Autodesk University 2012

Triage for Tour Autodesk Revit Family

Speaker(s) Name – Company Name
(Assistant/Co-presenter optional) [Arial 10]

AB2623 We have all built family content over the years. Some of us are proud of what we have built and have every right to be. Some of us, on the other hand, have built content we wish we could take back and never show to anyone. This class will answer any of your family questions and will strive to not only answer the question but provide a background to the answer. No "yes" or "no" answers here. Expect dissertations for an answer. Expect to learn. Be warned—this class will not cover the basics of family creation. You are expected to know how to build a family when you walk in the door.

Learning Objectives

At the end of this class, you will be able to:

- Describe best practices for family creation
- Take full advantage of repeatable troubleshooting techniques used by advanced users
- Execute rapid repair techniques for multiple families
- Generate consistent quality families without the issues that are common to new content creators

About the Speaker

Shawn Zirbes is the Chief Technical Officer for CAD Technology Center, Inc., in Minnesota. With twelve years' experience using all Autodesk® AEC software, he is an expert in planning and executing software implementations for the AEC industry. He has worked with a wide variety of clients ranging from commercial, residential, health care, and industrial design firms, their consultants, and the building product manufacturers supplying the building materials for these projects. Shawn's wide range of Building Information Modeling experience is being leveraged by Autodesk as a special Revit® consultant for Autodesk.
support@cadtechnologycenter.com

The Essential Best Practices

Though this is not an essentials session, we need to ensure that we are all on a level playing field. The theories here can certainly be debated, but the results attained by following the essential best practices will speak for themselves. Yes, it is possible to attain good quality families by breaking these rules, but the rules will help to ensure consistent functionality between all families in your content library.

Establish Host and Family Category Early

The host and family category define the core abilities of a family. If defined correctly then the family can function as expected and desired. If either are out of sync, then the family will have issues at some point in its life cycle.

Host

If the host is correct, then the content will function as needed when used in a project.

If hosts are incorrect then insertion may be impossible or at the least difficult.

Category

If the category is correct then, like having a correct host, the content will function as needed in a project.

If the category is incorrect then visibility graphics, scheduling and other aspects will not function as needed.

Name All Reference Planes

Many individuals who hear this ‘standard’ disagree with it. However, any family encountered months later where the reference planes are not named is quite difficult to modify. Sometimes it’s impossible to modify or repair because the skeletal structure cannot be understood. Named reference planes are like coordinates on a map, or the bones in our bodies. Imagine if someone said, “Go over there”. They don’t point of indicate where over there is, but you are expected to just know where ‘there’ is. Or perhaps a doctor says that you have a broken bone in your leg. It’s perhaps broken by your thigh but with no way to identify it the cast may get placed on your ankle. The fix may be correct, but the fix may be in the wrong place.

Identify what the plane is for. Give it a name that is human readable.

Set Reference Type for All Reference Planes

The type of reference is even more important than the name of the reference plane. If the type of reference is correctly set, then family nesting and swapping can function as expected. Aligning and locking families to other objects can be properly maintained. Having the correct type of reference can affect everything about the family right down to the ability to have certain aspects of the family dimensioned in a model.

Only Dimension to Reference Planes

Now this best practice does have some flexibility, but whenever possible it is best to dimension to a reference plane and have the graphics link to the reference planes. Yes, it takes longer to draw, set the type of reference and name the reference planes. However, if an issue arises and all the dimensions are buried in sketches, then it is almost impossible to determine where the issue is and how it can be corrected. I've noticed that families constructed with few reference planes tend to have the greatest quantity of flexibility issues. They also tend to be the most difficult to troubleshoot.

Do Not Nest Deeper Than 3 Levels

Nesting is a good thing. It is an excellent mechanism for leveraging swappable components or reusable components. However, nesting does have issues. The deeper content is nested the slower the family will function and the larger the final family will be.

Use Type Instead of Instance Whenever Possible

Instance parameters increase the final project's file size more than type parameters. Too often we see that a family author will choose instance for every parameter in place of understanding what parameters need to be instance vs. type. Having the correct balance will ensure the best possible functionality of a family and the best balance of file size to user convenience.

Use Shared Parameters Whenever Possible

This best practice is simply to ensure consistency. However it does have far reaching benefits elsewhere. This can affect the generation of future content. This can affect the creation of parameters for the current family. If a parameter has to be manually thought about it may cause a person to think a little harder about what type of parameter they are generating. In place of simple quickly creating a 'Voltage' parameter as a text field, perhaps we can promote the creation of a true voltage value.

Granted, the shared parameter system is cumbersome, but it is what we have. Using what we have to best promote good quality families consistently is important. Hopefully when Autodesk improves upon the shared parameter system the current system will easily migrate and if we follow a consistent standard. Of course if we follow a standard now, then we will reap the benefits now. If the system changes, then it changes, but not following any standard will certainly cause issues now, and certainly if the standard changes we are will going to have issues as we have no standard to attempt to migrate.

Flex Everything Often

Following the entire above standard is good, but something can still go wrong. We are after all human and we make mistakes. So continuous testing of family development is imperative to ensuring that it works as desired and expected.

Rapid Repair/Testing Techniques

Having quick points to check in a family is meaningful for testing when something goes wrong. Likewise having good tools for maintaining a library of repairing a single family will be vital. This will not resolve every problem, but it will help to resolve a few of the common issues quickly.

Using Family Processor

Family processor is a tool that can assist in standardizing family content in a library. It can change parameters and parameter functionality. It can manipulate materials to assist in standardizing naming conventions and settings. This is an invaluable tool in any content managers arsenal.

Changing Parameter Setups

An incorrectly generated parameter can be the cause of many issues. Mentioned earlier was a voltage parameter configured as a text value. This will not serve the needs of others as families are developed. It is important to check into how a parameter will be used by other disciplines as it is developed. Initially it will take longer to generate the single family, but time will be saved in the future as that value is able to be reused and as the data stored within the parameter is actually containing meaningful data that can be used elsewhere in design.

Flexing Quickly

When flexing family content it can be cumbersome to change all flexible values to ensure that components flex as expected. Instead, generate a “Flex” type within the family that has preset values differing from the base type. This will ease the flexing process making it faster to execute flex tests.

It is common to forget to flex the host geometry. This can be as important as flexing the family. Many times we attempt to insert content realizing that in a project is does not work as it did in the family creation environment. This can usually be avoided by generating a new type for the family host as well. This makes flexing the host simple and efficient.

Testing Live

Of course the proof is in the actual functionality in a project. Never release a family without fully testing it in the project environment. Testing doesn’t only mean inserting. It has to be run through the paces in its actual use environment.

Architecture Testing

Doors and windows must be fully flexed in all types of walls. Railing components must be tested in railings on stairs that actually flex floor to floor height. Simply testing for the single initial instance will rarely serve the future of design changes. Unless your architects never make design changes??

MEP Testing
Mechanical equipment must be tested with Pipes and Duct attached. Electrical equipment and devices must be circuited to ensure that electricity flows as expected. Objects should be inserted on a linked Architectural model, not live geometry.

Repeatable Trouble Shooting Techniques
Having a few known tricks in your arsenal to test why something may not function as expected will greatly speed up some of the testing and repair process.

Why Will The Object Not Insert?
One of the most frustrating issues we can encounter is content that will not insert into the project for testing in the first place.

Host is not correct
Quite often the original family may have been generated with an incorrect host. This is not easily correct in most cases, but identifying it as the issue early will allow for more rapid correction and perhaps stop continued incorrect hosting.

When is the host incorrect? When the object does not have to cut the host, yet is still hosted, it is probably incorrect. A door should be hosted to a wall as it needs to cut an opening. A piece of casework likely doesn’t cut a hole on the wall and therefore shouldn’t be wall or floor hosted. A lighting fixture may cut a hole in a ceiling, but if it is used by an electrical engineer, then they won’t have ceilings. The electrical engineer may have an architectural linked model, but a ceiling hosted lighting fixture regardless of how well built will be useless to them.

Host is correct
If the host is correct, but the family will still not insert it is likely that there are constraints within the family that are being made impossible by the current host’s configuration. This can sometimes be tested by using larger or smaller hosts.

Why Will The Object Not Schedule?
Not scheduling or improper scheduling is a fairly common issue and can often have quite simple resolutions.

Wrong category
If a family is in an incorrect category it will not schedule as expected. It may schedule, but will not appear in the desired schedule. Opening the family to check if its category is correct will inform the fastest if the family is properly categorized.

Wrong parameter setup
If a family parameter is not shared it will not schedule or will not schedule the same as

Nested Family Will Not Swap?
There are 2 major reasons for having issues when swapping nested families.

Align/Lock is not maintained
This issue can generally be attributed to improper Reference plane configurations. There is one primary culprit. The Reference planes used for attaching a nested family to a specific location must have consistent 'Type of Reference' set between nested families. All too often we notice that the nested families have the aligned/locked reference planes set to 'Weak Reference', the default reference type. The reference planes that define the origin point of the nested family should be set to one of the 9 primary reference types. 'Strong Reference', 'Weak Reference' and 'Not a Reference' will not allow the family to swap as expected. Generally resolving this setup will make the families begin to swap correctly.

Parameter links are lost
When the nested family is swapped warnings are displayed indicating that the parameter links are not able to be maintained.

Why Will Data Not Flow?
When working with MEP content the connectors are the heart and soul of the family content. It could be a simple box, but if it has correctly configured connectors, then the data can flow and value is added. If the content looks great, but no data flows, then the content is almost useless.

Connector configuration
Connectors must have the internal values linked to appropriately configured parameters. Generally values receiving flow that must be communicated to another connector should be configured as an instance parameter. Generally this also shouldn't be the designed value, but rather the actual value read form the system. Internal or schedule math should be used to determine if the design value and the actual values align. Don't tell the system what it should be, ask the system what it is and respond accordingly. Allow Revit to be used as a design and validation tool. Allow it to provide the 'Information' it is supposed to.

Random reason
Sometimes the correct configuration of a connector can be difficult to derive. Sometimes Revit doesn't work as we expect it to. It works; we just have to figure out what the heck it needs to actually work.

Closing Thoughts
Leveraging proper standards and documenting company processes will assist in generating content that doesn't require triage. We all do have content that does require said triage. It may have been our fault, or the fault of someone else in our company. Of course, all that content that originates outside our company control will need some medical assistance, and having a properly documented standard, and a process for handling that content will make it far easier to repair and use the content that at one time didn't work.

Consistency Doesn't Equal Quality

The above stated, it is important that the standards follow good industry standards. One of my favorite statements is, "Of course I believe in standards, that's why I use mine wherever I go." Using your own standards isn't a standard. A standard that conflicts with what the rest of the industry is doing is disruptive. It costs time and credibility. Job security? Sure, perhaps initially, but when the rest of the team see other firms succeeding and progressing much faster, will that not speak volumes to the leadership in places where progress comes much more slowly?

The field of Architecture and Engineering has been using Revit for a while now. Over 10 years. The standards have been developed and tested. There is very little that can't be achieved in Revit. There are few things that haven't been tried and in some way standardized. Sometimes invention and innovation is good, but when we spin our wheels in areas where standards have been developed all we do is waste time and slow the implementation of Revit and BIM in our firms. Where standards are vague, spend your time. Where the standards are well known, I recommend trying to follow those standards.