in place to guard these very complex systems against slow and sometimes unexpected degradation during development. Software development is characterized by the fact that many people are working on the same code base at the same time. For reasonably large projects no one developer can fully understand the system in detail as a whole. The potential is always there that one block of code may become broken by code changes of code in an unrelated area. The full TDD process and in particular regression testing helps to provide consistent checks for unexpected consequences of changes.

Applying these paradigms to building energy modeling is complicated by the unusually internally interconnected nature of building energy models. It is possible, if not common, to produce a model that has a correct prediction but that has many internal errors that are mutually self correcting. Is is also possible to have models that have no defects on the object and system level and interactions of the systems produce incorrect predictions. Additionally, many QC tests are a comparison between two models, a baseline and proposed, a complication not found in other disciplines. Because of these issues, application of QA and QC activities to building energy models needs to be applied over several hierarchical levels of the model by an automated system to be effective.

BUSINESS AS USUAL QA/QC OF ENERGY MODELS

Observed practices of the standard of care of QA/QC applied by the majority of energy modelers today can be characterized as highly variable, highly inconsistent (non-standardized), lacking transparency and often failing to create sufficient records to support auditable processes.

Popular simulation engines such as DOE2.2, EnergyPlus and DOE2.1e lack built-in features for supporting automated model QC. These engines offer differing native capabilities for parametric manipulation and generating both standard and user-defined reports - which are often post-processed by users to support model QC efforts.

eQUEST provides an embedded quality control report (see figure 1). The eQUEST Quality Control Report organizes the QC tests and provides color highlighting of two levels of potential model defects - errors or warnings. Defect tolerances are visible to users but are not exposed for editing. The companion eQUEST/DOE2.2 support document describes a number of additional QC checks and directs users to specific DOE2.2 output reports where the checks can be manually performed.

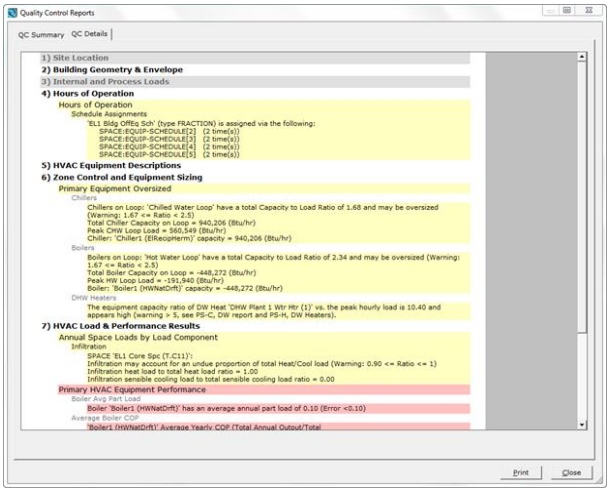

Figure 1 - eQUEST QC Checks

Other popular proprietary energy modeling tools such as Carrier HAP and Trace 700 also provide mechanisms for developing standard and custom reports that can be post-processed for QC tests. These tools also lack common mechanisms for standardized report generation, and can require significant resource investment to develop and automate intelligent QC post-processing of custom generated results.

Even though these platforms provide limited QA/QC implementations, aside from spreadsheet based solutions, there are no mechanisms for deploying a single QA/QC tool across multiple platforms. Providers of QA/QC services who need to be tool agnostic, such as utility program administrators and USGBC, are forced to review model outputs manually - which is costly and error prone. Rising administrative costs associated with manual model QC reviews put significant pressure on overall cost effectiveness, making energy modeling cost effective only when deep levels of program savings are targeted. Utility Program Administrators have responded by investing in automating model QC, often using common spreadsheet tools (Vega, 2014) as both the deployment and implementation layer. While functional, these QC frameworks can inadvertently contain errors in programing logic that are difficult to ascertain, use poor data structures for data storage and manipulation, and have poor data security and control features even if best practices for spreadsheet structure and documentation are adhered to.

In 2012, in an attempt to alleviate this, COMNET(R) published a software tool agnostic .xml schema (COMNET, 2012) for transferring limited (but common) energy model results using a standardized data format. COMNET also launched a fee based service for uploading COMNET.xml files, processing them, and providing standardized QC feedback to modelers and automated data transfer to USGBC for LEED data submissions, when applicable. The specific QC tests that COMNET intended to process were not published. This service is no longer functional (COMNET, 2012), and work on expanding the COMNET xml schema appears to be stalled. The majority of the schema definitions have been incorporated into the BuildingSync xml schema (NREL, 2016).

Beginning in 2011, RMI developed and published a series of simulation tool agonistic resources aimed at practitioners (RMI, 2016). These resources included checklists for supporting energy model quality assurance and suggestions for defect criteria for several building performance metrics. These resources were distributed via a combination of spreadsheet templates, pdf documents and executable software tools. The latest tool, an application for manipulating energy model weather files, was released in October of 2015.

In 2015, DOE published the BuildingSync xml schema. The BuildingSync schema expands beyond the reporting elements included in the original COMNET .xml, and may emerge to support the data transportability needs for software tool-agnostic QC to be achieved. These schema do not include QA or QC testing specifications, they are attempts at creating a common data transfer format for building energy models.

Energy modeling practitioners, lacking standardized QC systems, have resorted to building simple or elaborate custom systems to meet their own needs. Many of these solutions also use spreadsheets as the implementation layer. While spreadsheet based solutions simplify distribution and internal development effort, they complicate documentation, debugging, distribution and maintenance efforts and are generally a poor implementation choice for scaling flexible and dynamic software to a large community of users.

Characterizing current practice of Energy Model QA activities remains difficult as it is highly fragmented and characterized by many custom and private solutions. Based on experience and discussions with peers, the range of QA activities (scope and depth) implemented by different organizations remains broad. Even simple and routine tasks such as archiving and documenting variations during model development, appear to be done in an ad-hoc fashion. Performing model development in a manner that support a QA Audit is not a concern for the majority of today's production modelers. Like QC, much work remains to standardize commons model QA methods that could be distributed to communities at scale.

AVAILABLE FRAMEWORKS FOR AUTOMATING ENERGY MODELING QA/QC

The open source and community involvement that surrounds the EnergyPlus ecosystem makes it an ideal place to stage a standardized QA/QC project. Current available platforms that can support automated QA and QC tools include OpenStudio (DOE 2016), Params (BigLadderSoftware 2016) and eppy (Philip, 2016). Because OpenStudio, Params, and Eppy enable scripted routines to be applied to a building energy model an automated audit QA and QC tests can be written and run on an arbitrary model in any of these frameworks. However, test instantiations are framework dependent - a script developed to run in the OpenStudio

environment (for instance) is not applicable to a Params model or a model accessed through an eppy interface.

The US Department of Energy s OpenStudio platform includes a scripting feature called Measures, which can be applied by users to specific OpenStudio models as part of model development workflows by using the Ruby programming language (Roth, et al., 2016). One type of Measure, a Reporting Measure, is not allowed to alter model properties, but is allowed to query the input model and query and transform any simulation results. Reporting Measures allowing model QA/QC procedures to be both encapsulated and codified in a consistent way that eliminates much of the human error inherent in document interpretation and data transcription, providing a much more scalable solution than spreadsheet QC deployment. Reporting Measures can also be stored and transparently distributed via the simulation tool-agnostic Building Component Library (BCL). For example, the XCel EDA Reporting Measure and QAQC measure, available on the BCL, describes a number of standardized defect identification checks that can be applied to a model, including examining model end-use EUIs and Unmet Hours. The quantity and quality of these standardized checks can easily be extended by users. Reporting Measures represent an example of an automated QC workflow that can be applied to OpenStudio models which can significantly reduce the time associated with manual model QC review. In addition to QA/QC measures available from the BCL, to coordinate and minimize duplicate efforts, a publically editable roadmap document is available for users to view NREL s prioritization efforts for extending the QA/QC framework with additional standardized QC tests (NREL, 2015).

Other open source EnergyPlus based modeling frameworks that could support an automated QA/QC framework include Params and Eppy. Params is a templating engine built to work with EnergyPlus models and provides parametric control over the manipulation of models and the execution of simulations. Params uses embedded ruby commands to accomplish the templating and parametric model manipulations. Eppy is a scripting library for working with EnergyPlus idf files and simulation outputs written in Python. It allows for scripted access and manipulation of the model and the simulation outputs.

DESIRED PRACTICE OF QA/QC FOR ENERGY MODELING

When a robust and well accepted system for QA and QC for building energy models is built it will meet the criteria of being transparent, repeatable, and reproducible, be tool agnostic, and be operable across multiple platforms. Its execution will be automated and there will be a public repository of tests (or test data) that all vendors, practitioners, and institutions can access freely. BEM platforms that follow closed and proprietary practices can implement tests in their own systems and still reference the pubic standards, and still maintain transparency in their intentions and standards. BEM platforms that are open with source code can publish the scripts that are the instantiation of each test. The details of our proposed QA and QC criteria follow, which we hope will constitute an early sketch of the system.

Transparency in the sense that all test methods and the pass / fail criteria should be fully available for review and comments by any and all interested parties.

Repeatability and Reproducibility in the sense that the all tests should be non-ambiguous so that the repeated application of the same test to the same system will get the same result and that anyone applying the test will be able to reproduce the same answer.

Automated execution in the sense that using a QC system should require very little effort from the user. A suite of tests is selected and a simple one-click process causes them all to be applied and the results reported. The user should not need to retrieve simulation results manually, handle post processing data management or apply multiple processes to run the tests. Without this level of automation the criteria of repeatability and reproducibility are weakened. Level of automation and ease of use is left to the individual developers of software tools and frameworks that support the BEM industry. The test specification outlined next informs the development and sharing of individual QA and QC tests. A higher level executive program is required to achieve full automation. It would require as input a model and a set of references to publicly available QA / QC tests, and it would perform all of the tests on the model and produce a report. Using such a system individuals are enabled to develop collections of QA / QC tests that they can apply to their own models, and larger institutions can develop and publish the standard collections of QA / QC tests that models submitted for review are expected to satisfy.

The test requirements should be located in a public repository that allows for public access and public review. Users should be able to share their favorite tests. Reviewing agencies should be able to distribute their specific suite of tests. The descriptions of the tests should be tool agnostic with individual implementation

left to the tool vendors. By separating the implementation from the knowledge base the testing criteria can be stated clearly and agnostic of programming languages and vendor specific mechanics. A wiki or the BCL are a good choices for hosting the libraries of tests and test scripts. They provide a publically accessible and updatable format for sharing and updating test ideas, for hosting scripts that are implementations of the tests, and APIs for automated interactions. A useful public example of a similar system can be seen in the Rosetta Code Project (rosettacode.org , 2015) which is a public repository of instantiations of common computer programming functions in every programming language. For instance, on the page that demonstrates a simple for loop there is a description written that describes a for-loop, how it should function, and how to demonstrate that is it working correctly, followed by canonical examples of for-loops in 148 different programming languages.

Specification of tests should be standardized so that searching and references to tests are repeatable. Each test should have at minimum the following attributes:

1. Unique Identifier: a uuid or ID number
2. Label: a few words to help users understand what it does
3. Test Level: Object, System, Whole Building.
4. Description: A paragraph explaining what the test is, how it is performed, and what the expected ranges and test / fail criteria are, along with any references or backing documentation.
5. Technical Description: A description written as pseudo code of the test written as a specific recipe that could be implemented in a specific language but that not language specific.
6. Implementations: Downloadable scripts that are specific implementations of this test.

CONCLUSION

The RMI Energy Modeling Summit identified the importance of model QA/QC, and many individuals and organizations since have taken steps forward to build infrastructure for executing model QA/QC. However, an overall vision and strategy for coordination of resources to prevent duplicate and competing efforts appears is lacking. Achieving a future where energy modeling predictions are accepted by a broad community of users as the best available technology for determining energy usage will require improvements in developing and automating model QA/QC processes. Any broadly accepted open source project requires that there is a person or entity that takes on the responsibility of parenting the project as it is built and as it grows. This project requires a home and long term support more than most. Possible candidate organizations include ASHRAE, as a QA/QC process for BEM support is similar to a published standard, but maintenance of live databases and websites has not been much adopted by ASHRAE. If in the BCL framework, it is possible that DOE could take a leading role. IBPSA, as a well trusted and neutral resource, would be a prime candidate for leading the way in the development and hosting of a BEM QA and QC system.

Examples of a QA and QC specification:

ID	6da56f4f-d2d1-4b84-a6c6-d657527d2c4d
Label	Chilled water loop minimum loop temperature
Level	System
Description	Unless an ice storage system is attached to the loop, a warning is triggered if the specified minimum chilled water loop temperature is less than 41 [F]. An error is triggered if the minimum loop temperature is less than 36 [F]. If attached ice storage systems are detected, the warning limit is reduced to 25 [F] and the error limit is reduced to 20 [F].
Technical Description	Check Model: is there an attached ice storage system? Check Model: what is the water loop minimum temperature setpoint? If: there is no ice storage system and the setpoint is less than 41 [F] send a warning. If: there is no ice storage system and the setpoint is less than 36 [F] send an error. If: there is an ice storage system and the setpoint is less than 25 [F] send a warning. If: there is an ice storage system and the setpoint is less than 20 [F] send an error.
Implementation	6da56f4f-d2d1-4b84-a6c6-d657527d2c4d.rb 6da56f4f-d2d1-4b84-a6c6-d657527d2c4d.py 6da56f4f-d2d1-4b84-a6c6-d657527d2c4d.params

ID	800f7e62-0168-440f-9eba-330898e8dab3
Label	Pump motor to whole building energy ratio
Level	Whole Building
Description	Check if the sum of energy consumed by all heating and cooling pump motors divided by the whole building electric energy consumed meets reasonable criteria. Only be applicable to models having both central chilled and hot water systems. If pump energy to whole building energy use is greater than (0.0000141*HDD65) + (0.0000284*CDD50) a warning is triggered. If pump energy to whole building energy use is greater than (0.0000159*HDD65) + (0.0000326*CDD50) and error is triggered.
Technical Description	Check Model: Central chilled and hot water systems present? If: both systems are not present send Not Applicable message and end test. Check Model: What is HDD65 and CDD50 Check Simulation Results: What is total pump energy divided by total building energy use? If: total pump energy divided by total building energy use greater than (0.0000141*HDD65) + (0.0000284*CDD50) trigger a warning. If: total pump energy divided by total building energy use greater than (0.0000159*HDD65) + (0.0000326*CDD50) trigger an error.
Implementation	800f7e62-0168-440f-9eba-330898e8dab3.py 800f7e62-0168-440f-9eba-330898e8dab3.rb 800f7e62-0168-440f-9eba-330898e8dab3.params

REFERENCES

DOE, 2016. OpenStudio Project Department of Energy, Office of Energy Efficiency and Renewable Energy. https://www.openstudio.net/

Huffman, John Pearley 2010 How the Chevy Vega Nearly Destroyed GM Popular Mechanics. http://www.popularmechanics.com/cars/news/vintage-speed/how-the-chevy-vega-almost-destroyed-gm

NREL, 2015. MasterOpenStudioMeasureSpreadsheet https://docs.google.com/spreadsheets/d/1QUP6Nz1X6gdOeNEIBuxByONHOz5m__tTpr-llp2avp0/edit#gid=897267309

NREL, 2016. BuildingSync XML Schema https://buildingsync.net/index.html

Philip, 2016. A Scripting language for EnergyPlus idf files. Santosh Philip, https://github.com/santoshphilip/eppy

RMI, 2011. Post Report from the BEM Innovation Summit 2011. Rocky Mountain Instititute, Basalt, CO. www.rmi.org/Content/Files/BEM_Report_FINAL.pdf

RMI, 2016. Modeling Tools Resources 2016. Rocky Mountain Instititute, Basalt, CO. http://www.rmi.org/ModelingTools#Additionaltools

Rosettacode.org, 2015. http://rosettacode.org/wiki/Rosetta_Code

Roth, et al. (2016) There s a measure for that! Energy and Buildings, Volume 117, Elsevier Press, April 2016.

Vega, K., Beaulieu, S., Karpman, M. 2014. Lessons Learned: Performing QC on Energy Models aceee.org/files/proceedings/2014/data/papers/1-731.pdf