Rulesets are Great. Certified Rulesets are Greater.

Amir Roth, Building Technologies Office, U.S. Department of Energy

ABSTRACT

Whole building energy modeling is the basis of building-energy efficiency processes including performance-path code compliance such as ASHRAE 90.1 “Appendix G”, green labeling via programs like LEED, and documentation for up front incentives. A key step is the creation of a code minimum “baseline” model from a model of the proposed building, to which the latter is then compared. Manual baseline generation is a detailed, mechanistic, error-prone process. Fortunately, it is slowly giving way to automated implementations commonly referred to as “rulesets”. CBECC automates California's Title 24 ACM, while ASHRAE 90.1 Appendix G rulesets are available via several applications. Unfortunately, ruleset-generated baseline models are manually reviewed along with their corresponding proposed building models. Manual baseline model review is also a detailed, mechanistic, error-prone process.

This paper argues that the solution is to certify rulesets, allowing baseline model results to be accepted without additional review. Certifying rulesets is simpler than certifying BEM engines. Rulesets are human defined and so a single transparent implementation can be certified by a combination of inspection and fiat—California availed itself of these benefits by certifying CBECC in this way. From there, other rulesets, including proprietary ones, are tested and certified by comparison to the reference. This paper describes the mechanics of the “bootstrapping” certification process and proposes a certification program for ASHRAE 90.1 Appendix G rulesets based on the OpenStudio Standards gem. It also argues that other BEM tasks that benefit from automation can similarly benefit from certified automation.

The Setup

Whole building energy modeling (BEM) is a multi-purpose tool that supports building energy efficiency. BEM directly influences building energy performance via applications like architectural, HVAC, and control design. A significant amount of modeling time is spent on use cases that don't directly improve energy performance but support financial or regulatory transactions including demonstrating compliance with energy efficiency codes like ASHRAE 90.1, documenting above code energy performance for green certificates like USBGC's LEED, and calculating savings for up-front energy efficiency incentives.

The performance documentation use cases for code compliance, certification, and incentive are conceptually and physically based on the same procedure, a performance comparison between the proposed building and a theoretical version of the same built to prescriptive code specifications referred to as the baseline building. If the proposed building meets the performance of the baseline, or exceeds it by some threshold, the proposed building complies with code, receives a certificate, or qualifies for a payment. This comparative self-referential setup accommodates the great diversity that exists in buildings. The use of BEM allows the performance of a theoretical building to be evaluated and for all variables ancillary to the comparison, e.g., weather, to be controlled.

Because of the regulatory and financial nature of these applications, the relationship between the proposed and baseline building models needs to be precisely defined. Documents such as ASHRAE 90.1 Appendix G "Performance Rating Method" (ASHRAE 2013) and California Title 24's "Alternative Calculation Method" (CEC 2015) contain detailed context-sensitive descriptions of the transformation that must be applied to a proposed building model in order to generate a baseline. These transformations are comprehensive, affecting window and shading geometry, constructions, HVAC systems, and operational settings and schedules. In some cases, transformations must also be applied to the proposed building model *in situ*.

Energy modelers spend a large fraction of their working hours performing these mechanistic transformations manually, severely cutting into time and budget that could be spent on tasks that directly support building energy efficiency. On the other side, code officials, certificate reviewers, and energy-efficiency program administrators spend a large fraction of their time reviewing pairs of proposed and baseline building models to ensure that the prescribed transformations are interpreted and applied correctly and that those are the only transformations performed, i.e., that "cheating" is not taking place. Manual processes on both sides degrade productivity and project throughput (RMI 2011).

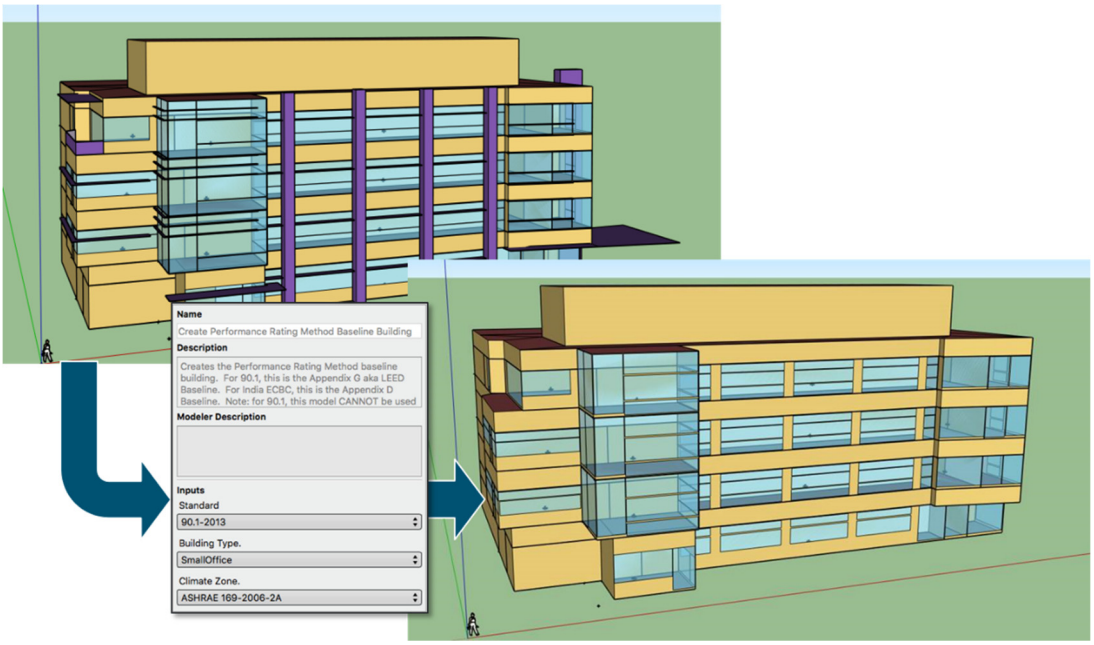

Figure 1. ASHRAE-90.1 Appendix G baseline automation using the "Create Performance Rating Method Baseline Building" OpenStudio Measure found in the OpenStudio-Standards gem. The visible impacts of the procedure include removal of shading devices and resizing of windows to meet the glazing ratio threshold. Credit: NREL.

Automation enhances productivity in any field and has recently caught on in BEM. Computers are excellent at performing detailed mechanistic tasks and doing so tirelessly, consistently, and without error, innocent or intentional. Multiple vendors have fully or partially automated the baseline model derivation procedure (DesignBuilder 2018; Trace 2018). Figure 1 shows one of these implementations, the "Create Performance Rating Baseline Building" OpenStudio Measure (Goldwasser 2016; Roth 2016). Automated baseline implementations are often referred to as "rulesets" reflecting their mechanistic nature and distinguishing them from physics simulation. Rulesets transform or query the model and its results; they are (in theory)

agnostic of the simulation engine used. Baseline automation frees energy modelers to focus on more creative modeling tasks that more directly inform and enhance building performance.

The crux of the matter—model review productivity

Baseline automation does not similarly benefit model reviewers. Even automatically-derived baseline models need to be reviewed alongside the corresponding proposed building model to simultaneously ensure correct ruleset application and absence of tampering. In fact, the argument can be made that baseline automation exacerbates the existing inefficiencies of model review. Model review is a critical BEM ecosystem service, one that is both under-appreciated and under-subscribed relative to modeling itself. For instance, the Green Building Certification Institute (GBCI) website lists 80 staff—although it claims over 300—that are responsible for a half-dozen international certificate programs including LEED, EDGE, and WELL (GBCI 2018). Anecdotally, model review contracts pay for between three to five hours of review per project insufficient time for a thorough job. If baseline automation increases modeler productivity and energy-efficiency project submission rate, already-thin review resources will be stretched even further or be forced to grow and cannibalize program and project budgets (Rocky Mountain Institute 2011; Elling 2014).

The issues of productivity, effectiveness, and quality of model review have received some attention in recent years (Beaulieu 2010; Vega 2014) including several proposals and actively deployed projects for automating aspects of model quality-assurance/quality-checking (QA/QC) (Elling 2014; Balbach 2016). Model QA/QC is a broad and significant challenge that can benefit from these and other techniques. For the specific case of reviewing pairs of proposed and automatically-generated baseline models, there is a more direct and complete option—certify the baseline automation software such that its outputs can be accepted without manual review. Certified software can be combined with a tamper-resistance mechanism such as watermarking, digital signatures, or even … blockchain—or the tampering problem can be sidestepped by having the reviewer press the button on final baseline generation, simulation, and reporting—to create a trusted "end-to-end" solution. This paper discusses the theoretical and practical aspects of ruleset software certification.

Software Testing and Certification

The words "validation" and "verification" are often used to describe the process of ensuring software fidelity to a set of requirements. These terms are aspirational or statistical as there is no general-purpose way of "proving" that a piece of software obeys a specification or is otherwise "correct". The futility of this pursuit is one of the seminal results of computer science (Turing 1936) and it is evident in the number of bugs that exist in production software. In practice, software is verified by the dual actions of source code inspection and input-output testing, i.e., checking that when given a certain input the software produces the corresponding expected output. The greater number and variety of tests a software passes the more correct or robust it is deemed. A large test suite is needed to achieve high code coverage, i.e., to exercise all relevant parts of the source code. Software certification programs essentially consist of suites of test cases. A certification sticker implies that the software has passed the test suite within an acceptable tolerance. The problem of gaining confidence in rulesets is a (third-party) certification problem and therefore a testing problem.

The primary challenge faced by software certification programs is developing a sufficiently large test suite that achieves meaningful coverage. Static coverage, i.e., simply hitting every block of code once, is necessary. A meaningful test suite also achieves high dynamic coverage, i.e., exercises different sequences of blocks and logic paths through the code. Here there are two sub-problems as well: i) collecting a large number and variety of input cases; and ii) creating the corresponding output for each input case. Inputs can be collected from real-world projects, generated randomly from templates, or a combination of both. Generating matching outputs is trickier. Unless the certifying organization has a "gold standard" reference implementation of ruleset in hand, outputs must be generated manually from inputs. This re-introduces the human productivity bottleneck as well as human interpretation and error issues into the process.

A Partial Case Study: California Title-24 ACM Ruleset

When the California Energy Commission (CEC) set about to update its Title-24 energy efficiency code for 2013, the Alternative Calculation Method (ACM) procedure for demonstrating compliance, and the software implementation of the ACM, it recognized these challenges and made several decisions to alternately address or avoid them (Brook 2012).

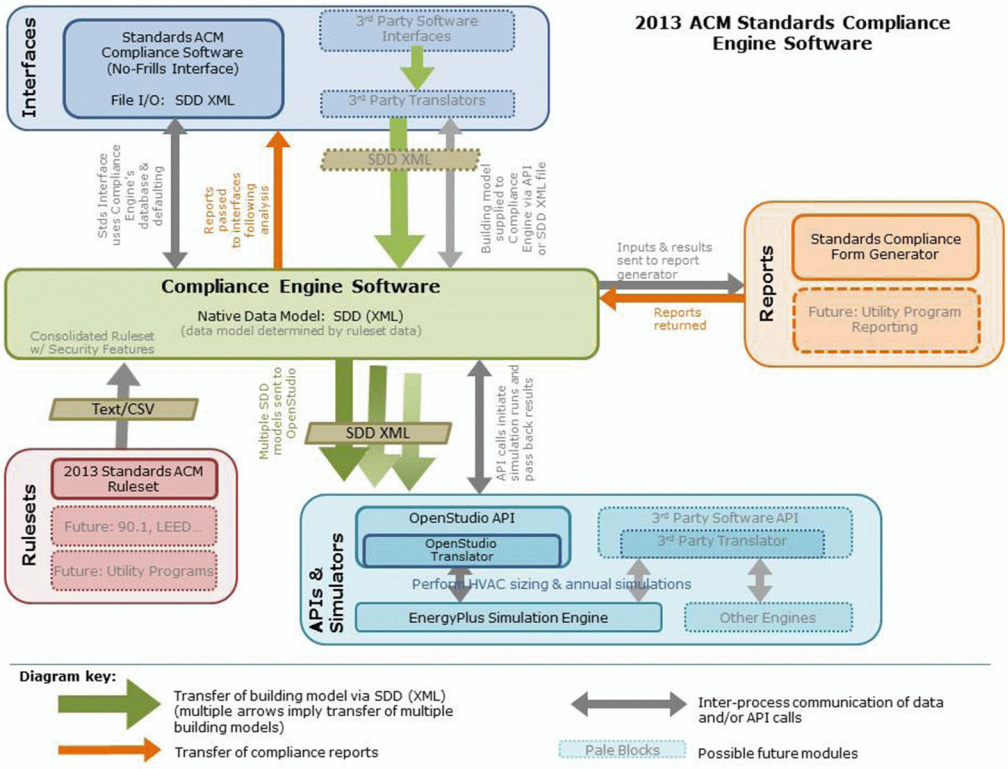

Figure 2. CBECC-Com software architecture. Shown on the bottom left, rulesets are separate from BEM engines like EnergyPlus which are shown on the bottom right. The main part of CBECC is the "compliance engine". It takes as input a proposed building model in the SDD format and rulesets written in the rules language; it applies the rules to the input model to produce a modified proposed building model and a baseline building model, both in SDD. Credit: CEC.

The CEC sidestepped the issue of certifying third-party software by developing its own ruleset, California Building Energy Code Compliance (CBECC); certifying it by a combination of inspection, testing, and fiat; and mandating it for all compliance transactions. Any application that wants to provide Title-24 compliance functions must use the CBECC ruleset via a custom input-output schema called the Standards Data Dictionary (SDD). The CBECC software architecture is shown in Figure 2 above (Criswell 2014; CBECC-Com 2018). This architecture has supported the 2013 and 2016 codes and will support the 2019 code.

To reduce the burden of manually generating test cases for CBECC, CEC designed CBECC to closely mirror the Title 24 ACM specification, shifting more of the verification burden to inspection or even (ruleset) design. The SDD schema mirrors the nomenclature of the ACM. The rules themselves are written in a domain-specific ruleset language that embodies and internalizes the logic of applying rules to models (Brook 2012). An example of a CBECC rule for exterior north facing constructions is shown in Figure 3 below. They are open-source and available for public review as is the ruleset interpreter. A final and important ingredient was a revision of the ACM itself to remove ambiguities and generally to facilitate automation—rules that were difficult to implement in software were rewritten or discarded. The side-by-side development of code specification and its computer implementation is a key process innovation.

```
ConsAssm    "WallSFU0.064"
   CompatibleSurfType = "ExteriorWall"
   ExtRoughness = "MediumRough"
   ExtSolAbs = 0.7 ExtThrmlAbs = 0.9
   ExtVisAbs = 0.8 IntSolAbs = 0.7
   IntThrmlAbs = 0.9  IntVisAbs = 0.8
   SpecMthd = "Layers"
   MatRef = ( "Stucco - 3/8 in.",
              "Gypsum Board - 5/8 in.",
              "Compliance Insulation R13.50",
              "Gypsum Board - 5/8 in." )
```

```
RULELIST "AssignNorthWallConst" 1 0 1 0
;*****************************************
"Assign construction to new north facing
ExtWalls on conditioned spaces"
ExtWall:ConsAssmRef = {
  if( Status = "New" .AND.
      (RealAz > 315 .OR. RealAz <= 45) .AND.
      Spc:CondgType != "Unconditioned" )
  then
    ; Assign library construction to wall
    RuleLibrary( ConsAssm, "WallSFU0.064" )
  else
    ; Otherwise, skip wall
    UNCHANGED
  endif
  }
END
```

Figure 3. CBECC rule for envelope constructions for exterior, north facing walls. The snippet on the left shows the construction assembly data structure. The snippet on the right shows the rule that applies this assembly to all exterior north facing walls. The ruleset interpreter "pattern-matches" for Exterior Wall objects and applies the if-then logic to their construction assembly field. Code credit: SAC Software Solutions.

Components of a Certification Solution

The CBECC case study is "partial" because the California context is unique, and the mandatory approach taken by the CEC cannot be easily replicated elsewhere. In California, a single entity controls energy code development, energy code adoption and enforcement, and the certification (and sometimes creation) of compliance software. Outside of California, these functions are distributed among ASHRAE and ICC on one hand and states and municipalities on the other. No individual entity has enough clout to mandate a single software implementation. And outside of the Energy Department—which supports code-development organizations and states but is not a direct participant in adoption and enforcement—no public entity has enough resources to support the development and maintenance of such an implementation. In general, a solution that supports multiple implementations is needed.

The CBECC case study does contain the components required to support a general, scalable certification solution: i) a standard open schema for specifying test case inputs and outputs; and ii) a transparent gold standard reference implementation that can be used to automatically generate a large suite of tests. If it wanted to certify alternative implementations, the CEC could use CBECC to generate a suite of test cases against which those would be tested. That CBECC is open-source plays a key role in operating the certification process and in helping candidate implementations obtain certification. Discrepancies will invariably arise during the testing of candidate rulesets. These will be due to interpretation mistakes or bugs in the candidate ruleset, or perhaps in the CBECC reference ruleset. That CBECC is open-source theoretically allows ruleset vendors to discover the source of the discrepancy, correct the error if that error is in their own code, or petition the CEC if she believes the error to be in CBECC.

A Proposed Case Study: ASHRAE 90.1 Appendix G Ruleset

As a component of code-compliance outside of California and of LEED certification everywhere, ASHRAE 90.1 Appendix G arguably gets more use than Title-24 ACM. What would it take to develop and operate a certification program for Appendix G rulesets?

From a technical standpoint, not very much. The necessary pieces are an open schema for specifying test cases and an open-source reference ruleset implementation. The latter already exists in the form of the "Create Performance Rating Method Baseline Building" Measure, part of the OpenStudio-Standards gem. A gem is a collection of Ruby code and related resources. The Ruby scripting language is more compact than compiled programming languages like C++ and it operates on the OpenStudio API which itself provides some convenient abstractions that reduce verbiage and enhance readability. OpenStudio nomenclature for space types and attributes matches the nomenclature used in ASHRAE 90.1 and the Standards gem tabular data—which is housed in Excel Spreadsheets—mirrors the table structure used in the standard. The Measure reads and outputs OpenStudio Models, i.e., OSM files. The OSM schema is public and used by a number of tools. OpenStudio also reads a number of other schema such as gbXML (gbXML 2018) and if need be could be extended to write gbXML and output baseline models in this format.

The "political", organizational, and administrative requirements are greater. First, entities that use Appendix G for regulatory and financial purposes such as state energy offices and the Green Building Certification Institute (GBCI) must see the benefit of certified Appendix G rulesets, streamline their review procedures for submissions that use these certified rulesets, and ultimately perhaps require use of certified rulesets. Next, the OpenStudio Appendix G Measure, or some other open source implementation, must be certified as the reference implementation to "seed" the certification process. This part should be done by a group intimately familiar with Appendix G, perhaps even ASHRAE's Appendix G committee itself. This task is likely to take some time and involve both inspection and testing. A benefit of having the Appendix G committee performing it is that it would see firsthand where standard language leads to ambiguity or difficulty in automated implementation and would be moved to clarify or simplify Appendix G to accommodate automation as the CEC did with the ACM. Of course, the aforementioned regulatory entities must also approve.

The final piece will come from vendors, a number of whom already have Appendix G rulesets. To obtain certification, vendors will need to import and export the test suite schema—potentially a non-trivial investment. A benefit of using gbXML is that many vendors already

import it; a possible drawback is that it is currently not expressive enough to accommodate Appendix G yet. Figure 4 below shows the certification procedure flow diagram. The certification body is responsible for the top, yellow part of the Figure, i.e., the open-source reference ruleset and the test suite. Ruleset vendors implement the bottom blue part.

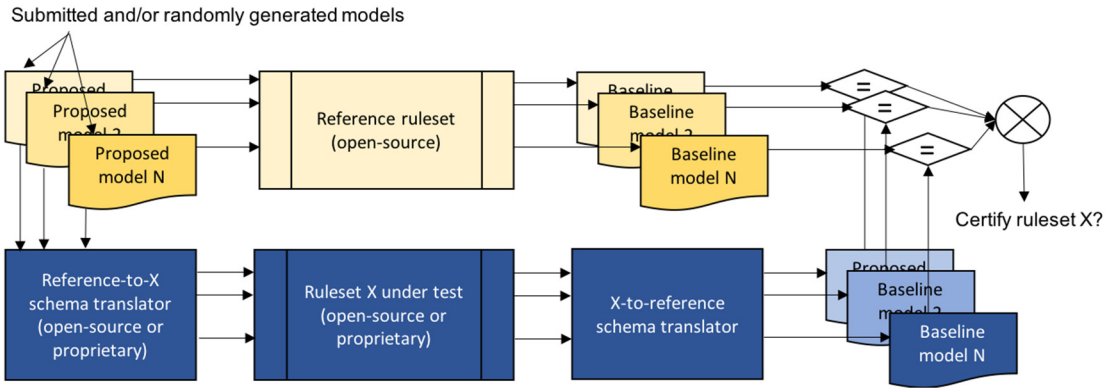

Figure 4. An open, scalable ruleset certification process. The process requires a "gold standard" reference implementation of the ruleset. Combined with a facility to generate proposed models, the reference ruleset is used to generate a large test suite against which alternative implementations can be compared. Although the process is "open", it can be used to certify proprietary ruleset implementations. The process is "scalable" because most of the work lies with ruleset vendors, not the certification entity.

Vendors may have some legitimate concerns about this process. "If I export to a common schema that others can import, am I not allowing my users to take their models to other software?" Either way, exports to the standard ruleset schema need not be public; vendors can keep these paths private and use them only to certify their ruleset implementations. Observers may also have some questions. "If vendors can import and export OSM or gbXML, why not just use the OpenStudio Standards ruleset?" There are many reasons vendors may prefer their own ruleset implementations. These may be faster than the OpenStudio implementation which is interpreted and thus slower than implementations written in compiled languages like C++, better in some other way, or simply better integrated with the rest of their software environment. As with the point above, vendors may not want to create even temporary files in open formats that would allow their users to take models to other software. Of course, vendors that don't have their own ruleset implementation can use the reference indefinitely, and until such a time as they develop their own.

Aligning financial incentives

The biggest challenge to standing up a ruleset certification program is not technical and has little to do with intellectual property, it is aligning the financial incentives of the relevant parties. The benefits of baseline automation accrue to modelers, a constituency large enough to motivate vendors to implement rulesets. The benefits of certified automation accrue primarily to model reviewers, a smaller constituency with weaker market pull.

By requiring projects to use certified ruleset software, or by giving preference to projects that use such software, reviewers can indirectly push vendors to certify their implementations. Reducing vendor certification cost—primarily creating import and export functions—can also spur certification program subscription. The use of popular schema would

help here as would the development of free libraries for reading, writing, manipulating such schema. The use of a small number of rulesets to support multiple programs also helps reduce both automation and certification cost—this was a primary motivation for ASHRAE 90.1 addendum BM which fixed the code baseline procedure to Appendix G-2004 for all code updates beginning with 2016 (Rosenberg 2013). Programs that require intensive model review may choose to invest in helping vendors implement automation and obtain certification to reduce their ongoing review costs

Discussion: Interoperability vs. Equivalence

Interoperability is a frequently invoked goal for software, including BEM. If all BEM software spoke the same universal language then a single implementation of any function would suffice, development productivity would increase by orders of magnitude, and enlightenment would be achieved. At first blush, it would seem that the BEM software world is small enough—at least in relation to other software worlds—to make interoperability seem achievable. In practice, the BEM software world is small enough so that complete chaos does not ensue even in the absence of interoperability. The range of BEM functions and services is still not broad and diverse enough to compel vendors to inter-operate just to take advantage of it all. Integration of external software takes effort, effort that vendors would often rather expend implementing functions themselves than provide users with easy paths to outside software, especially competitor's software. This dynamic was evident in California, where vendors were slow to integrate CBECC-Com into their applications, forcing users to recreate the proposed building model using the free CBECC-Com interface, undermining much of the efficiency gains of baseline automation. Instead of manually creating proposed and baseline models, modelers manually created two proposed models, one in their BEM software of choice and one in CBECC-Com. In another contrast with other software worlds, the BEM world also doesn't pervasively use “platforms” that enforce some degree of interoperability as a byproduct. OpenStudio is the closest thing the BEM world has to a platform and it is far from pervasive; a majority of applications and services don't use it.

With inertia and business considerations impeding interoperability, the BEM community should set its sights on a less ambitious but still worthy goal—testable equivalence. Different implementations don't have to be physically interchangeable (i.e., interoperable) in order for their results to be considered interchangeable; it is enough to demonstrate that they are functionally equivalent. Equivalence transmits confidence not only in functionality, but in “meta” properties as well. Regulatory and financial applications often require transparency. With testable equivalence, an implementation doesn't have to be transparent itself to support these applications, it must only be equivalent to one that is.

Discussion: Physics vs. Rulesets

The BEM community is quite familiar with software testing via test suites such as BESTEST and ASHRAE Standard 140, and with certifications such as the IRS list of qualified BEM software. These tests and certificates apply to BEM (physics) engines. For BEM physics engines, the gold standard reference implementation is physics. The input-output schema of physics is standard and universal—even if the units are not—and the number of test cases is infinite. But even carefully constructed physical tests can be quite noisy. A deterministic

simulation will always differ from a noisy physical experiment in some way, and when and where it does it is not always easy to determine why or whether the difference is meaningful. Although earthbound physics is mostly open-source at this point the subset that prevails in a given experiment, not to mention the noise, is not explicitly specified.

Rulesets are much simpler than physics. They are discrete and man-made, and gold standards can be created and declared. Ruleset testing is much simpler than physics testing. Determinism makes precise matches possible, virtual testing environments are noise free, and source code makes differences easy to track down. Ruleset certification is achievable. And fairly easily. Yet it is essentially unknown in the BEM world. Why?

Conclusions

A ruleset is a general term that describes a BEM procedure—a model transformation or query—that does not involve physics simulation itself. Baseline generation rulesets are the most common because baselining is the most commonly used transformation in financial and regulatory applications and because they save the most work. But as baselining rulesets become entrenched and baseline ruleset certification takes hold, modelers and model reviewers will begin to spend their time on other tasks. Like baseline generation, these tasks will first become modeler bottlenecks. Then they will be automated. Then they will become reviewer bottlenecks. Finally, they will be certified. "Create Performance Rating Method Baseline Building" is only one OpenStudio Measure. The Building Component Library contains over three hundred others. Many correspond to energy conservation measures (ECMs). Some correspond to reports. It is easy to imagine some of these being embedded into regulatory or financial applications and following this path. The ruleset box in Figure 2 contains not only the Title 24 ACM ruleset but also potential future rulesets that support utility programs. Baselining ruleset certification is only the beginning, but we do have to get started.

Acknowledgements

Thanks to RMI for the 2011 BEM Innovation Summit at which many of the challenges addressed in this paper were surfaced. Thanks to the CEC for championing the concept of open-source rulesets and their use in code-based applications, and for creating a system that can be extended as well as serve as a model for other systems. Thanks to Southern California Edison and TRC Companies for a September 2017 workshop on compliance software at which the ideas presented in this paper germinated. Thanks to many including Chris Balbach, Dimitri Contoyannis, Scott Criswell, Supriya Goel, Andrew Parker, Michael Rosenberg, and Greg Thomas for discussions and to some for comments on an early draft. Finally, thanks to reviewers for helping to refine the arguments made in this paper.

References

About Measures [Website]. 2018. Retrieved from http://nrel.github.io/OpenStudio-user-documentation/getting_started/about_measures/

ASHRAE Standard 140 [Website]. 2017. "Standard 140-2017: Standard Method of Test for the Evaluation of Building Energy Analysis Computer Programs (ANSI Approved)". https://www.techstreet.com/ashrae/standards/ashrae-140-2017?product_id=2001489

ASHRAE Standard 90.1 Appendix G [Website]. 2013. "ASHRAE Standard 90.1 Appendix G Performance Rating Method 2013" https://www.ashrae.org/technical-resources/bookstore/standard-90-1-appendix-g-2013-performance-rating-method

Building Component Library [Website]. 2018. Retrieved from http://bcl.nrel.gov/

Balbach, C. and D. Bosworth. 2016. "Automated Methods for Improving Energy Model Quality Assurance and Quality Control", Proceedings of ASHRAE and IBPSA-USA SimBuild 2016 Building Performance Modeling Conference, Aug. 2016.

Beaulieu, S., T. Rooney, and M. Karpman. 2010. "Modeling Energy Use of Green Buildings: Metric-Driven Quality Control". ACEEE Summer Study on Energy Efficiency in Buildings, Aug. 2010.

Brook, M., and S. Criswell. 2012. "The BEE Software Collaborative: An Open Source, Rule-Based Architecture for Building Energy Efficiency". ACEEE Summer Study on Energy Efficiency in Buildings, Aug. 2012.

CEC (California Energy Commission). 2015. "2016 Nonresidential Alternative Calculation Method Manual for the 2016 Building Energy Efficiency Standards", Nov. 2015. http://www.energy.ca.gov/2015publications/CEC-400-2015-025/CEC-400-2015-025-CMF.pdf

CBECC-Com Non-Residential Compliance Software [Computer Software]. 2018. Retrieved from http://bees.archenergy.com/

Criswell, S., J. Glazer, E. Hale, D. Contoyannis, and M. Brook. 2014. Functional Requirements for ACM Standards Compliance Engine Software, Version 1.0. San Francisco, CA: NORESCO.

DesignBuilder Software LTD – LEED and ASHRAE 90.1 Appendix G [Website]. 2018. Retrieved from http://designbuilder.com/leed/

Elling, J., L. Brackney, and N. Long. 2014. "Energy Design Assistance Program Tracker (EDAPT): A Web-Based Tool for Utility Design Assistance Program Management", ACEEE Summer Study on Energy Efficiency in Buildings, Aug. 2014.

EnergyPlus [Website]. 2018. Retrieved from http://energyplus.net/

EnergyPlus Input/Output Reference [Website]. 2018. Retrieved from https://bigladdersoftware.com/epx/docs/8-8/input-output-reference/

gbXML [Website]. 2018. Retrieved from http://www.gbxml.org/

Goldwasser, D., D. Macumber, A. Parker, E., Lee, R. Guglielmetti, and L. Brackney. (2016). "The Life Cycle of an OpenStudio Measure: Development, Testing, Distribution, and Application", Proceedings of the ASHRAE and IBPSA-USA SimBuild and Building Performance Modeling Conference, Aug. 2016.

OpenStudio [Website]. 2018. Retrieved from https://openstudio.net/

RMI (Rocky Mountain Institute). 2011. "Collaborate and Capitalize: Post-Report from the BEM Innovation Summit", Rocky Mountain Insititute, 2011. http://wwx.rmi.org/Content/Files/BEM_Report_FINAL.pdf

Rosenberg, M. and C. Eley. 2013. “A Stable Whole Building Performance Method for Standard 90.1”, ASHRAE Journal, May 2013.

Roth, A., D. Goldwasser, and A. Parker. 2016. “There’s a Measure For That!”, Energy and Buildings, Volume 117, Apr. 2016.

Trace 700 LEED Guide Tutorial [Website]. 2018. Retrieved from http://www.trane.com/commercial/north-america/us/en/products-systems/design-and-analysis-tools/analysis-tools/trace-700/trace-700-leed-guide-tutorial.html

Turing, A. 1936. “On Computable Numbers with an Application to the Entscheidungsproblem”, 1936.

Vega, K., S. Beaulieu, and M. Karpman, 2014. “Lessons Learned: Performing QC on Energy Models”, ACEEE Summer Study on Energy Efficiency in Buildings, Aug. 2014.