Leveraging BIM Downstream: Real-World Examples of Using BIM Data Throughout the Project

Jeffrey McGrew - Because We Can, LLC

BD33-2  With the addition of IFC exporting and the API within Revit 8, as well as the new efforts and tools for interacting with Revit and its data, we are starting to see real BIM sharing taking place. This session will cover both real-world and potential uses of Revit's information, starting with the basic exporting and manipulation of schedules and DWFs, moving onto the intermediate ODBC exporting and gbXML, and then onto the advanced API issues. We'll also look at several of the tools available today for interacting with Revit data for automated spec writing, report generation, gbXML, and more.

About the Speaker:
Jeffrey is the main organizer of the Bay Area Revit User's Group, one of the first Revit user groups in the world. He is a senior modeler at Tectonic Network, a company devoted to creating content and tools for Revit that help make the whole picture of BIM more accessible to all. Prior to this, Jeffrey worked for various architecture firms as both a designer and Revit expert on large and complex projects. He also worked for a Reseller providing training and implementation for Revit. As an early adopter of Revit, starting with version 3.1, Jeffrey has also helped many of those new to Revit, both locally and via AUGI. Currently undergoing his ARE licensing exams, Jeffrey hopes to be fully licensed as an architect within the year.

jeffrey@becausewecan.org
Simple BIM sharing and usage

(the past, or stuff we've been able to do for a while)

Schedules

Schedules are one of the best things about BIM, and is where Revit really shines. We'll quickly review how one makes a schedules in Revit, and then focus on how they can be extended with custom parameters and calculated values, and then show some real-world examples of their use. Schedules are just another view within Revit, and are fully bi-directional, just like all other views within Revit.

When we generate them, first we pick what information we want somewhere within the Schedule, then what information we want filtered out, then how we want what's left sorted, formatted, and then finally how we want it to appear when on the sheet.

First off, we'll talk about adding more parameters. In Revit, you can add at will a new parameter of any type. Parameters can track any information about your Project, and can be used in calculations as well. They are the way that you 'teach' your Project about important information you want to keep track of, and derive things from. The default ones that are built into Revit are a good start, but they don't cover all cases, and we can add a lot of value to our Project just by extending the parameters to track whatever information is important to us. We can make two types of parameters, Project Parameters and Shared Parameters.

Project parameters live just within your Project, and can typically only be applied to a single category. For example, if we add a 'wattage consumption per hour' number parameter to the light fixtures for energy calculations, and we use a project parameter, we'll be able to track that information for all the light fixtures in the project now.

We can then total it up, per room and/or area, and have an idea about how much power it's using.

However, our other projects won't now 'know' about wattage consumption per hour. Also, we can't have that value show up in a tag. But for non-graphic project information, they work fine, and a big advantage of them is that they are easy and quick to make. Now that we've got a total of how much energy our rooms are consuming, we can export that out to Excel (or any other Spreadsheet software), and either do more complex calculations or pass it to the energy consultant to aid their work.
Leveraging BIM Downstream: Real-World Examples of Using BIM Data Throughout the Project

However, let’s say that we want this value to show in the light fixture tag, or we want it to be applicable to rooms or larger areas so we can total up consumption per room, or, or we just want it consistent between all our projects. For that we need a Shared Parameter. These are just parameters that are pulled off a master list, so that you can use them within multiple files, Families, Category’s, and the like. The parameters themselves are ‘Shared’ between Families and Projects, not the information within them. We’ll see an example of them in a moment.

Now let’s move onto calculated values in schedules, another great way to apply BIM. Calculated Values are pretty easy. In our example, we’re going to automatically calculate the number of people in a room from an occupancy load factor and the area of the room. As long as we spell (and capitalize!) the parameter names exactly, and keep consistent units, we can have any number of calculated values. However, sadly, we can’t have the result of a calculated value used within another calculated value yet (but we can do that sort of thing within Families). First, we’ll generate a room schedule, that lists the room number, name, area, and occupancy, and then add a number parameter called ‘Occupant Load Factor’.
Then we'll create a new calculated value. We'll call it 'Calculated Number of Occupants'. We'll have to set it to 'Area' to keep consistent units (or use a more complex formula that converts an area to a number), and we'll use the simple formula 'Occupant Load Factor * Area'.

We'll also need to adjust the formatting for the calculated value so that it doesn't show the default 'SF' tag areas get, and we'll want to set it to total that column up for us. This then gives us a room schedule that when a room's Occupant Load Factor is entered in, we get a calculated number of occupants within that room.

<table>
<thead>
<tr>
<th>Number</th>
<th>Name</th>
<th>Area</th>
<th>Occupant Load Factor</th>
<th>Calculated Number of Occupants</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Waiting Area</td>
<td>191</td>
<td>15</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>Kitchen</td>
<td>163</td>
<td>15</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>Small Conference</td>
<td>207</td>
<td>20</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>Large Conference</td>
<td>434</td>
<td>15</td>
<td>29</td>
</tr>
<tr>
<td>5</td>
<td>Office</td>
<td>114</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>Connector</td>
<td>918</td>
<td>100</td>
<td>9</td>
</tr>
<tr>
<td>10</td>
<td>Accounting Office</td>
<td>766</td>
<td>10</td>
<td>16</td>
</tr>
<tr>
<td>13</td>
<td>Interior Room</td>
<td>155</td>
<td>15</td>
<td>3</td>
</tr>
<tr>
<td>20</td>
<td>Stairwell</td>
<td>142</td>
<td>50</td>
<td>3</td>
</tr>
<tr>
<td>21</td>
<td>Small Meeting Room</td>
<td>210</td>
<td>20</td>
<td>2</td>
</tr>
<tr>
<td>22</td>
<td>Small Meeting Room</td>
<td>110</td>
<td>20</td>
<td>2</td>
</tr>
<tr>
<td>23</td>
<td>Small Meeting Room</td>
<td>107</td>
<td>20</td>
<td>2</td>
</tr>
<tr>
<td>24</td>
<td>Small Meeting Room</td>
<td>117</td>
<td>20</td>
<td>2</td>
</tr>
<tr>
<td>25</td>
<td>Small Meeting Room</td>
<td>111</td>
<td>20</td>
<td>2</td>
</tr>
<tr>
<td>26</td>
<td>Stairwell</td>
<td>137</td>
<td>100</td>
<td>3</td>
</tr>
<tr>
<td>27</td>
<td>Stairwell</td>
<td>122</td>
<td>100</td>
<td>2</td>
</tr>
<tr>
<td>31</td>
<td>Private Office</td>
<td>185</td>
<td>20</td>
<td>4</td>
</tr>
<tr>
<td>32</td>
<td>Private Office</td>
<td>244</td>
<td>20</td>
<td>4</td>
</tr>
<tr>
<td>33</td>
<td>Private Office</td>
<td>161</td>
<td>20</td>
<td>4</td>
</tr>
<tr>
<td>34</td>
<td>Meeting Room</td>
<td>176</td>
<td>20</td>
<td>3</td>
</tr>
<tr>
<td>35</td>
<td>Private Office</td>
<td>216</td>
<td>20</td>
<td>10</td>
</tr>
<tr>
<td>36</td>
<td>Private Office</td>
<td>275</td>
<td>20</td>
<td>13</td>
</tr>
<tr>
<td>37</td>
<td>Private Office</td>
<td>435</td>
<td>20</td>
<td>22</td>
</tr>
<tr>
<td>43</td>
<td>Private Office</td>
<td>272</td>
<td>20</td>
<td>9</td>
</tr>
<tr>
<td>44</td>
<td>Office Pool</td>
<td>3/36</td>
<td>20</td>
<td>1</td>
</tr>
</tbody>
</table>

These kinds of parameters and formulas can live inside of Families as well. Here's a real-world example from a company that does a lot of large multi-family housing and condo projects. Code requires on such project that every apartment/condo unit has a certain amount of natural daylight and ventilation. By making Shared Parameters for those items, adding them to the Window, Glass Door, and Ventilation families that Project used, and having their value generated by a formula within the Families themselves we were then able to generate a Schedule that totaled up how much natural light and ventilation each room in our project was getting, and quickly know if we were within the Code or not.
This company also exported a lot of their schedules to Excel to give to the developer, for they were also the client & contractor typically on most jobs. Those schedules helped the developers out a lot, for they didn't have to re-count things on paper, and it gave this company a real edge and the developers a lot of incentive to rehire them on future jobs over other firms not using BIM.

Another good example of what could be done with just schedules in regards to BOMA calcs. If you create your Area Plans in Revit, it's already smart enough to try to follow BOMA standards when defining Areas. Once you get your Area Schedules, you then export to them to Excel, where more values can be calculated automatically for all the various rentable vs. non-rentable ratios and percentages.

**gbXML**

gbXML, in conjunction with the Green Building Studio, is an interesting application of BIM that many people are leveraging. In a nutshell, your Project gets exported to a industry-standard format, the gbXML format, that anyone could write tools to parse and generate reports from. Green Building Studios has a suite of such tools, and they give them away for free. The workflow here is that you export out your Project to gbXML, upload it to them, tell them more about the Project itself, and then a while later you get a report.
These reports can be really useful to those that understand them, and for doing LEEDS efforts. You can run multiple reports, too, so early on in the Project you can make several versions of the Project, send them in, and then compare the reports to find which orientation or glazing gives the best results. There are many real world Revit-using firms that are relying on gbXML to give them some design feedback on their Projects.
Note that you have to have no undefined rooms and/or areas, and you have to have an project address with a zip code in the project information in order for it to work.

**Design Visualization & Lighting Analysis**

Revit's BIM models are very detailed, and as we've seen we can extend that information pretty easily. However, the rendering tool included in Revit, Accurender, while great for small stuff and design study, it's not on par with dedicated visualization tools like 3D Studio, and it doesn't contain any lighting analysis abilities like Radiance/Ecotect does. By exporting our BIM model, however, we can intelligently leverage BIM using these other packages.

Revit can produce DWG, DXF, and SAT files consisting of Solids or Polyface meshes. Revit also follows the import/export filters when exporting your 3d model, so you can set up sub-categories and a custom export filter to get a well-organized 3D model. In Revit, what you see is what you get Exported, so we're going to go to a 3D view, turn everything on, and export that to a DWG and a DXF.

Now we can open our 3D model in other applications. Here it is in 3D Studio:
Where we could then produce high-level renderings and animations, or use a 3rd party rendering engine, such as Maxwell or Radiance, to do complex lighting simulations.

We can also bring those models into complex analysis & design packages. Here’s our BIM model in Ecotect (I turned the furniture off, for we wouldn’t have it part of our analysis:

Right now, at least with Revit, the tools for this kind of BIM leveraging are pretty bare. Only 3D Studio can read Revit materials, so unless you use an add-on for 3D Studio that does real lighting analysis you’d have to redefine your lights and materials within the model. Again, many Revit-using firms out there are leveraging this within their firm, or sending these models out to consultants to aid in their work.

**Revit & DWF**

Revit can easily export to both 2D and 3D DWFS, as well as read red lines made to DWF sets. An added advantage or doing this with Revit is that all the view tags remain links within the DWF, allowing for people to navigate around your set. This DWF exporting allows for two things, a easy and fast way for someone to review and red line the drawing set, and very lightweight 3D models that are small enough to e-mail yet retain colors, information, and more.

Additionally, with the DWF API being open, there are interesting opportunities in the future of tools that could run calculations or analysis on submitted DWF’s, for the 3D DWF contains most of the project information in a form small enough to e-mail.
Intermediate BIM sharing and usage

(the present)

Exporting to ODBC

Revit allows you to export your BIM Project via ODBC to any number of compatible databases. What this means is that our whole Building Model, and all its information, can become a real database outside of Revit, and have any number of interesting applications done with it.

Our first example is just exporting via ODBC to Excel. As with any ODBC, we've first got to give it a source that we're going to send our Data too. In this case, we'll generate a basic, blank Excel spreadsheet. It doesn't have to be blank, mind you, Revit will just add new tables (sheets) to the end of the spreadsheet if it's already got data in it. However, you can only export via ODBC to Excel once per spreadsheet, for Excel doesn't have the ability to track changes via ODBC (more on this later). We then go to File, Export, and choose ODBC. In our case it's a 'Machine Data Source' export, and then we browse to our blank file and then hit OK.

We then go and open up the file and see that Revit has filled it out with data from the Project, two sheets per Revit category, one for the Types, and one for the Instances. Each one has a ID number that's unique to it within the Project (but not across projects) and all the default and shared parameters from the Project.
This is enough information to do a quick-and-dirty cost estimate. By just adding or creating some of our own sheets that total up the number and/or linear feet of elements within our model, and applying a simple unit cost to it, we can wind up with a rough estimate for our Project.

However, if we export via ODBC to more complex databases, and do more complex things with the data, we can arrive at more complex numbers. For example, if you exported to a SQL database, it would be possible to have another tool parse the data, calculate more complex costs, and then generate a cost report as a auto-updated web page. Another advantage of exporting to a 'real' database is that you would be able to export to the same database multiple times, and ODBC is smart enough to change and delete 'old' entries based upon the most current data.

A great real-world application of this is being used by a major home builder in the Midwest. Their designers put together the house based upon the client's wishes from a kit of common parts, and then export via ODBC to a standard in-house database. The project manager can then go to an online web form, and price certain options within the house, such as the quality of the trim, the manpower available, and the current cost of lumber in the client's area, and get a informed report of the cost and time frame it will take to deliver the house within that area. Via ODBC they are able to have the real data from the building design participate with other databases of information, becoming a key player in the planning of the project.

Also, there are tools available that make use of this ODBC exporting to leverage the BIM Project in interesting ways. One great example is E-Specs, made by InterSpec (http://www.e-specs.com/). This tool uses a custom ODBC driver to collect and then harvest the Revit data, which it then generates a spec from. It looks at all the CSI codes within the Revit model, and those in turn trigger what spec sections get included in the spec. You then edit and modify the generated spec further using their special editing toolset.

**Collision Detection**

One interesting and quick example of leveraging the BIM model is in using the collision detection within Revit for coordination and other things. We can test if there are conflicts within the model, between models, and (if imported correctly) solid models from ADT/ABS. Here, in this first example, we've imported in a mechanical model from ABS, and placed it into a in-place Mechanical Family in Revit. As long as we exported that model as a solid out of ABS, Revit will be able to have it participate in the collision detection.

We can also use the collision detection to design in a little bit of building code awareness into our Project. In this example, two Revit families, a door and a sink, have invisible solids that represent their required clear area. When we run the check, if they are too close to each other (or anything else that encroaches on that clear area), they will wind up colliding and we'll see it in the report.

**Revit as a 'platform'**

Another great application of the BIM model comes from Autodesk itself,
and that’s the ability for Revit Building, Revit Structure, and 3rd party analysis tools to work together to deliver a coordinated, intelligent BIM model.

Revit Building and Revit Structure can ‘talk’ to each other, in that you can link a Building model from one into the other, and do so in an intelligent way. You can also schedule items within the links, perform collision detection between the models, and even track elements to know when/if they have changed.

You can also use Revit Structure with 3rd party analysis software that will review the Building Model and make intelligent changes based upon that review. I won’t go into much detail here, for there are whole classes here at AU on this topic alone.

**Revit & IFC**

Another exciting application of the BIM information is with IFC. IFC is an open, industry standard for BIM information for sharing between BIM applications and other packages. Currently in it’s infancy, it’s hoped that IFC will become the mainstream way of moving this rich and intelligent BIM data between products.

Revit can export to IFC, and with careful application of the included IFC Shared Parameter file, we can arrive at a properly documented IFC model. However, it’s still rather limited as to what can be done with such a model at this time. A good metaphor, from someone on the IAI committee that’s responsible for the IFC specification, is that it’s a lot like the phone system, in that due to it’s standards one can pick up the phone and dial someone in Finland without any issue, but when the call goes through, they will be speaking Finnish while you’re speaking English. Right now we have the applications talking to each other, but not really understanding each other. That’s a future step. For example, if you export out an IFC model from Revit or ADT, you can import some of it into ArchiCAD, and some of it will even retain some data and intelligence. But it’s only certain parts, and not the whole building, and it only kind of works at this time. What’s a lot more exciting is non-BIM systems use of IFC.

One very interesting application of IFC is the Singapore Building and Construction Authority’s automatic plan checking system, CORENET e-Plan Check system. By submitting an IFC file of your Project, automated checks for simple building code compliance can be performed. Revit recently became certified for generating such IFC files, so we should begin to hear more exciting things about this in the near future.

Some major clients, such as the GSA, are also looking at IFC as a future required deliverable at the end of the project, for them to use later for facilities management and record keeping.

**Advanced BIM sharing and usage**

*(the future)*

**Revit API**

Revit now has an API. It’s a doosy. It’s all .Net based. What this means is that it’s more for allowing Revit and other programs to talk to each other, than it is about extending Revit internally or scripting Revit itself (two things not really possible currently). It’s more of a ‘bridge’ than it’s something like what LISP is within AutoCAD.

It’s how the Structural package ‘talks’ to the analysis packages, for example. However, seeing that the API gives access to the whole Building model, some very interesting things are possible in the near future. Many of the things done with ODBC and DWG exporting could possibly be not only done automatically, but could go both ways, allowing the spec tool to change something in the Project, or the lighting analysis package to change the light fixtures. Exciting (but scary!) stuff.

If you want to get started with the API, probably the best way to do so is to join the Autodesk Developer Network. They have multiple code examples to get you started.
If you don’t want to join or don’t have the money, you can actually get started pretty easily, it will just take a lot longer to get up to speed (unless you already know a lot about .Net software). You’ll need two things: A .Net compatible language and/or IDE, and the Revit API help file. The first is actually freely available from Microsoft (they have a free version of VB.NET and C#.NET that you can download), and the second comes with Revit, it’s the file called ’RevitAPI.chm’ in the Revit Program folder.

Once you’ve written your code, you compile it to a .DLL, place that .DLL into the Revit Program folder, and then add an entry to your Revit.ini file pointing to it. You’ll then have it available under ’External Tools’.

There are several people actively working on some very interesting things using the API that participate within AUGI. IF you want to learn more about the Revit API, I’d highly recommend that you go there.

**Revit-to-fabrication**

Here’s an area I’ve been doing a lot of research in lately. Seeing that Revit produces solid models, it’s possible to export them to CADCAM software for CNC-toolpath generating. While this could be used to just generate physical models, it can be carried much farther, and be used to generate actual parts of the project.

By creating custom Revit families, and then using them within our Project, we can arrive at a design for a custom element very quickly. Here we’re going to use our reception desk as an example, we’d like for it to have a complex curve to it’s surface.
Leveraging BIM Downstream: Real-World Examples of Using BIM Data Throughout the Project

Then we can take those elements, and export them one at a time to separate solid files. We can then bring them into tool-pathing and/or CAD/CAM software to create the job files required to produce those elements directly. Here we've isolated that front of the desk, and exported it to a DWG file, and then into our toolpathing software.

This would be if we wanted to have this piece carved from a solid block of material. We also could have broken this down into 'ribs' onto which a skin would be applied in field. Once we've gotten to this point, we can generate any number of custom elements to work together as parts of systems within a presentation model, within our actual Building, or just use it for custom elements within our Project.

For example, there's an Architect who's been doing this same process but with Rhino for a few years now. He generates the house completely in Rhino, breaks it down into a kit of parts, has a CNC mill carve out molds for those parts, casts the parts in concrete, and then builds the house like a giant puzzle.
This is exciting, for it makes completely custom and complex construction available for the same cost as typical or prefabricated construction, and opens exciting new doors.

**Conclusion**

So, as we can see, there are many things we can do just within Revit to leverage BIM in interesting and productive ways, as well as many things that can be done today to make better use of that rich BIM data. I hope that this class has given you a lot of ideas, and I'd love to see some of your results and examples. Please send them to jeffrey@becausewecan.org, if you've got the time.

Thanks!