A BIM WORKFLOW FOR ITERATIVE AND INFORMATIVE ENERGY MODELING

Ryan Welch1, Amy Egerter2, Shanta Tucker2, and Christopher Connock1
1KieranTimberlake, Philadelphia, PA
2Atelier Ten, New York City, NY

ABSTRACT
A Building Information Modeling (BIM) workflow is developed to automate data transfer across software platforms and facilitate an iterative energy modeling practice. The workflow comprises a translation of Revit model geometry and parameters using a custom application for the automated export of an eQUEST input file. Revisions can be made in Revit and re-imported into a revised version of the eQUEST file. This workflow further utilizes a shared web-based Access database of project information, which includes energy analysis parameters such as ventilation rates, internal loads, and use schedules that can be synchronized with Revit model parameters and subsequently fed into the eQUEST input file. Ultimately the workflow could expand to support other text-based input files and platforms.

INTRODUCTION
The intention of this exercise is to reduce the amount of time and information lost from model sharing between architects and technical consultants, specifically in energy modeling workflows. This exercise is applied to a large, mixed-use project for a university client in which there are numerous stakeholders and aggressive environmental goals. The complexity of this project created the need to track data in a shared online database. Integrating the database with the Revit and energy models is intended to reduce data entry redundancy and increase the fidelity between platforms. This workflow was developed through a desire for iterative modeling as opposed to a more typical process of compliance modeling at project milestones.

Beginning with a design team charrette, KieranTimberlake (a 100-plus person architecture firm with a focus on research-integrated practice) and Atelier Ten (a 100-plus person practice of building services engineers, environmental design consultants, and lighting designers) identified iterative modeling as one means for developing quantitative building performance goals and compelling evidence for design decisions. As such, integrating the two teams and their respective workflows would be critical to the design. Prior to BIM development, the project’s detailed program requirements were captured in a central program database (Figure 1).

A custom application was subsequently developed to transfer data between that master database and the rapidly-evolving models. The program database captures features that are essential to space planning such as desired room dimensions and adjacencies, as well as features essential to energy modeling including lighting power densities, occupancy schedules, and ventilation rates. Therefore, the database provides a natural framework for linking the spatial BIM information together with zonal information contained in the program database. This integration helps to formulate inputs for open-format building performance simulation tools.

In typical workflows, BIM is used to generate a series of 2D plans from which a stand-alone energy model is built. Often, the energy modeler makes informed departures
from the original BIM geometry to address the specific
demands of energy modeling practice. As these models
develop independently, reconciling significant changes
in BIM geometry and program definitions with the
energy model is a painstaking, time-intensive, and
potentially error-prone endeavor, making it challenging
to practice iterative and collaborative design. The
authors investigated existing workflows for integrating
BIM and energy modeling information using gbXML
and IFC file types (Ali 2010, Garcia, et al. 2015,
Bazjanac 2008). However, these methodologies did not
meet one or more of the following objectives: (a) enable
all design team members to participate in the generation
and verification of model inputs; (b) leverage full
capacity of BIM content to minimize information lost in
transfer to the energy model; (c) incorporate changes to
building geometry into a revised energy model without
requiring extensive reconciliation; (d) customize
resolution and details of exported BIM content to meet
the needs of particular energy modeling objectives; and
(e) facilitate the visualization and interpretation of
energy model results in a BIM environment.

After reviewing the potential workflows already
available through other software, both firms decided to
pursue the integration of Revit and eQUEST through the
development of a bespoke Revit plugin, herein referred
to as the Custom Application. This particular direction
was chosen mainly because KieranTimberlake and
Atelier Ten each have expert users in these respective
programs and because a co-authored workflow would
mean that both parties could have full control over the
export from Revit. The workflow presented here allows
for a large project team to efficiently incorporate updates
to building geometry and program requirements into a
building energy model. Because the program definitions
in the BIM and energy models share the same underlying
structure, simulation results may be brought back into
the BIM environment to aid in interpretation and model
refinement.

METHODS

Information Transfer

Our workflow employs two modes of information
transfer between a building information model (Revit
2016 R2) and an energy model (eQUEST 3.65 DOE 2.2).
The first mode relies on the content of the Revit model
elements and parameters to derive an eQUEST input file
(INP) using our Custom Application. This methodology
presumes that features of each space that factor into
energy model definitions must be populated in the Revit
model before generating the input file. While this
workflow has the advantage of allowing designers to
participate in the definition and validation of such
parameters, it precludes the direct involvement of parties
who do not have access to the Revit model. Therefore, a
second mode of information transfer created for this
study involves centralizing information in a web-based
database (Microsoft Access 2013 / Sharepoint Online) so
that multiple parties can participate in defining quantities
relevant to the energy model, as well as to other project
features. This information is subsequently synchronized
with the Revit model using the Custom Application prior
to generating an eQUEST input file (Figure 2).

Figure 2 Information flow between tools in a typical
workflow (left) and the present workflow (right)

In this manner, all members of the design team can
participate in managing program and energy model
information. Because this methodology ensures that the
Revit and eQUEST models share the same naming
conventions, we anticipate that further development of
the plugin will recombine energy model results with
Revit geometry so that they may be visualized in a
spatially explicit manner.

The Custom Application is programmed in C-Sharp
using the Visual Studio 2013 development environment.
The user interface utilizes Windows Presentation
Foundation (WPF) components of .NET Framework 4.5.
Connection to the cloud-hosted Access database is made
through the OleDbConnection class of the .NET

---

1 KieranTimberlake has broad experience developing
bespoke building performance analysis tools, including
the Tally plug-in for Revit. Atelier Ten is a 25-year-old
firm of thought leaders and practitioners in building
performance consulting, with extensive energy
modeling experience.
System.Data Reference Assembly, and interaction with the Revit model is enabled through the Revit 2016 API.

**eQUEST and Revit Model Elements**

The primary motivation for this work lies in the seamless translation of model geometry from a Revit model into the geometry-related features of an eQUEST model. The hierarchy of geometry definitions in eQUEST was used to develop exportable features from Revit. This hierarchy includes: Polygons, defined by ordered sets of vertices in a 2D plane; Spaces, which contain the schedule and internal gain criteria necessary to represent thermal zones; and the construction surfaces of Floors, Ceilings, Walls, and Windows, containing material definitions and adjacency relationships to interior Spaces and to the exterior, which thereby form the boundary conditions necessary for eQUEST to define a complete set of coupled equations for its heat balance algorithms (Hirsch 2014, York, et al. 1981).

While Spaces derive their geometry from references to named Polygons, construction surfaces contain no such reference to their parent Space. These relationships are instead derived implicitly from their relative position in the text of the input file so that the order of Spaces and construction surfaces mirrors the hierarchy of eQUEST’s Component Tree. Furthermore, a wall that forms a boundary layer between Space A and Space B will appear only once in the input definition, where it will fall under Space A and include a reference to Space B in its “NEXT-TO” property. Thus, the first task of the parsing algorithm is to relate the geometric contents of the Revit model in such a way as to maintain an ordered set of adjacencies, so that proper references can be maintained and duplication of boundary surfaces avoided.

In the present study, Revit’s Area class rather than its Room class is used to define thermal zones. This has the advantage of allowing rooms with related functions to be combined, thereby reducing the complexity of the energy model. As will be shown later, the Custom Application retains the hierarchical relationship between Areas and Rooms so that at some later point Rooms may be used as the space-defining feature for the entire model or a portion thereof. Areas are created in the Revit model using a set of Revit Area Plans dedicated specifically to energy analysis. These views allow one to use Room boundaries as guides for drawing Area-bounding polygons (Figure 3).

Once Areas are placed inside these boundaries and their energy model parameters have been fully defined, they can be referenced in the Custom Application using the Revit API’s FilteredElementCollector class, along with their parameters and source geometry.

**Geometry and Relationship Parsing**

The workflow begins by partitioning Areas based on their Level parameter using a language-integrated query (LINQ). This ensures that each Area need only look to Areas on the same level to find adjacency relationships, thus greatly speeding up the algorithm. In instances where Areas are open to the floors above or below, a boolean parameter is assigned which instructs the Custom Application to identify those adjacency relationships as well. In principle, this approach may be modified to treat non-adiabatic floor assemblies.

Next, the Revit API’s GetBoundarySegments and GetCurve methods are used to extract the (x,y)-coordinates of the vertices and segments that define each Area’s closed boundary. A useful feature of Revit’s GetBoundarySegments method is that if a demising wall creates a condition in which one face of an Area is adjacent to two or more Areas, a separate segment will automatically be created for each adjacency. Therefore, it is certain that any point on the other side of a given boundary segment will always be contained in the same Area. The algorithm for determining adjacencies takes the midpoint of each segment, displaces it six inches (adjustable) from the boundary, which itself has zero thickness, and queries the remaining Areas’ closed boundaries to determine which, if any, contain the point. If no container is found, the point must represent an unenclosed exterior space, and the method represents that boundary through an Exterior Wall definition. If a container is found, then the surface is treated as an Interior Wall, and a reference to the containing Area is appended to that definition. Since each boundary

![Figure 3 Simplified energy analysis area plan detail, color-coded by activity (left), and with room geometry overlaid (right)](image)

2 N.B.: Revit and eQUEST class names are capitalized throughout
condition must appear only once in the eQUEST input file, a second operation is performed to test whether the parent space or the adjacent space appears first in the model definition, based on alphabetic sorting of the Area name. If the parent space appears first, then the Interior Wall definition is added as is with a “NEXT-TO” reference in the adjacent space. If the parent space appears after the adjacent space, then the wall will have been defined previously and must be omitted. For sake of legibility and error-checking, the definition is retained but prepended with the comment character “$”, which instructs eQUEST to ignore it.

While Floors, Walls, Ceilings and occupied Spaces can be defined directly through this methodology, Plenums and Windows require additional input. Plenums, as defined by eQUEST, have a specific relationship to the space they typically reside above insofar as they are coupled by a non-adiabatic surface. Because of this, they may share in the distribution of internal gains and in the routing of HVAC supply and return air. Thus, treating them as belonging to a separate level in Revit would be impractical and would break with eQUEST conventions. Instead, a boolean parameter is used for each area to indicate the presence of a Plenum and, if one exists, derive the Plenum definition using Revit’s built-in Ceiling Height parameter. Revit Windows, much like eQUEST Windows, are hosted in an Exterior Wall. However, since the algorithm relies on boundary segments to define Walls, there is no direct means of querying the model to identify specific Window elements for each Exterior Wall segment. Instead, a Window-to-Wall Ratio parameter is defined for each Area that contains at least one Exterior Wall. For each Exterior Wall with a non-zero Window-to-Wall Ratio, a single eQUEST Window is constructed as a ribbon window running the full length of the wall and scaled vertically to achieve the specified ratio.

Although Revit Rooms are not presently used in the derivation of eQUEST geometry, it is anticipated that their geometry and parameter content will eventually feed into certain model inputs, such as area-weighted internal gain criteria and thermal mass. Therefore, knowing a Room’s place in the model hierarchy has intrinsic value. Room relationships are determined in much the same manner as Area relationships. First, Rooms are partitioned based on parent Level using their Level parameter so that each Room need look only to Areas on the same Level for a potential containment relationship. Next, the centroid of the Room is calculated from its boundary vertices using the same methods as for Areas. Finally, each Area’s closed boundaries are tested to find which Room centroids it contains. The results are used to partition Rooms according to their parent Area.

Figure 4 Custom Application interface showing TreeView (left) and eQUEST input definition (right)

User Interface
The algorithms described above may be executed without the aid of any user interaction. However, the complex spatial and relational nature of this exercise makes quality assurance difficult without the aid of a user interface, particularly as new features undergo development. The interface created for this exercise organizes information hierarchically in a WPF TreeView that resembles the hierarchy of an eQUEST Component Tree, with Areas grouped based on their parent Level (Figure 4). Each Area node contains its constituent Room nodes, and each Area or Room node has a child node that displays all of its parameter values (Figure 5).

This allows one to verify inputs and identify new parameters that may contribute to energy model definitions, such as Window-to-Wall Ratios. Each Area and Level also contain a set of nodes that display the verbatim content of their constituent eQUEST input definitions, which also appear in their standard format in the right window (Figure 6).

Since the hierarchical nature of the eQUEST Component Tree is not readily apparent in the formatting of the input definition, this side-by-side interface has proven helpful in constructing and properly sequencing this text. For instance, while the eQUEST Floor definition logically falls prior to its constituent Area definitions in the TreeView, the eQUEST definitions are aggregated first by class, so that Area Polygon definitions precede Floor definitions (the latter being grouped together with Spaces, Floors, Walls, Windows and Ceilings). The TreeView hierarchy is used together with the eQUEST class definitions to sequence the output properly so that it may be imported as is into eQUEST without further modification. As the resulting eQUEST model demonstrates, the model’s geometry and naming conventions are retained (Figure 7).
Figure 5 Reference display of parameter values for Areas and Rooms, color-coded in red

Figure 6 Reference display of eQUEST Polygon, Space, and Boundary definitions, color-coded in blue

Figure 7 eQUEST output from a generated INP file
Energy Model Parameters

To develop complete eQUEST Space definitions, a considerable number of properties are defined in addition to the aforementioned Polygon references and boundary conditions. These include numerical properties such as occupancy, lighting power, and equipment power densities; named reference fields, such as occupancy, lighting, equipment, and infiltration schedules; and keyword fields that instruct eQUEST to set defaults based on a matching switch statement (Hirsch 2014). As this exercise began in Schematic Design, the initial development of the Custom Application relied heavily on keyword fields that are coded to instruct eQUEST to select standard ASHRAE default inputs. These inputs relate to common program types using a set of keywords such as “CLAS” for classrooms, “SRES” for student residences, and “PERF” for performing arts. This method allows for minimal input on KieranTimberlake’s part in early stages of design and leverages the intelligence built into the switch statements of Atelier Ten’s standard eQUEST workflow. However, as the project progresses and the energy model moves away from default values toward more nuanced definitions of Spaces, Zones, and Systems, more information may need to reside in Revit’s Area parameters to facilitate automated exporting. Both firms have identified certain features of the eQUEST input file that could benefit from automation. For instance, Zone definitions, which are not presently part of the automated export, contain an outside air flow per person property, which depends on both floor area and occupancy. Determination of this numerical value for each zone requires time-consuming calculations, which are normally done in Microsoft Excel. Since the floor area and occupancy type of each Area and Room are easily queried using the Revit API and are already displayed in the user interface, the Custom Application can be leveraged to expedite and quality assure such calculations (Figures 5-6).

As was discussed previously, the Custom Application automatically tracks relationships between Rooms and Areas even though Rooms do not presently feed into eQUEST input definitions. Future development will see two ways in which Rooms may factor in. The more direct way would be to add a boolean parameter to Areas that would allow the user to specify whether Areas or their constituent Rooms should be used as the Space-defining criteria. This would allow for flexibility in grouping certain Rooms together while leaving others itemized. In order to facilitate this feature, additional routines must be incorporated into the geometry-parsing algorithm to identify Room-Area adjacencies along with Room-Room and Area-Area adjacencies. The less direct way of incorporating Rooms into the eQUEST input definition would be to derive certain Area quantities from values contained in the constituent Rooms. For instance, a collection of Rooms with various lighting power densities could be aggregated into a single Area, and their effective lighting power density could be accounted for by taking an area-weighted average. This practice seems well-suited to numerical fields, but would not readily accommodate reference fields such as schedules, which cannot be averaged in a straightforward manner.

Access Synchronization

A potential drawback of predefining eQUEST model parameters in a Revit model is that it could potentially lead to information bottlenecks in which KieranTimberlake, which does not possess Atelier Ten’s energy modeling expertise, must nonetheless populate energy model inputs in the Revit model prior to furnishing Atelier Ten with an eQUEST model. To address this potential shortcoming, the teams have collectively authored a web-based program database which can be accessed remotely by either firm. This database has been developed with energy modeling criteria in mind so that inputs can be defined to align with energy model nomenclature and may be specified either at the category level, such as “Performing Arts,” or at the space type level, such as a “20-person Classroom.” Relationships between the Access database and Revit model are maintained through a Space Type ID parameter that matches the corresponding Row ID of the Access Space Type table. The Custom Application queries this table using an OleDbConnection class and highlights space types that are not synchronized, or space types for which there is a discrepancy in the room count or parameter values between Access and Revit (Figure 8).

By selecting an item in the Custom Application’s interface, one can inspect discrepancies and synchronize

![Figure 8 Interface for synchronization between an Access database and Revit model](image)
with Access before exporting to eQUEST. In this manner, Atelier Ten is empowered to update energy model inputs upstream of the Revit model, and these changes will become readily apparent to KieranTimberlake as it performs itemized synchronization prior to generating a new energy model with updated building geometry.

RESULTS AND DISCUSSION
Efficiency Gains and Quality Assurance
The efficiency gains set as targets for the design team include savings in data entry time and model iteration time, and agility in answering specific design questions. One benefit of the designed software workflow is that model geometry and space types automatically update themselves within Revit, reducing the amount of time that needed to bring the energy model up to the current design. In a recent test, the Custom Application took under 30 seconds to parse a Revit model containing 201 Areas and generate a new INP file containing over 20,000 lines of code.

In contrast to typical energy modeling practice in which most inputs reside solely within the energy model, the workflows presented here allow multiple parties to participate the creation and review of energy model inputs. This holds intrinsic value from a quality assurance standpoint and reduces dependence on end-of-phase reports to coordinate and review energy model assumptions.

Furthermore, placing energy model content in a Revit model provides a means of reviewing analysis results in a spatially explicit context. For example, Area or Room parameters may be used to generate a series of floor plans that are color-coded to represent any desired energy model inputs, such as lighting power density or activity code (Figure 3).

Current Development
Several aspects of the present workflow have been designed to anticipate ongoing feature development. As mentioned previously, tracking Room-Area relationships will allow for increased spatial resolution of the energy model as the Revit model content continues to develop. The design of the side-by-side interface, which decouples the natural model hierarchy from the particular sequence and syntax of the eQUEST input definition, anticipates extension of this work to other energy model exchange formats such as the input data file (IDF) format for EnergyPlus. Additionally, the consistent nomenclature assured by this workflow provides a natural framework for incorporating energy model results back into the Revit model to aid in spatially explicit inspection by the design team. Efforts are presently underway in developing a web-based viewer that will combine BIM and energy model content to permit a more interactive visualization of analysis results. This tool will allow loads and system responses to be viewed across space and time domains, thereby highlighting opportunities for passive design and load-sharing strategies. It is anticipated that an exercise in visualizing energy performance will elevate the design team’s collective understanding and solidify the development of this project as a rich ecosystem of relationships.

CONCLUSION
While still under development, the interactive modeling workflows presented here demonstrate value in a seamless integration of building geometry and collectively authored energy model parameters in service of a more efficient iterative energy modeling practice. By placing architects in closer dialog with the inputs and results of energy models, this work has the potential to elevate energy model literacy among non-experts.

ACKNOWLEDGEMENTS
The authors would like to thank the following individuals from KieranTimberlake and Atelier Ten who helped make this work possible: Stephen Kieran, James Timberlake, Billie Faircloth, Matthew Krissel, David Riz, Kit Elsworth, Zinat Yusufzai, Uk Jung, Michael Tillou, Jagan Pillai, Leanora Paniccia and Paul Stoller.

REFERENCES
York, D.A. and Cappiello, C.C., 1981. DOE-2 Engineers Manual (Version 2. 1A) (No. LBL-11353; LA-8520-M). Lawrence Berkeley Lab., CA (USA); Los Alamos National Lab., NM (USA)