We share work all the time, but with Autodesk® Revit® software we also have to learn how the software enables us to interact and integrate the work we all do in a new and different way. Revit introduces new language and rules. Most of all, it makes us think differently about what we mean when we define what is your stuff and my stuff. This class will tackle the language and rules. It will give you a better grasp of what is involved as well as give you greater confidence.

Learning Objectives

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

- Explain fundamental worksharing concepts
- Communicate and manage the structure of worksets better for a project and the team
- Deal with conflicts or confusion with more confidence
- Use the features of Revit to collaborate more effectively

About the Speaker

Steve has been presenting at Autodesk University since 2004 as well as RTC since 2006. He started using Revit seriously in 2002. He is an established Revit consultant. He has been very active in the Revit community at large. He created, writes and manages three Revit blogs: www.Revitoped.com - www.revitinside.com - www.revitjobs.com. He currently lives in Southern California with his wife and two children. You can contact him via email: steve@aecadvantage.com
# Table of Contents

- Learning Objectives .......................................................... 1
- About the Speaker ............................................................... 1

## Basics ............................................................................. 7

- Why Worksets? ................................................................. 7
- The Library Metaphor .......................................................... 7
- How Many worksets? ........................................................... 8

## Creating the Central File ..................................................... 10

- Step One - Good Location/Name ........................................ 11
- Step Two - Enable Worksets ................................................. 11
- Step Three - Save your Work ............................................... 12
- Step Four - Synchronize with Central (SwC) ....................... 13
- Step Five - Close the Central File! ..................................... 14

## Your Username ................................................................ 14

## Why a Local File ............................................................... 15

## Creating your Local File .................................................... 15

## Synchronizing with Central (SwC) ................................. 16

## Reload Latest ................................................................. 16

## Working in the Central File ................................................. 17

## Active Workset ................................................................ 18

## Ownership vs. Borrowing ................................................. 19

## Open versus Closed ........................................................ 20

## Techniques ..................................................................... 21

- Project as a Template ....................................................... 21
- Manage Linked Files ........................................................ 21
- Dummy Files ................................................................. 21
- Security Guard mode ....................................................... 22
- Gray Inactive Worksets .................................................. 22
- Make Elements Editable .................................................. 22
- Improve Your Borrowing Experience .............................. 22
- Making a new Central File .............................................. 23
Working by yourself ........................................................................................................... 24
Clean Up after Others ........................................................................................................ 24
Worksharing Monitor (WM) ................................................................................................. 24
History .................................................................................................................................. 24
Ghost Users .......................................................................................................................... 25
Notifications ......................................................................................................................... 26
Conflict ................................................................................................................................. 26
Wrong Workset ..................................................................................................................... 26
Can’t Edit Element ............................................................................................................... 27
Changes are Temporary ........................................................................................................ 27
Can’t Grant Request ............................................................................................................. 28
Can’t Synchronize ................................................................................................................ 30
Now Own Element But ......................................................................................................... 30
Created Elements are Not Visible ....................................................................................... 30
Getting Keyed ...................................................................................................................... 31
Delete Elements ................................................................................................................... 31
The Difference between Stand Alone and Central/Local Files ............................................. 31
Can’t Create New Local ....................................................................................................... 33
Borrowers Who Don’t Exist ............................................................................................... 33
Undo History ......................................................................................................................... 36
Hardware Matching ............................................................................................................ 36
File is Being Accessed ........................................................................................................ 36
Accidental Ownership ......................................................................................................... 37
Open and Crash .................................................................................................................... 37
Shared Coordinates ............................................................................................................. 37
Editing Request Granted ..................................................................................................... 38
Workflow Quirks .................................................................................................................. 38
Duplicate Mark Values ........................................................................................................ 38
Filters Rename ...................................................................................................................... 39
Element Name Changed ...................................................................................................... 39
Stair Sketches Can’t Complete ............................................................................................ 40
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Central File</td>
<td>56</td>
</tr>
<tr>
<td>Local File</td>
<td>56</td>
</tr>
<tr>
<td>Synchronize with Central (SWC)</td>
<td>56</td>
</tr>
<tr>
<td>Save Local</td>
<td>56</td>
</tr>
<tr>
<td>Reload Latest (RL)</td>
<td>56</td>
</tr>
<tr>
<td>Ownership/Borrowing</td>
<td>57</td>
</tr>
<tr>
<td>Owner</td>
<td>57</td>
</tr>
<tr>
<td>Borrower</td>
<td>57</td>
</tr>
<tr>
<td>Editing Request</td>
<td>57</td>
</tr>
<tr>
<td>Make Elements Editable</td>
<td>57</td>
</tr>
<tr>
<td>Make Workset Editable</td>
<td>57</td>
</tr>
<tr>
<td>Relinquish</td>
<td>57</td>
</tr>
<tr>
<td>Editable</td>
<td>57</td>
</tr>
<tr>
<td>Not Editable</td>
<td>57</td>
</tr>
<tr>
<td>Viewing/Interaction:</td>
<td>57</td>
</tr>
<tr>
<td>Active Workset</td>
<td>57</td>
</tr>
<tr>
<td>Open</td>
<td>58</td>
</tr>
<tr>
<td>Closed</td>
<td>58</td>
</tr>
<tr>
<td>Gray Inactive Worksets</td>
<td>58</td>
</tr>
<tr>
<td>Maintenance/Special Features:</td>
<td>58</td>
</tr>
<tr>
<td>Compact Central File</td>
<td>58</td>
</tr>
<tr>
<td>Detach From Central</td>
<td>58</td>
</tr>
<tr>
<td>If There is Time</td>
<td>58</td>
</tr>
<tr>
<td>Revit Server</td>
<td>58</td>
</tr>
<tr>
<td>Q&amp;A</td>
<td>59</td>
</tr>
</tbody>
</table>
BASICS

Worksharing seems intimidating or difficult at first for two reasons, new language and rules. It isn't really that bad though. We each adjust to language and rules in our own way. After the initial period of adjustment it won't be so intimidating anymore.

Why Worksets?

They exist to allow us to share the same project data model. The typical structure of software is based on single user access, file level locking, and one user per file. Obviously file level locking or control won't work if the whole project is in one file. They had to provide a way to share the model with others, at the same time. As a result they created a way for Revit to allow for element level “locking”. Imagine the in-place Xref editing feature that AutoCAD allows, file level editing access to an external reference. If we could all do that at the same time in AutoCAD that would be very much like worksets provide to us in Revit.

The Library Metaphor

The notion of a library was in the back of the developer’s minds when worksets came about. The words or phrases we use to describe certain activities, like borrowing or checking out a workset, are shared with library interactions. As such I’ve always used a public library as a way to help describe worksets. In Figure 1 library concepts are in black and the equivalent Revit concepts are in red.

![Figure 1 - Library concepts compared with Workset concepts](image)

The library in this metaphor is the Central File, Revit's term to describe the main project file. Our library card is a personal copy of the Central file, called a Local File. This library card tells Revit who we are and what books we want to borrow or have borrowed. The bookshelves in the library are called worksets. The books in our library are equivalent to every element in our project. That means walls, doors, ducts, pipes, outlets, switches, windows, sheets, views, line types, dimension styles, text, tags, design options...yes...everything is a book. Revit also allows us to create these bookshelves and decide which books should go on them (worksets). We only have to concern ourselves with the 3D elements of our building (virtual representation of real things). The annotation like dimensions, text, symbols and tags are managed by Revit for us.
The phrase “Synchronize with Central” (or SwC) is a command and the phrase “Relinquish Editable” is an option for the command. They both refer to how we return our books to the library so others can read them. The command “Reload Latest” allows us to see what other people have done without letting them see what we’ve done yet. That’s not really part of the library metaphor but important to Revit use.

We work on our project by choosing books from shelves (from the available inventory), we read them, and then when we are finished with them we return them. When we create new elements we act as librarian briefly by adding a book to the library’s inventory as well as choosing which bookshelf they should be place on. The notion of a bookshelf is really just something elements have in common with one another, not a real shelf. Elements in Revit don’t physically move when they are assigned to one workset or another.

How Many worksets?

In our project, in our library, we get to decide how many worksets (bookshelves) we need. We only need to create shelves for the building elements we are using (virtual real stuff versus annotation). Technically Revit only requires a single workset to function. That’s the first workset it creates, called Workset 1 by default. Using several worksets will make it easier to define what your computer is loading into views and RAM. The whole model is still there, just not visible, that'll provide some relief for performance sake. How tangible or detectable that performance gain really is will depend greatly on the scope of your project. Recently I was working with a team, their building is a million square feet overall, and opening the whole file took considerably longer than selectively focusing on a single sector's workset.

This means each project will very likely have its own workset structure based on its own conditions. There may be some worksets that many projects have in common, but it isn't the same thing as a Layer standard in AutoCAD for example.

There are four kinds of worksets; User Created, Views, Project Standards, and Families. We deal with User Created (as the name implies) and Revit manages the rest for us. We don’t create those or ever have to worry about managing those (the other three). As you know, annotation elements belong to the view they are created in. That means View worksets are like bookshelves for annotation. This means that we don’t have to worry about assigning text to a workset, putting them on the right bookshelf. Revit takes care of that for us. We only have to be concerned with which workset model elements are assigned to.

These are some examples of user created worksets. This is not an exhaustive list and some represent different perspectives (architect vs. engineer for example) about organizing a project. This means some of these might end up in your project while others certainly won’t.

- Building Shell or Envelope
- Vertical Circulation
- Grids - Structure
- Grids - Architecture
- Wing West
- Wing East
- Wing South
- Wing North
- Floors Retail
- Floors Condos
- Floors Rental
- Floors Residential
- Floors Examination Rooms
- Floors Laboratories
- FFE (Fixtures, Furniture and Equipment)
- FFE NIC (by owner that may be easier to manage alone)
- Mechanical AHU1 (all connected equipment related to AHU1)
- Mechanical AHU2 (all connected equipment related to AHU2)
- Telecom
- Security (this scope usually has "need to know" restrictions)
- Linked Files - Architecture (Revit)
- Linked Files - Structure (Revit)
- Linked Files - Mechanical (Revit)
- Linked Files - Plumbing (Revit)
- Linked Files - Electrical (Revit)
- Linked Files - Telecom (Revit)
- Linked CAD - Architecture
- Linked CAD - Mechanical Contractor
- Linked CAD (etc... a separate workset for each intrinsically related DWG file may be advisable)

The list is focused relationships between elements and who is working with them (workflow). Is this part of the building separated obviously from another part and therefore could easily be dealt with separately? When I mention improving performance I don't necessarily mean that the file will suddenly be 10x faster. I mean that Revit won't waste any time displaying information that I don't need to see right now, not here in this view or any view until I choose to open that workset(s) again.

Keep in mind you don't really need a workset for something that is already assigned a Revit category, like doors or windows. That's what the category is for. It is rare you don't want to see any doors. It is more common to think I don't really need to see the doors on the west side of the building along with everything else on the west side of the building. An exception would be levels and grids. We can use different worksets to make it easier to manage when we see various levels and/or Grids by expanding on how we assign them to worksets.

Unlike a library, our use of worksets isn't meant to be fixed or rigid. A library might increase or decrease the number of shelves they dedicate to one topic like science fiction but they aren't likely to eliminate the topic altogether. You can expand their number if necessary and you can
collapse into fewer if the need no longer exists. I let Revit (and a team) speak to me. The conversations I hear help guide me into using more or fewer worksets.

A collateral benefit of using worksets is added control over the visibility of elements via Visibility/Graphics. Using them for this purpose can cause a fair number of worksets, even some of those in the list above are motivated by "seeing" elements or not. In my case I'm interested in closing their worksets, "hiding" their elements globally until I need them again which is still "seeing" but not from a documentation standpoint.

The risky part of using worksets to help manage the visibility of elements is it requires all of us to be consistent, to assign elements to the correct workset every time. That can be quite elusive. As such, keep in mind that the visibility control over elements that Worksets provide is really a collateral benefit. It is not their purpose. Their purpose is to allow multiple users to concurrently work on the same project.

Filters were added for controlling visibility of elements on a view by view basis. You'll get a lot more mileage and options with them than using Worksets for visibility. If you reach for a workset to manage element visibility you owe it to yourself to consider a deeper look at Filters. Remember that with Revit 2013, because of the changes to how View Templates work, Filters get sharper teeth because of the greater integration between Views and their Templates.

For Revit MEP users Filters are much more necessary because many disciplines share a few categories that each other don't wish to see in their views. A water heater for example is Mechanical Equipment. I may not want a water heater visible in my HVAC floor plan and the plumbing views probably don't want to see a VAV in them. Both are assigned the same category. Filters will let us deal with that reasonably well, assuming the content is organized well. A workset might be tempting but we've got to remember to assign every element to the correct workset for it to be reliable. I've been using Revit a long time and I still forget to do it.

The single advantage that a workset has over a Filter is the ability to affect an entire project with a single change either while you are opening a project or when you need to transition to completely different portion of the project. The effectiveness of worksets for this does hinge on users remembering to correctly assign the Active Workset as they work. Yes, it is easier said than done.

Creating the Central File

When you are ready to prepare your project for using worksets you'll need to take these four steps.

- Step One - Good Location/Name (server folder and nice name)
- Step Two - Enable Worksets (adds two parameters to DB)
- Step Three - Save your Work (commit the DB changes)
- Step Four - Synchronize with Central (SWC, return your "books")
If you scroll first you'll realize that these four easy steps seem quite long. It takes me some time to explain them in writing but they don't take long in practice. The variable in how long it will take has to do with when you decide to start the process. Doing it earlier with very little in the model will take less time than doing it later with most of the project modeled.

Step One - Good Location/Name
Application Menu > Save or Save As - Browse to your server and project folder. Use a "good name". This is the most important step, if done incorrectly you can still fix it now. If you wait until after step three, you have to "start over".

Keep in mind, a central file must be created in the correct folder at the outset. If you create a central file in the wrong folder and then try to move it Revit will change the way it regards the file, it will be considered a Local File instead. That might be confusing but that is how Revit is "wired". It records the original folder location (and file name) and if it is moved the file is treated as a Local File instead.

Similarly you must use the name you really want, now. There is no changing your mind or the same thing happens, Revit considers the new file a Local File.

This means you need to pick your location and name carefully and then act. If you make a mistake, fix it NOW, assuming you realize it is wrong.

Step Two - Enable Worksets
Collaborate ribbon > Worksets panel > Worksets button or alternatively its "little brother" the Workset button on the status bar (see Figure 2).

Revit opens this dialog to confirm your intentions (see Figure 3).
Clicking OK causes Revit to add two parameters to every item in the database, Workset and Edited By. Using the Library metaphor these are the bookshelf and the library card user information. I recommend you keep the two default worksets that Revit offers you (keep them intact). If you need others you can create them once Revit offers you this dialog box (see Figure 4), as soon as it is finished adding the two parameters to the database.

Keep in mind that Revit won't let you delete "Workset1" now or in the future. You can rename it but you might find yourself confused later if you want to delete "that" workset and Revit won't let you. For this reason I tend to keep it as a "catch-all" workset for items that I can't decide how to organize yet. You can delete "Shared Levels and Grids" if you like or rename it but again it serves an "obvious" role. Obvious worksets work better for everyone.

Step Three - Save your Work
Application menu > Save - This is your last chance to bail out. This dialog confirms that you are going to commit your changes to the database and your project will become a real Central file.
This commits the changes to the database structure that we made in the previous step. The file is now a Central File and you've gone beyond the point of no return. Any changes to the file location or name are going to require special action on our part to keep our team collaborating.

**Step Four - Synchronize with Central (SwC)**

When we started this process, once we moved beyond Step Two we became the "owner" of everything in the database. When we did Step Three Revit returned most of the elements we borrowed for us but it did not return the User Created worksets. We need to use SwC in order to return everything so that the whole team can begin working on the project. There are two SwC options to choose from; Synchronize and Modify Settings and Synchronize Now (see Figure 5).

![Figure 5 - I say avoid the SwC Now button.](image)

As a rule I don’t encourage people to use Synchronize Now because it doesn’t give us a chance to make sure everything is relinquished or to add a comment when we do decide to use SwC. This is the dialog Revit (see Figure 6) presents to us when you use the Collaborate ribbon > Synchronize and Modify Settings.

![Figure 6 - The SwC dialog gives us a chance to relinquish and save a comment.](image)

Note the check box for User-Created Worksets and the comment field. We need to check that so we will pass ownership of them back to the Central file. Also do yourself and anyone else who works on your project with you a favor, add a comment. Put something meaningful in each time you bother to use SwC. This can help us track down a problem later or to determine how far back to go when you must start "over" again at an earlier point in the project's design phase.
These comments are visible when using the Collaborate ribbon > Show History or Backups features. You now have a Central file ready for your team to begin collaborating!

**Step Five - Close the Central File!**

Okay, you caught me, there is a fifth step! Application menu > Close - This closes the central file and from now on nobody should work directly in the central file, as a rule. There are exceptions to the rule but more on that later.

**Your Username**

How does Revit know I am different from anyone else? Your User Name in Revit defines who Revit “thinks” you are. This is found via the Application Menu > Options button > General Tab (see Figure 7).

![Figure 7 - The username field in the Options dialog.](image)

Why does it matter? In a stand-alone project it doesn’t. In Workset projects it matters a lot! Two users with the same user name are not regarded as two people with the same user name. They are regarded as the same person working in two files. We don’t want Revit to think this because the first of two “Mike’s” to Synchronize with Central wins!!! The other Mike can’t use SwC at all. He’ll have to resort to using copy to clipboard and paste to get the work he did into the project.

A good strategy is to just use the same user name you log into your computer with. IT needs your user name to be unique and so does Revit, “That was Easy”. This is the default behavior when Revit is installed. Each unique user that runs Revit (logs into the computer) will have their username stored. If another person uses your computer, logs into it with your credentials, they’ll end up changing “your” username to theirs. This means the next time you use your computer (if you don’t check) that you will end up working as “them” instead of as yourself. The moral of the story is to check your username.

The user name is stored in the Revit.ini file (a general and user specific version are located in two different locations) and looks like this Username=MyName. If you delete the name Revit will use the logon user name of the next person to run Revit. If you use Application menu > Options > General tab > Username to set the user name it gets stored in Revit.ini and the cycle begins again.

When you use the current methodology to create a local file it will include your username in the file name. A quick glance up at the name will help you realize if you are not who you think you are. If you aren’t, just close the file without saving (relinquish all elements), change your username, make a new local file.
One tip that surfaced during a discussion at Revit Technology Conference a few years ago was to add the telephone extension to a person's Workset username. This makes it easier to contact that person if you need to because you'll see their extension as part of their username when it appears in warnings or messages.

Why a Local File

Working in a separate file gives us a sandbox to play in until we are ready to share what we've done. The segregation of our data just gives us our own space to design and contemplate what is necessary. If we could see everyone else's design process live on screen it might be maddening to watch things show up, change, disappear and reappear in a different location. A common reason offered in years past was that the local file is stored on your own PC which could improve performance by moving most tasks to your PC instead of relying on network access. That might be somewhat relevant but it isn’t THE reason we use a local file.

Creating your Local File

I believe that new local files should be created at least each morning and used throughout the day. In fact, more often than not these days, I create a new local file anytime I start working on a project. The current process implemented in Revit supports this thinking by allowing people to easily create a new local (by default) and append a time/date stamp to previous local files when creating a fresh one each morning.

For a little history (my own perspective), my original reasoning for new locals each day, beginning years ago, was never about corruption. Questions in user forums about this habit suggest that creating new local files is/was necessary due to frequent corruption. Not in my experience, for me it was always about faster loading and not having to remember to use Reload Latest and to wait for that too.

Originally it was recommended to open a central file and use Save As to create a local file. This meant we had to wait twice for the files to open (still does now technically too). Once for it to open and a second wait for it to save as. Users discovered that we could accomplish the same task by using copy and paste instead of "Open/Save As". That approach resulted in a routine greeting with an error message that confirmed we were working in a local file. Still, for me it was all about being "faster".

From a data integrity perspective, sporadic team involvement means we have inconsistent local file "sell by dates". Mine was current but "Joe's" is three days old when he opens it because he’s been out of the office for the last few days. If he starts working in that file he’s got at least two day's data to acquire by using "Reload Latest". If he creates a new local instead he’s in sync within the window of other people working "today", between when they started and he arrived and started...pretty close to in sync.
The reports of file corruption I encountered from time to time did go away (stopped happening) when people made new local files each day too; surely a happy coincidence. Is it necessary? I don’t know, but it worked to resolve those situations. The corruption probably had more to do with inconsistent participation and “sell by dates” being out of alignment than the central/local file arrangement itself. I’ve even seen an error message tell me that my local file is too old and to make a new local file, so clearly Revit is thinking about it too. In the end, it was easier to “eat an apple a day” (make a new local) than to wait and see if the issue would arise again.

Since the addition of the new local file option I’ve abandoned other “custom” local file techniques but I still create a new local each day, it isn’t any slower than opening an existing local file and I don’t have to remember to use Reload Latest. I am getting older and my memory isn’t what I remember it to be. In fact I tend to make a new local anytime I start working on a project again, regardless of whether I created a new local a few hours ago. Large teams can create a lot of change in a short period of time.

**Synchronizing with Central (SwC)**

As I mentioned earlier, there are two options; Synchronize and Modify Settings and Synchronize Now. These are both a two-way conversation between the central file and your local file. One direction is pulling changes from the central file and the other is pushing your changes to the central file. The first and best is Synchronize and Modify Settings (see Figure 6). I claim it is the best because it offers you a chance to review the worksets you’ve interacted with as well as store a comment about why you are using SwC.

Just remember the biggest difference between Synchronize Now and the better one is that it does not relinquish User Created worksets (normally bad) and it does not give you a chance to store a comment (really bad in my opinion).

**Reload Latest**

This is a one-way conversation between your local file and the central file. It just pulls changes from the central file to your local. This is very much like reloading an external reference in AutoCAD to see changes others have made to their files. In this case you see the work contributed to the model by others. It might be right next to or around you or completely outside of your scope or interest. Either way it is added to your local file so you’ll see it when you end up in a view that can show the changes to you. Use Reload Latest when you are only interested in changes made by others, assuming you aren’t ready to share your own changes. If you are ready to share your work with others then SwC makes more sense.

On very large projects with a significant number of users working on the same central file through their own local files it may actually be advisable to use Reload Latest before using SwC. If you consider the volume of data from everyone else may be significantly greater than your contributions, using Reload Latest first allows Revit to reconcile everyone else’s work with your
local file before added you own changes to the mix. In this kind of situation the sequence will look like this:

- Save Locally
- Reload Latest
- Save Locally
- SwC

These separate steps do mean that it will take a bit longer to complete the whole operation but it will likely ensure that you don’t encounter error messages. I’ll repeat that I’ve only encountered a need for this on very large project files and teams. What is very large? Large in this context means files that are 300+ MB and teams of ten or more all accessing the same central file routinely during the day. Then again each project is unique so your mileage may vary.

**Working in the Central File**

Once worksets are enabled we are supposed to work in local files. Occasionally people will fire up Revit and end up working in the central file instead. These days it is more deliberate because they have to un-check the Create New Local option (see Figure 8) to do so.

![Figure 8](image)

Revit has been tweaked over the years to deal with central file editing. If I’m working in a local file I can use SwC as much as I need. If someone is working in the central file on the other hand, they will be forced to create a local file when they try to use SwC, assuming other users have made some changes to the model via their local files. The person working in the Central file will get this message.

![Figure 9](image)
We can work in a central file as long as necessary, as long as nobody else is working on the project in local files too. Isolation is fine but collaboration is where the conflict arises. We encounter the message above (see Figure 9) when a change has been made via a local file while the central file is being edited. While someone is editing the central file a local file user has been able to add or alter an element and then use SwC. When the local file user completes a SwC and the person working in the central file attempts to use SwC the central file editor will get the message (remember Save is disabled in a Central File). The message also explains how to resolve the situation, just Save As to save the file with another name, which turns it into a local file. Now SwC will work. There are some legitimate reasons to work in the central file, just do it in isolation.

Active Workset

What is the purpose of the Active Workset setting? You see it in the Workset panel, the Workset Dialog or on the Status Bar (see Figure 10). It’s meant to allow us to deliberately and automatically assign elements to a specific workset, as long as we remember to set the Active Workset first.

![Figure 10 - Three places you'll find the Active workset displayed.](image)

When you add elements to your project they are assigned to whatever the Active Workset is. Referring back to the Public Library as a metaphor for worksets, the active workset is the book shelf you are standing in front of as you prepare to put books away. You need to go to the non-fiction section before putting any non-fiction books away. If you put a fiction book on the non-fiction book shelf people will have a hard time finding it later. In Revit elements don't go anywhere when you assign them to the wrong workset but it does make it harder to be effective when choosing to open or close a workset if things aren't assigned properly.

*This means Active Workset is your nemesis*

As it happens there is an equivalent concept of this nemesis in AutoCAD, it's the notion of Current Layer and the Current Layer selection tool on the ribbon. In the same way, if you don't change your current layer in AutoCAD before sketching lines you'll end up with them assigned
to the wrong layer. There is a significant difference however. In AutoCAD you are probably used to this Current Layer display changing to show the layer of a selected element. Revit’s Active Workset display does NOT do this. This means you can't reassign the Workset parameter here. You can only do that in the Properties Palette.

Much the same way Current Layer has been problematic in the past, try hard to remember to set your Active Workset.

Ownership vs. Borrowing

This little piece of information (see Figure 11) confuses people too often.

![Figure 11 - Confusing language "Not Editable".](image)

An experienced user will ignore it. So will a beginner, so will most anyone. The difference is an experienced user will be able to tell you what it means, maybe. I say maybe because sometimes explaining it just leaves people more confused.

In the simplest terms, if you see it...that is good, not seeing it...is not. Yes, I know that doesn't help understand why or what it means.

In an attempt to explain it I'll fall back on my library metaphor. We can borrow books from a library, not the shelves themselves. We could try but I bet the librarian would object. We could borrow every book from a shelf. They just won’t let you take the shelf too. In Revit User Created Worksets are bookshelves (and Revit doesn't care if you take a shelf too) and everything in the model is a book. With this in mind these two terms take on the following meaning.

- **Owner** = you take possession of a bookshelf
- **Borrower** = you take possession of a book(s)

In the workset dialog (see Figure 12) we want to avoid being listed under the Owner column because that means we've taken an entire shelf and all of its books from the library. Imagine the librarian running down the street after you.
After figuring out what worksets you should create, the most important thing to care about is which workset is the Active Workset, remember that from the last section? I'd argue that nobody would get confused if (Not Editable) wasn't there at all. When you add elements to the model they are assigned to the Active Workset. That is simple enough. We can add new elements to a workset even if we are not the owner of the workset.

That's significant enough to repeat, we add new elements to worksets even if we are not the owner, the workset only needs to be the Active one. There is no confusion on Revit's part about new elements. We created it therefore we are the owner but it is assigned to the Active Workset, which we don't have to own. This is an important distinction.

So the notion of editable or not really is a bit too subtle for the typical workflow of Revit users. It's technically accurate but a bit irrelevant.

For now, remember that seeing (Not Editable) means you have NOT taken ownership of any User Created Worksets...and that's GOOD. If you don't see (Not Editable) that means you have...and that's NOT good, usually.

**Open versus Closed**

In Figure 12, while I've got the Workset dialog open, I highlighted the Open/Close buttons (item #1). Those define whether Revit loads the elements assigned to the corresponding workset or not. Open means load them up and closed means don't show them anywhere, a global result.

Referring back to the previous topic Ownership vs. Borrowing, item #2 is for taking ownership of a bookshelf (workset), which isn't necessary or appropriate most of the time. The Editable column next to Owner changes to Yes if you take ownership of the User Created Workset and your username will appear in the Owner column. Don't do that, at least not routinely.
TECHNIQUES

Project as a Template

As you may have discovered already we can’t enable worksets in a project template. If you want to establish some office standard worksets then you can consider using a project file with worksets enabled as your template instead. This means when a project gets started you need to use to create a new central file from this project file instead of the typical “Click the New button” approach. You can either open this project file and use Save As or use Detach from Central to get things rolling.

Manage Linked Files

It is common practice to use a distinct workset for each linked model and even DWG or DGN file you need to include in your project. We use a separate workset for each linked Revit files for a couple reasons. One is to avoid the manage links reload/unload process and the other is we get to decide which linked files to load when we open our local file.

Reloading and unload linked files in a worksharing project is quirky. The Unload option in the Manage Links dialog is regarded by Revit as a global change and all users see this occur when they use SwC. That means if I unload a link that everyone will eventually get the link unloaded too. That’s not always what we want as a team. I can avoid affecting others if I remember to reload the link before using SwC, but I have to remember. The Reload option however is treated as a local change which means that we all have to play the unpleasant game of reloading links to see what we want. If we assign each link to its own workset and only open and close their workset instead the change is regarded as local only (personal change). This allows us to avoid the reloading game entirely.

The performance aspect of unloading and reloading a linked file is essentially the same whether done with a workset or with manage links but we do benefit from being able to decide whether or not to load linked data by specifying which worksets shall be opened while opening our local file. We’d only be able to do that after waiting for the file to load everything if we just relied on the Manage Links dialog instead.

Dummy Files

It isn’t the most obvious concept but you can use empty project files to preset visibility/Graphics standards for linked project files. You use Manage Links > Reload From to replace them once your consultant models become available. This can be quite effective if you discuss your needs with all the consultants during a BIM Kickoff meeting.
Security Guard mode

You want to make Revit interfere with users who seem determined to delete or move important elements repeatedly. This can be done by checking out a specific workset like Shared Levels and Grids for example. When you SwC you just refuse to return the User-Created Workset or other specific worksets if necessary. This should be done using a special username, different than the rest of the team like Super Admin for example. Keep in mind that this is merely a modest hurdle for an experienced user to get past but it should deter the average user enough to point out that they were getting in harm’s way.

Gray Inactive Worksets

We can use this setting (see Figure 13) to help us because remembering to set the Active Workset is hard enough as it is. It is a local setting, meaning it only affects what we see.

![Figure 13 - The Gray Inactive Workset button.](image)

This is useful because it helps us see the contrast between elements we are either adding to the model or are working with. If the furniture in the view is gray (half tone) and we don't think it should be then it will help us realize that the active workset isn't set correctly.

Make Elements Editable

We can borrow elements by simply interacting with them, such as moving or changing their parameters. As long as nobody else has already done something to the element Revit will let us do it. This is referred to as transparent element borrowing, it happens as we work with as little interruption as possible. While this is efficient from a click to click perspective it may not be the most efficient way to work if we are going to borrow a hundred elements this way, over the course of a few minutes.

Improve Your Borrowing Experience

When you select an element and move it Revit has to see if it is currently borrowed by anyone else. As I just mentioned, this is referred to as transparent element borrowing. Click on a wall and drag it to another location. The slight or pronounced delay you "feel" or "see" between starting to drag and seeing it happen is affected by the "back office" work that Revit does between your local file and the central file.
You can alter the perception of responsiveness some by borrowing elements explicitly instead. I wrote perception because it may not actually take less time overall but you will experience less lag/drag/delay if the borrowing process is already sorted out.

What I'm suggesting is that you select the elements you intend to work on first, then right-click and choose "Make Elements Editable" (see Figure 14). This is an explicit instruction to Revit, "I want to borrow these!" It will check and let you become a borrower of everything that is currently available. Anything that isn't will generate a message to that effect. You'll still have the rest though.

![Figure 14 - Using Right Click "Make .... Editable".](image)

This means you can open a view, decide what you want to work on, select the elements and finally make them editable. Then go about your business. Synchronize with Central when you finish with them and repeat.

Remember that the right-click option "Make Workset Editable" is like becoming an Owner of a "shelf" and that normally isn't the best strategy. It will however accomplish the same thing as I suggest above, just for potentially many more elements than you need.

**Making a new Central File**

You can use Detach from Central (DFC) to create a new central file instead but it still involves at least selecting the original central file. If you don't use DFC then you use Application menu > Save As > Click the Options Button and check the option: Make this a Central file after save (see Figure 15).
Figure 15 - Check the box for “Make this a Central Model after save”.

Working by yourself

If you are the only person working on this project routinely and you just want to take advantage of collateral benefits of using worksets then you can just work in the central file. There is nothing wrong with working in a local file instead however and it is even a good idea as it gives you some additional redundancy should something go wrong with your network or computer(s).

Clean Up after Others

Sloppy workset use means lots of elements on the wrong worksets. It is easiest to clean up after everyone when nobody is working in any local files. Open the central file, borrow every workset and get to work.

*Keep in mind that the absolute best time to fix this is when it happens. That means when you or someone else adds an element to the project assigned to the wrong workset, fix it now before you use SwC.*

Worksharing Monitor (WM)

History

This is my favorite part of worksharing monitor though the button to access it is pretty sneaky (see Figure 16).
Notice that Floyd is currently working in the Central file? Floyd is red. Don’t be red Floyd!

Back to the history feature, this gives us a quick snapshot of the synchronization history of a project file (see Figure 17). If we see that the typical SwC take a few seconds or maybe a minute for several days but suddenly starts reporting ten minutes for a typical SwC we can quickly see when a potential issue began. It is a great little feature to “take the temperature” of a project now and then.

**Ghost Users**

Since the Worksharing Monitor is a separate application it relies on data that Revit makes available to it every few minutes. This means when you first get started working there is a period of time where we can’t really trust the list of users that we see as being actively working on the project. The list will refresh after a few minutes and then we can be reasonably certain it is accurate. There will be times at first where you can see duplicate users or users that were in the file last night but not now. This should clear up shortly after getting started. When someone leaves their local file to work on something else they can remain in the list for a short time too.
Active users will appear at the top of the list and you’ll see their activity (SwC, RL and saving their local file). User that are not running WM will appear in the list in gray and those that are using WM will be listed in black.

Notifications
A frequent complaint in the past about Revit was that it didn’t tell us when our file was out of sync with other’s work or that we had an editing request. Revit is doing a better job of it apart from WM but it has been providing interim improvement in that regard in the meantime. If we use WM as a team it can help us be more aware of these two conditions more easily, editing requests and file status, as well as others. If we use the option “When the central file changes” we’ll get a notification each time another user updates the central file with their changes (see Figure 18). This gives us an opportunity to use Reload Latest to get back in sync.

CONFLICT

Wrong Workset
As mentioned earlier, the best time to fix elements that are assigned to the wrong workset is when they are first created, by the person doing the work. If that doesn’t happen then it is easy to reassign elements. It is easier see the issue by isolating elements according to the workset they are assigned to. One way to do this is using Visibility/Graphics overrides. Another is to use the relatively new Worksharing Display options mentioned elsewhere. Once you can see which elements are assigned to the wrong workset you can reassign them by editing the workset property in the Properties Palette. You may run into conflict with other users actively working on some elements. If this happens you can either delegate the task to them since they are already borrowing them or you can wait until after everyone else is out of the project.
Can’t Edit Element

When you attempt to change an element you may encounter this dialog (see Figure 19).

![Image](image.png)

Figure 19 - Bubba is in the way, he’s borrowed something I want.

This means that someone else beat you to it so to speak. They are already doing something to it. You can either ask them to complete the task you want to see done, wait till they finish and relinquish it or create an editing request. I prefer to avoid editing requests in favor of asking them to take care of it, especially since they are already borrowing the element.

Changes are Temporary

This concept relates to views as a book, something we can borrow. It is a bit more abstract but keep in mind that views are also a bookshelf for books, such as tags, dimensions, text and symbols (2D annotation information). When we open a view we are simply looking at the model through the view. If we decide to change the view from showing elements with Graphics Style: Hidden Line to Shaded we are now changing a property of the view itself. This means we are no longer just looking at a view but altering, and in turn borrowing, something about the view itself. When this occurs we are listed as a borrower against this view. If someone else attempts to do the same thing or change some other aspect of the view they will usually receive this dialog (see Figure 20).

![Image](image.png)

Figure 20 - Changes will be temporary because someone else has already changed it.

This means that the change they want to make can’t be preserved, meaning using SwC won’t apply their change, but they can at least see what the result of the change would be. This is a special condition that is only true for views. We can’t do the same thing with a door or a wall for example. If someone changes a door then it is locked out for everyone else. A view on the other hand is a bit more flexible. The first person to alter the view will be able to use SwC to preserve their changes and all others will only be able to see the view change. If you think about it for a bit you’ll realize that we often want to manipulate a view temporarily so we can evaluate something more easily.
Revit started creating separate 3D views for each user as a result of this competition for views. So you’ll see that when you first click on the Default 3D View button you’ll get a view that has your username appended to the name, like this “{3D - sstafford}” (see Figure 21). Bubba says, “That’s my view, you better leave it alone!”

![Figure 21 - Revit makes user specific 3D views.](image)

**Can’t Grant Request**

When you review an editing request (see Figure 23) from someone else and decide to grant it you may get this message (see Figure 22).

![Figure 22 - Can’t grant this editing request because the elements were changed.](image)

This deals with the subtle difference between possessing an element and changing an element. You can borrow an element explicitly when you use the right click context menu option “Make element editable”. You now possess the element but you may not have changed it at all.

*Imagine borrowing a pencil from a co-worker. You put it in your pocket. Later they ask for it and you apologize for keeping it, you forgot you had it, and hand it back to them. That’s the same as granting a request. You simply possessed it. If you had broken the pencil in half then you’d have to apologize for both having it and breaking it. In Revit’s case you would not be able to grant their request because breaking the pencil is altering it, changing it. Changes must be reconciled with the central file with SwC.*

This means that you can grant a request if you merely possess an element. If you have changed the element in some way you’ll only be able to resolve their request by using SwC. The Editing Request dialog offers a bit more feedback now too (see Figure 24).
Figure 23 - Two different buttons for editing requests will open the dialog.

It shows two separate messages at the bottom that explain the icons that appear next to the editing requested items (see Figure 24). As shown the * is used for elements that will require using SwC and the ^ is used for deleted elements that will be ignored.

Figure 24 - More detail offered in the list of request.

Revit also offers us better feedback when we use Editing Requests (see Figure 25).

Figure 25 - Editing request giving better feedback now.
Can’t Synchronize

This message isn’t a good one. If you get this message it may be the result of opening the central file directly. If you have done so you can either close it, possibly losing the work you’ve done or use Save As to create a local file. You’ll be able to use SwC once you’ve established your local file.

It can also occur if your network is experiencing a problem and your local file can’t connect to the central file anymore. It’s time to reach out for some assistance if you get this message and you aren’t working in the central file.

Now Own Element But

This occurs when you and others are competing for the same element and they’ve either granted your editing request or they’ve used SwC. The message (see Figure 26) tells you that you’ll need to use Reload Latest to see the changes before you’ll be able to complete your task. In other words, maybe you should see what others have done to the element before you decide to act. If they moved the wall already you might not realize it is already in the correct location or that they changed the door swing already. The solution is offered in the dialog, just follow its advice.

![Figure 26 - You need to reload to see what's changed before carrying out your work.](image)

Created Elements are Not Visible

When you choose to close or turn off a workset you may run into this error message when you add elements (see Figure 27).

![Figure 27 - Can't see the element I just added, the workset isn't open or on.](image)

This happens when you add new elements and your Active Workset is one that you’ve chosen to close, using the Workset dialog, or turned off, using Visibility/Graphics. There are many other
reasons you could see this error message, in this case it has to do with the Active Workset feature.

**Getting Keyed**

This message means that Revit is busy, “Please listen to this Aerosmith song while you are on hold”. You can either choose to leave the dialog (see Figure 28) in place until Revit gets around to your request or click cancel and try again later.

![Figure 28 - Revit is experiencing high than usual "call volume..."](image)

Keep in mind that if several of us are all looking at the same message that Revit will take longer to deal with this, for all of us. Large teams will often create a queue for using SwC and RL to avoid bumping into each other this way. The end result is faster reconciliation once our turn comes up.

**Delete Elements**

This most often occurs because elements have changed in linked files that Revit cannot reconcile during a SwC. For example dimensions that are referencing elements in a linked file get deleted when some or all of them are altered in such a way that Revit can no longer identify them. Families that are swapped for another that has different reference plane settings are a classic culprit for losing dimensions. Walls that change type or are deleted will cause tags and/or dimensions to be deleted as well.

Unfortunately the only option we get when this message appears it to accept the problem or not open the file. If there are a significant number of elements involved it may be a good idea to expand the warning dialog and attempt to determine which elements are affected, not open the file and return to the linked file and see if you can prevent or fix the issue there first. Success however is frequently less than often than disappointment.

**The Difference between Stand Alone and Central/Local Files**

When you are looking in a project folder on a server how can you tell whether you are looking at a standalone Revit project or a Central and Local File set up for Revit Worksharing? Let's look at a standalone project first (see Figure 29).
If you saved this file 12 times during your first day working on the project you’d see backup files numbering .0010, .0011 and .0012. Revit replaces the oldest of the three with the next backup, cycling through just three backup files. That is because Revit’s default setting is to allow for three (3) backup files. Each time the file is saved Revit will cycle through these three backup files, unless you specify more or fewer backups. The project team and/or IT should/will determine how many there should be.

Now take a look at the project after it is converted into a Central File when worksharing is enabled (see Figure 30).

Notice the naming convention used? The project name with a suffix of “- Central” added. We started doing this many years ago to make it more obvious that you are looking at a Central file. This approach is less effective today when you use the relatively new “Create New Local” option. Now it probably makes more sense to just add the word “worksets” or “WS” to the name instead.

We used to alter our local file by changing the name of the copy by removing “- central” and putting our login name and today’s month/day just to make it distinctly different. Such as: “Police Station - sstafford09/08.rvt”. As mentioned in the previous paragraph Revit automates this part for us now.

The key difference between a stand-alone file and one using worksharing is the matching folder (name matches the file) with the suffix "_backup". Revit adds this folder during the process of creating the Central file automatically (and local files). Pretend this folder has some really "smelly" stuff in there and just stay out. Just like those stickers you see on electronic equipment that warns you with something like: "Warning no user serviceable parts inside - cutting this tape will void the warranty", imagine it says, "There are no user serviceable files in this folder".

Revit’s default worksharing setting is twenty (20) backups. Consider four team members may SwC five (5) times per day. This means there will be a discrete backup file for each time the file
was saved during the day, up to twenty times. If more saves occur then Revit will cycle through
the twenty backups, overwriting the oldest first.

Can't Create New Local

The Revit Clinic confirmed an issue I've been seeing for some time now. If you click on a central
file (since v2010) Revit detects this and checks the Create New Local option (see Figure 31).

If you find this option disabled (grayed out) then that's a clue that something is wrong. The
situation I've observed and the clinic's first listed problem is that Revit thinks that your computer
is finding the central using a different path than other members of the team that have also
created local files. In my case I usually find that someone is showing a path that reflects the
Network Neighborhood path instead of a formal Drive Letter mapping. In a couple of other cases
I've encountered a specific IP address instead. The person that creates the central defines how
Revit expects everyone to connect to it.

If you find the Create Local File option disabled, STOP. Get some help to fix your path before
you find yourself unable to Synchronize with Central.

There are two other explanations offered in The Revit Clinic's post; it is a stand-alone project, in
other words worksharing isn't enabled and a more arcane corrupt central file/thumbnaill issue.

I've also observed that the so called corrupt central file is occasionally really a local file. In this
situation someone saved their local file in the folder where the central file was and chose to
overwrite the file. The unfortunate consequence of this is that nobody else can synchronize with
central because it isn't a central file anymore. Take care, it's dangerous out there folks.

Borrowers Who Don't Exist

It happens often enough, the person that Revit tells you "owns" something you need to deal with
either isn't working on the project actively now or in some extreme cases doesn't even work at
the firm anymore, or is say someone like Elvis? (See Figure 32).
Before I describe how to cope with this don't apply it to a team who has a rogue member who seems to borrow stuff and not return them all the time. That's a separate kind of issue; I'll address that situation at the end of the post.

Most often this missing person has a view, project standard or some other relatively "minor" element checked out, like Elvis in Figure 32. Otherwise it would have caught someone's attention sooner. One way to check for rogue users is to open the Workset Dialog, check all four Workset filters and scroll the list looking for usernames that shouldn't be there, or click the heading twice which will sort the column by the information in the column. This would be true even for regular team members if you are the only one in the project at the moment, say after hours for example.

When you need to remove a person that is no longer part of the team you need to write down the username(s) listed in the warning message that tells you "no, you can't have it because Bubba does".

Take the following steps if you are a timid Reviteer, this uses a Local File.

- Open Revit (if it isn't already, don't open any files yet)
- Close your own Local File (if Revit is already open)
- Change your Username to match the missing person
- Open the Central File, check the Create New Local File option
- You now have a local file assigned to this past user
- Synchronize with Central (SwC)- relinquish all worksets
- Close the Local File
- Reset your Username - Open your previous local file
- Resume work
Take the following steps if you are a brave Reviteer, this opens the Central File.

- Open Revit (if it isn't already, don't open any files yet)
- Close your own Local File (if Revit is already open)
- Change your Username to match the missing person
- Open the Central File (don't check the local file option)
- You are now working in the Central File (most reliable if no other users are working in Local Files, if they Synchronize with Central while you are doing this Revit may insist on having you create a local file before being able to Synchronize with Central)
- Synchronize with Central - relinquish all worksets
- If there are other users involved - change your Username to match another
- Repeat the process (again, other users might force you into a Local File, most reliable if no other users are actively working in the project)

If you'd like to take a brute force approach, definitely after hours, you can:

- Open the Central file - Check the Detach from Central option
- Save the file - provide a new name when prompted
- You now have a new central file
- Synchronize with Central - relinquish all worksets
- Close the file
- Have users create a new Local File based on this new Central File to begin work

If you have a co-worker that didn't relinquish everything before leaving the office the kindest/gentlest approach is to open their Local File on their PC and SwC for them.

- This means you have to be able to log onto their PC as yourself
- Change the Revit username to theirs if isn't already
- Then you have to know where their Local File is, open it
- Use SwC
- Reset their Username
- Close Revit
- Log off their PC.

Let them know what you did! They'll probably see your log-on-name when they return and wonder why.

You can also use either of the first two methods described above, as if they are no longer part of the team. The reason I suggest the kinder/gentler approach is that it protects any changes they made but didn't publish via SwC. That assumes they just saved the Local File but didn't opt to use SwC, which does happen from time to time when people get rushed off to a meeting or emergency. If someone is a repeat offender and shows no remorse you might consider the negative reinforcement of using the less kind and gentle approach.

Now if you are part of a big project, many central files and collaborating with other people that are referencing your files and theirs...slow down, take a deep breath. The same rules apply but before you rush into creating a new central file you'll need to consider file naming agreements.
You'll also need to be mindful of who else is cranking away on the project when you decide to deal with this situation. Go slow, methodical. The turtle may win this race as opposed to the rabbit creating some havoc.

**Undo History**

When you do something that causes Revit to take ownership of a workset first, in order to do what you want, like deleting a view, Revit clears out its Undo "stack".

In the case of a view if you borrow the view first and then delete it the undo stack remains intact. To borrow it first you'd have to select the view's annotation symbol, right click and chose Make Workset Editable. You can do the same thing in the project browser over the view name, choose Make Workset Editable too. You could also alter a parameter and reset it or use the Workset dialog to make it editable.

Same thing happens if you delete a family from the project browser without borrowing it first. To do that you can click Make Workset Editable via Right Click on the family in the project browser too. A component family will offer you the option “Make Workset Editable” while a family’s type will offer you the option “Make Element Editable”. The workset it refers to is the family workset found in the Workset dialog when you select the Families “Show” filter.

It isn't really intended to happen that way. Optimistically we could see a fix in a future update but more likely not till another release.

**Hardware Matching**

Attempt to keep project team workstation specifications equivalent. A dramatically weaker machine specification used by a single team member can reduce overall project performance when they use SwC or RL.

**File is Being Accessed**

Occasionally you might run into a message that warns "the Central file is being accessed by someone else...". This prevents you from successfully syncing with central.

You can try adjusting the frequency that Revit checks for worksharing updates. Beginning with Revit 2012 they added some graphic display features to help us see what's going on with our project and worksharing. It turns out that when people who are suffering with this situation change their setting to Manual only instead they don't encounter the situation anymore.

This setting is found on the General tab of the Options dialog (Application menu > Options, see Figure 33).
It might be worth trying if you’ve run into the issue too. Keep in mind that it will affect the ability of Revit to offer up the new graphic display options for worksharing.

**Accidental Ownership**

There are times when people become the borrower of elements or settings and they can’t figure out how or fail to recognize how their minor interaction could cause this to happen.

**Open and Crash**

When Revit opens a local file it quietly and temporarily borrows from the project standards workset (Project Info I believe). It needs to borrow “something” in order to process the opening of the file. When all goes well this temporary condition is reset, relinquished and nobody can tell it happened. This is not true when Revit crashes while the file is being opened or if someone uses “End Task” in Task Manager to bail out of opening a file for some reason. It happens and when it does that user can end up as a borrower even though they never actually finished opening the file.

**Shared Coordinates**

When files are using shared coordinates a user can end up borrowing project standards again if they accidentally move a linked file. Normally an error message will appear that warns them that the linked file is using shared coordinates and therefore their action will affect the position of the file. If they disregard the warning and fail to undo the change they will show up as the borrower
of the setting as well. If they disregarded the warning there isn’t much chance that they will realize they are also a borrower of the workset.

Editing Request Granted
This is quite subtle but if you place an editing request but close your local file before it is granted you can end up the borrower of the elements you requested even though you effectively said, "never mind". The reason for this is that SwC resolves editing requests quietly. If you made a request and the person that was borrowing the element just uses SwC Revit simply passes possession of the element on to you, regardless of the fact that you aren’t working on the project at the moment. If you open your local file again you’ll find your username listed against the element(s). This situation can be avoided if the person you generated the edit request for denies your request before using SwC. They’d have to know to do that though which means more often than not you can quietly end up a borrower.

Workflow Quirks
There are a number of very normal activities that can result in warnings or odd behavior that are directly related to using worksets.

Duplicate Mark Values
This is a tragic problem based on the fact that the “next number” my local files thinks applies to an element is the same “next number” that your local file thinks is appropriate for the same element. The typical example (but true for all elements that Revit automatically numbers) is doors. If I place a door Revit thinks the next number is based on whatever value was used last. Let’s say that last door number used was 100, the next is 101. That’s true for my local and yours. If we both add doors we end up creating two doors that both use number 101, stored in the Mark parameter. When we use SwC Revit displays the warning (see Figure 34).

![Figure 34 - Duplicate mark value error appears when we use SwC.](image)

This can be avoided is we agree to be careful to use unique numbers. For example we could be careful to place doors on different floors and therefore we won’t compete for the same range of numbers. If the numbering of the elements is somewhat irrelevant, in other words we don’t usually keep track of unique Mark values, then we can just be careful to put a unique prefix in the value. For example, if we don’t keep track of Mark values for Air Terminals we could agree
that we put our initials in front of the number. The first air terminal I place I’d take care to renumber it to use sks-100. This would cause Revit to number the rest sks-101, sks-102 and so on. If we all behave this way we won’t cause Revit to generate the error in the first place. We can also avoid having to deal with them as part of the Review Warnings dialog later.

It would be better if Revit offered us the option to auto-number certain categories, only those that we worry about as a firm. Many of these warnings do resolve themselves (for example door numbers as they get associated with rooms properly) as the design progresses but it is still tedious to have to encounter them, review, consider and/or resolve them as we go.

**Filters Rename**

This is a quirky condition that is more a bug than our fault. If users create a new filter while filters are getting applied by others a situation can occur that will cause Revit to rename an existing filter to avoid a naming conflict. The only way to completely avoid the situation is to only create and alter Filters when no others users are actively working in local files for the same project file. David Baldachinno wrote about it in January 2012 on his blog “Do U Revit I Do”.

**Element Name Changed**

This is similar in concept to the duplicate mark value issue. For example, when multiple people add the same family to the project Revit sees the same family during SwC. To protect the existing family in the project as well as reconcile this it will rename the additional versions of the same family with an extra number increment, you’ll something like Double-Flush, Double-Flush1 and so on (see Figure 35).

![Figure 35 - Names changed to protect the families involved, duplicate Type Mark are still a problem.](image)

You’ll also notice in Figure 35 that there are six warnings total. Five of them are warning us that there are doors that are using the same Type Mark values, each of the five types in the same two door families that were loaded in separate user’s local files.

This is an indication that we are not communicating when we add new fundamental elements to a project. We do not all have to add that new door type, or wall type. Only one of us needs to do that and then use SwC to make it available to the rest of the team. The trouble is that new families are not usually just loaded without following that by placing them. We can very quickly
end up with a lot of elements that are really meant to be the same thing but aren’t as far as the Revit database is concerned. Don’t disregard this message for long!

**Stair Sketches Can’t Complete**

When working in a Central or Local file you will find that you cannot finish a stair sketch if you sketch Risers first then Boundaries. This assumes you used the Riser and Boundary tools instead of the Run tool.

It’s strange but it makes no difference in a stand-alone project but seems to matter a great deal in a central/local file. The workaround at this time is to sketch your stair Boundary completely first then add the Risers. Here is the familiar error message dialog that you’ll get if you sketch poorly (see Figure 36). You can see from the warning dialog it is from Revit Building 9.1, a very long time ago.

![Figure 36 - Did you sketch poorly or is it just the workset bug?](image)

Sketched poorly perhaps in some cases except that I have and several students have encountered this message while learning stair features and using worksets at the same time. At first we attributed the errors to basic sketch mistakes. Then we started to think maybe upgraded templates might fail to produce the stair. In the face of repeated tries as well as in brand new stock templates it became obvious that it was deeper than simple sketch mistakes or upgrade issues.

Revit support isolated the behavior to the order described above because they are accustomed to Boundaries first, Risers second and this process didn’t generate the error. I’m not sure when this behavior crept into being but I know that it is present in Revit Building 8.1, 9.0 and 9.1, just
tested it in those versions, so it has been with us for quite some time and still with us in 2013 (only by sketch). The image depicts a "U" stair but this issue will appear in any stair shape other than a single straight run, those will finish without error messages.

Another way to deal with this is to create the stair in a stand-alone project and then either copy/paste the finished working sketch from there to the a stair in sketch mode within the actual project or just copy/paste the entire stair. You do need to make sure your level to level relationships are the same.

Yet another way to deal with it is to sketch the stair wrong by extending boundary lines without breaking them at changes in elevations (slope vs. flat) like normally required. The stair boundary (stringers) and railings will fail spectacularly but the sketch will finish. You can then edit the sketch and split the boundary sketches where they are supposed to “break” and finish the sketch and it will WORK! It is very quirky!!

**Purge Unused**

If you create new elements in your local file while someone else uses Purge Unused in their local and they STC (Save to Central) first, guess what happens to your stuff. Your work goes away! That's what!

While this might be logical and consistent with the existing relationship of local files and central file, it however seems to me that Purge Unused ought to be a bit “smarter” or disabled, except under special circumstances like working in a central with no other users or no other uses while in a local perhaps.

Real World Example: Today, during training, an adventurous Revit user thought he'd use Purge Unused. This was during our day long excursion into the workset environment with fourteen concurrent users. When I gave the gang the task of sketching some interior partitions on “their” floors we were a bit surprised, after working for a little while, to find only three walls in the project. I casually commented that this situation could only happen if someone chose to delete them or use Purge Unused. At this point one of our gang volunteered, "Oh? I used Purge Unused!". Much laughter and conversation ensued. On a real project much “beating” might have ensued?

**Deleting Worksets**

Ron Palma with Ideate shared a rather unpleasant customer story the other day via their Solutions Blog. The circumstances seem to me a perfect storm of things that wouldn't normally happen.

- The customer's project is using Worksets
- The customer reassigned the project and survey point elements to a different Workset (normally assigned to the default Workset1)
- The customer deleted the Workset they were assigned to
Revit permits these elements to be deleted, when otherwise it would be impossible to do so.

What allowed for them to be deleted? They chose the “Deleted” option in this dialog (see Figure 37).

![Figure 37 - Don't choose the option Deleted!](image)

It is far better to choose the Moved to option and select the appropriate Workset.

One other habit I have is to never rename Workset 1 because Revit won't let us delete it. It's the "original" workset and Revit just doesn't let us delete it. The survey/project base point elements are assigned to it when Worksets are enabled. Left alone the opportunity to delete them wouldn't have been possible. It's better to leave it there as the place to put stray elements.

*I think Revit needs to be a bit more curious when deleting a workset. It should scan the element types assigned to the workset for any that are technically not supposed to be deleted.*

**MAINTENANCE**

**File Size**

While I personally feel that file size is often incorrectly used as a measure of a project’s condition it is one measure we can use to help judge where our project stands in terms of health or condition. If performance is suffering for an entire team and other things like modeling habits, warnings, number of linked files, number of imported DWG/DGN to name a few are not more likely culprits then the file size itself might be an indication that the project file needs to be purged, audited, compacted and or have a new central file created. Some people set a maximum file size such as 150 MB or 200 MB as the point where they will begin considering splitting a model into separate project files. I prefer not to be so focused on file size for that. I’d rather have far greater integration between elements and have a larger file size than increase
the number of poorly integrated elements and aspects of working between linked files. We can usually use workset to our advantage to mitigate performance issues based solely on file size as a measurement. Just becoming consistent with using the Specify option for loading worksets can make a big difference day to day.

Audit

This function is available for any Revit file and it operates without any feedback loop. In other words we can choose to use Audit (see Figure 38) but Revit won’t tell us anything about what happened during the audit.

![Audit option on the Open Dialog](image)

If a project file’s performance is noticeably different than previous days and the network itself isn’t to blame then you can consider opening the project using Audit to see if it results in a noticeable improvement. That’s the only way to tell if it did anything unfortunately, assuming we can actually detect improvement.

Compact Central Model

This option is only offered to us when we use Synchronize and Modify Settings (see Figure 39).

![Compact Central Model option available when using SwC.](image)

This concept is similar, at least metaphorically, to defragmenting a hard drive. Revit works to clean up the database instead of leaving things as they are. I think of it as my desk after a particularly hard day of work. I can have a lot of different things spread across my desk. Using compact is much like me putting everything back in its place before leaving for the day. The
term “compact” will be familiar to anyone who has used database software like Microsoft Access and since Revit is, at its core, a database driven application, using that term is consistent.

**Working in the Central File**

Why would you do that? Usually the rationale for working directly in the central file can be defused by simply saying that anything we can do in the central can be done in a local. It works except when it isn’t true. There are a few buggy situations that are best fixed by working in the central file.

**Managing Linked Files**

The reason mentioned earlier for using worksets to manage linked files is the same reason for dealing with this in the central file. If someone does unload a linked file in their local file the only certain way to reset the “game” we play trying to get it loaded for everyone is to use the Reload option while working in the central file.

**Fixing Filters**

I mentioned filters earlier and the only way to ensure that you can fix their naming is when nobody else is working on the project in a local file. You can fix them in a local but you can do it in the central file too. Either way you have to be alone to get it done.

**Create a New Central File**

If you use DfC or open the central file to create a new central file you are technically working in the central file, just in an abstract nearly risk-free way.

**Compact and Audit**

We can do this from a local file but technically the action is getting applied to your local file. This means to actually do both and have the result directly apply to the central file you need to do it to the central file.

**Security Guard**

I used to recommend that this be done in the central file but in this case it can easily be done in a local file too. That means this really isn’t a good reason to bother to open the central file.

**Clean up after Others or Relinquish for Elvis**

As described earlier we occasionally need to cleanup elements that are assigned to the wrong worksets or relinquish elements that other users have not been careful to return. The only place we can continue to work and change our username casually is in the central file itself. If there are several users we have to deal with it makes more sense to wait until after hours. Once we are alone we can open the central file and fix the issues without worrying about others using SwC and forcing us to make a local file.
Taking the File Offline (or home)
If I need to prevent other people from doing anything to the project for some reason I can open the central file (while everyone else is out) and make all the worksets editable, in my username. This means that nobody can actually do anything to the project again until I use SwC to relinquish things. The exception to that claim is people can add new elements, not edit existing elements. You'll need to make sure everyone knows you are doing this to prevent them from feel safe enough to add new elements.

Obviously this is a bit counterproductive to worksharing but occasionally there are circumstances that suggest it ought to be done, like taking the file home for the evening to clean the file up instead of staying at the office for all hours.

Using Detach From Central (DfC)

Project Manager Review
This person may not necessarily be an active participant within Revit but may need to check things or create views so they can resolve design issues without having to ask someone else to create/plot them. A PM can open a central file choosing DfC and work freely without encumbering the project in any way. When they finish they can close without saving and they have in effect been working on a temporary file. No guilt and no un-relinquished worksets. If they want to save the file it becomes its own central file and poses no risk to the one the team is working on.

Plotting/Exporting
So you've got to plot an entire set on Wednesday and you need the team to stop cranking long enough to do this only they won't or can't. Tell them to put their pens down at noon, use SwC and then open the central file using DfC. Now they can get back to it, and you can plot till the cows come home. It's the same routine for exporting backgrounds for your consultants.

Archiving
Just got Schematic Design approval and want a record copy of your project? Open using DfC then save the file into the correct folder adding the suffix SD (or whatever you require) to the filename. All set, a free and clear central file for archival and the original project filename gets to live on for another phase. No relinking pain, at least not with the main project file. The archived file will need to reestablish the links if any of the linked files are new central files too.

Internal Rendering
So you like Mental Ray but wish the rest of the team would leave the project alone long enough to get some time to work on a rendering? Sounding like a broken record now...DfC and go to town. Same deal for exporting to other formats.
**What IF? Scenarios**
Your client and PM have been hanging out at the pub (and didn't take you) and came in today with some great new ideas to try out? Now you need to mess up the model but are loathe to do it live because you're pretty sure they should've sobered up first? DfC to the rescue

**Troubleshooting/Interoffice linking**
So you've got some warnings you can't quite resolve and you need to let the home office take a look but the last time you did this they accidently borrowed a workset from your central file 1800 miles away, oops... DfC and send the file! Now they can study the file and give you some ideas to try in your file.

**Remove Worksets**
When you choose to use Detach from Central you'll get two choices regarding how Revit should interpret this request; Retain or Discard.

**Detach and Retain**
If you use Detach for any of the reasons listed previously you'll want to choose this option because the worksets will be kept around and you can continue to use them to your advantage.

**Detach and Remove**
You'll want to use this if you want to reset the project back to a state prior to using any worksets. This wasn't possible in the past but enough people request the ability to remove worksets that it was added as part of the DfC feature.

**Archival**
We have to create an archive of projects at various project milestones or just as a precaution every now and then. The Detach from Central (DFC) feature makes it easy to accomplish this but we do have to consider how the resulting files should behave when we interact with them later.

**Files Only**
If we are not concerned about how linked files are dealt with after using DFC then we can simply use DFC and save the file in an archive folder. This approach does not provide any reassurance that this file will find any linked files that have also been placed in this same folder after using DFC on them too. This is because the path that the file is remembering may not be assigned to relative versus absolute.

**Working Files**
If we intend to be able to open an archived project and be connected to all related linked files (archived versions of them as well) then we have to take the extra step of opening the archived files to make sure that all linked files are mapped correctly to their archived location. This can technically be resolved later if the files are needed. Doing so during the archival process can
alleviate any confusion that can arise from opening the file later and finding that an absolute linked path is showing newer versions of a linked file in the archived version of the host.

**Recovery**

Occasionally we find ourselves confronted with a situation that means we need to recover a project from an earlier version or potentially start fresh from an older version of the project. This is where the Save History and Rollback features become useful.

**Save History**

This allows you to review the comprehensive log for each time a user used SwC for a given project file. This can be done at the central file or the local file. Which is appropriate just depends on what you are opening to learn.

**Rollback**

This feature permits you to pick a point in time, in the past and start the project over so to speak from the place you choose. This should not be done casually. If you must do this keep these things in mind.

- Get everyone out of the project.
- Archive the project so you’ll have the current version if necessary in the future.
- Then use the rollback feature to establish where you need to start over from.
- Create the new Central File - The new file that is created is technically a local file, you must use Save As and check the option to “Make this the new central file” in order to establish the new central file for everyone to use.
- The team must now create new local files from the new central file and abandon their previous local files.

**WORKING TOGETHER**

This discussion can easily wander away from worksets and worksharing alone. For the sake of focus we’ll stick with issues specifically related to the subject of this session and leave the broader direction to other sessions.

**Small versus Large**

What do I mean by small and large? The issues we face with a modest sized project and team are somewhat different from a bigger project team. That probably isn’t really shocking. The bigger things get the more complex they need to be, it’s a fairly natural outcome. Worksets don’t need to be as numerous for smaller teams and projects. In fact some teams can live with the stock worksets, putting everything on Workset 1. That’s not to say it is best to do so because even a small team can benefit with a bit more expansion of their use of worksets. Large teams on the other hand will almost certain suffer with a simplistic approach to applying worksets. It may not become apparent until later but eventually the project performance will suffer and with it
the team’s morale. Just keep this in mind as you approach each project and use a plan that will accommodate its needs.

Views

With large projects and teams you’ll end up with many more views of the model than you would with a smaller project. This is natural too because each person will have their own needs, it follows then that the more people there are the more views there will be. You’ll need to manage this increase in views because the more view there are the more impact on performance you can see. In particular are views that show all or a great deal of the project as well as schedules that report information on many elements, like a door schedule, walls or rooms. The more reports we generate the more things there are for Revit to update and keep track of. We use views to see the model, to edit it as well as to document it formally. They are necessary but the larger the project is the more careful we will want to be about how many unnecessary views there are.

Worksharing Display

This is a fairly recent development in Revit and it can be quite useful to get a sense of where things stand and what is going on around us. We can turn on this feature here (see Figure 40).

[Image: Worksharing Display off, click to turn it on]

Checkout Status

Why get started on something if two other people have already checked out several elements that you are interested in? We might as well find a better place to work.

Owners

This is also quite useful before getting started on a task. Activate this option to see what other users are currently owners of elements in the same area you are interested in. If you see several other users nearby or even in the same spot then you should probably have a conversation with them before getting serious about doing anything. Ideally there won’t be anyone else in the area.

Model Updates

I find it particularly useful to use this to see if anything nearby what I want to work on is actively being changed.
Worksets

This tool is particularly useful to see how well we are remembering to set the Active Workset as a team. If the colors we see don’t make any sense then somebody will need to take the lead and get things fixed before they get out of hand.

The appearance of each of the modes is defined via Worksharing Display Settings (see Figure 41).

![Figure 41 - The Worksharing Display Settings dialog.](image)

Selective Opening

This feature is harder than it might seem to get people use. We are used to opening a whole file (in AutoCAD) and working freely on everything we can see without worrying about sharing or bumping into anyone. When we open a project we can decide how much of the project we really need to load. This can be quite effective at improving our day to day experience with a large project, if we can adjust our way of thinking about what we need to see right now. For this to work really well, our approach to creating worksets and how well we apply elements to them will have to be well thought out.

The best approach to helping people remember to be selective about which worksets they load is to set the central file to require the Specify setting in the first place (see Figure 42).
Then when we open a project we will be presented with the choice at the outset. If we fail to do that then as an individual user I can always invoke the option myself by using the Specify option before clicking Open (see Figure 43).

**Editing Requests**

These can be both a blessing and curse. Used well they can be quite helpful. If you use them indiscriminately you’ll just end up annoying the rest of the team. My habit is to avoid using Editing Requests in favor of talking with the person that has the elements I need.

**Worksharing Monitor**

This is a separate application that permits us to see how other users are interacting with the project.

**History**

As mentioned in another section this is particularly useful for troubleshooting and project evaluation.
Red Users
As a general rule we are not supposed to work in a central file. This tool will display a user in
Red if they are working in the central file. This just makes it easier to avoid doing something that
will result in a potential conflict later. Just don’t be red (refer to Figure 16)!

Don’t Pile On
When we see another user using SwC we can choose to delay our own SwC. If necessary we
can use instant messaging to help organize the queue.

LARGER TEAMS
The bigger the team is the quicker and easier it becomes for something that might ordinarily
seem inconsequential at first to become a real problem.

Eye Tee
Revit taxes a computer network like few other applications. A network that is very effective at
serving up regular office documents and web pages will quickly bend under Revit if it is not
optimized well. This means the IT staff needs to understand and appreciate this difference. The
common response, “Well everything else is working fine, it must be Revit” suggests that they
really don’t appreciate the significantly different load that Revit places on the infrastructure they
are responsible for. This means they need to revisit most of their assumptions about what is
sufficient, including switches, wire, routers, internet access, WAN and LAN relationships,
Domain structure, shared resources and more. A classic example of failing to do this is sharing
significant resource demanding applications with Revit projects like VOIP or Exchange Services.

Data Transfer
When your firm has more than one office it is more difficult, with Revit, to share work between
teams that are spread across them than you are used to with say AutoCAD. Because Revit is
not file centric like AutoCAD the network needs to be able to serve the project file to users in a
very different manner than the other cad applications. Hardware solutions like Riverbed’s
appliances help to pass data back and forth more efficiently while being careful to organize how
routers and switches connect teams together will all have a role in ensuring our project
experience is as good as it can be.

Structure
While outside the scope of this class the overall planning and relationships of your network
(LAN/WAN) as well as interoffice connections can come into question if you are experiencing
issues. A bad router or poorly terminated (untested) network cables run above a ceiling can
cripple a team. Sometimes fixing a problem is much more difficult than blaming poor
performance on Revit. Finding a solution means being willing to reconsider what we believe
about how well our network infrastructure was conceived and implemented.
Servers
It might not be obvious but using dedicated servers for Revit projects can often improve the performance of your projects because there are no other competing applications demanding attention from the server and its operating system. It should be obvious that you don’t want to put other demanding applications on the same server, such as VOIP, Exchange Server, or other significant data serving applications.

Sharing Files
We have to contend with sharing our work, not just with worksets, but with each other through making our files available to other parts of the project team. It might be a bit easier logistically for a firm that represents most of the design team (AEC) but even they struggle with cultural issues related to sharing files.

Where
Ideally the files are located on the same server and we all just link to the files as easily. Ideal is rarely how it works though. More often than not files must be sent back and forth via FTP, Buzzsaw, Dropbox, or some other manner. Try very hard to define where files will be located as early as possible and then don’t move them or second guess your decision later so you can avoid the hassle of re-establishing new central files just because you don’t like a folder name or which folder it is under.

Naming
This is just like deciding where the files should go. Decide early on a naming scheme that supports everyone so that file names don’t play the “my name has changed” game for months on end. I’m not going to tell you what naming to use, you won’t listen anyway, or your project will be more complicated or simpler than whatever I suggest. Pick a scheme deliberately and stick to it, document it and share the plan. Be sure to consider how to describe what project it is, who the file comes from, what scope or phase is involved, what sector or building and what discipline the model is for as well as some unique identifier to set it apart from similar models.

How Often
In the truest sense of BIM we ought to be connected to live models as we work. This isn’t always feasible or sometimes not even appropriate. Each team needs to decide how often we need to share files to best serve the project.

File Prep
Once you’ve decided how often the project teams will share their files you are ready to address what steps will be taken to prepare files for sharing. As soon as a project file is sent to another firm the network location changes and Revit regards their copy of the file as a local file that is looking for the original central file, which is the same file they now have, except it is a copy. It is recommended that we link to central files because linking to a local file can potentially result in a
delay as the file is loaded. Apparently Revit attempts to find the central file in that circumstance. There is no point in allowing that to happen since it can’t find the original central file location in this new network (office). This means that the when we receive a file from a consultant we need to use Detach from Central or Save As to establish it as a central file within our own network.

To Purge or Not
Some firms swear by purging the files they receive from consultants before linking them into their projects. The logic is that it reduces the time it takes to load the linked files because the file is smaller with fewer elements loaded. I suspect that this practice is much more defensible or necessary the larger project files become. For average projects though I’ve not really noticed a significant difference either way.

To Strip Views or Not
If we don’t really need to see the sheets the consultant has created or any of the detailing work they’ve done, not to mention schedules, then it isn’t unreasonable to remove all the extraneous views their model(s) contains. If purging helps reduce file size and speeds up the loading of links then surely this will help. Again with average size projects the difference hasn’t been that significant in my observations.

Workset Planning
A project team should discuss how they intend to use worksets, not just in an office but between offices, between consulting firms. If we agree on the use of specific worksets in advance we’ll be better prepared to manage the visibility of their elements when we begin sharing files. For example we often struggle to manage the structural engineer’s grids and our own. The same is true for levels between all parties. We all need our own levels to function within our own versions of Revit. If I am using Revit MEP then I’ve got to hide the architect’s levels as well as those of the structural engineer. If we agree in advance to assign our levels to a specific workset then we can set up the same worksets in our own project files. Revit has an interesting behavior when it encounters a linked file that has a workset that shares a name with one of its own worksets. It respects its own visibility setting and forces the linked file’s workset to behave according to the local version of it.

For example we can all create worksets named like this; Levels - M, Levels - S, and Levels - A. In the MEP model I can uncheck the Visible in all Views option for Levels - A and Levels - S (see Figure 44).
This will cause them to be hidden in all views of the project very easily. If we need to see them for coordination review then we can turn those worksets on with Visibility/Graphics as an override rather than force them on everywhere. This does require each team to be consistent about applying their elements to the correct worksets. This is one reason why Filters can be more successful but even they are prone to inconsistent application of family names and other parameters used as criteria. They are only foolproof if we can focus on their category as the criteria.

The goal of planning is to set expectations and to help each team take advantage of Revit’s strengths, work around weaknesses and most importantly collaborate well to the advantage of the project and team.

Cloud

Autodesk 360, Drop box, Google Drive, Box.net, YouSendIt and others are all forms of the latest buzzword or concept, the Cloud. The goal is to make it easier to share information with each other, make it all more accessible. Another aspect of the cloud is what Autodesk is hoping to do, shift heavy lifting calculations to the cloud, like analysis and rendering. Then there is the notion of our accessing the applications themselves, like Revit, in the cloud instead of installing it on our own PC.

Private

The company BIM9 for example offers services based on their expertise at helping a company establish its own cloud service for projects with their own internal technology and resources. This is attractive whether you hire them to do it or not. You have direct control over the quality and methodology for sharing data with your own offices as well as other firms that might be collaborating with you on your projects.
Offsite
The company Advance 2000 offers a different solution to the literal private concept. They host your data and applications in their data center and make the technological commitment to hardware and bandwidth your company is prepared to lease. This takes the burden of dealing with the technology side off your plate while providing the same kind of versatile choices for sharing data in exchange for a regular fee.

Co-Locate
Another approach to collaboration that many enjoy is to pull the parties together in a single location to do the bulk of the work together, in the same place. This could last for months, into construction or just last during a few phases or even just as a new phase of work gets started. Once the team is rolling they could return to their own locations because they’ve establish a protocol that is easy to follow though farther apart. Then again teams that do co-locate often prefer to stay that way for obvious reasons, they get more done because they are all together and things can happen quicker that way. In some cases the co-location occurs in conjunction with some sort of cloud solutions too. Think of it as choosing to focus on what will help the project team be the most productive. What might seem expensive at first might actually save a lot of time and money.

GLOSSARY OF WORKSHARING TERMS

Concept - Workset(s) - A feature that allows Revit to manage concurrent multi-user access to a project's data, designed to prevent anyone from editing the same elements at the same time. It provides elemental control and access versus traditional file level access and control of other design applications.

Workset Types:

User-Created Workset
These are for all model elements, the virtual representation of real building elements in a Revit project model. We associate elements with individual worksets according to a common idea or theme in order to organize and share our project data. In the context of a library these are the shelves we use in order to group books together logically, according to their subject matter. We are responsible for creating and managing them.

Family Workset
These are Revit created and managed worksets for each family defined in a project. Not for each individual door but for each kind of door (each family) loaded in the project.

Project Standard Workset
These are Revit created and managed worksets for each project standard feature such as dimensions, line styles, fill patterns etc.
View Workset
As we create new views Revit creates and then manages a workset for each view that is part of a project.

Our own workset Interaction is primarily limited to User Created Worksets. Family, Project Standard and View worksets are managed entirely by Revit. Users need only decide which User Created Workset to establish in the project and assign model elements to them. The Family, Project Standard and View worksets are only engaged when the user alters a property of an element that belongs to one of them. For example creating a new text type or changing a dimension types tick mark setting. When this occurs Revit lends that Workset definition to the user and the User returns it when they use Synchronize with Central combined with relinquishing the relevant workset(s), which Revit will attempt to ensure we do by automatically checking the appropriate workset option.

File Features/Actions:

Central File
This is a Revit file that has had worksets enabled.

Local File
This is a file that is copy of a Central File that a user works in. Any time a central file is renamed, copied or moved to a new location Revit will regard it as a local file. It will no longer treat it as a central file.

Synchronize with Central (SWC)
This is a two-way transaction, the action of saving work (pushing) completed in a local file to the central file and loading (pulling) work completed by others via SWC. Your changes are now available for others to see.

Save Local
This is saving work within your local file, using Application menu > Save or the Save icon on the Quick Access Toolbar (3.5” floppy diskette icon). Changes you make are not visible to others until you use Synchronize with Central.

Reload Latest (RL)
This is a one-way transaction, pulling changes from the central file to your local file. Revit will display any changes that others have submitted using Synchronize with Central.
Ownership/Borrowing

**Owner**
This status occurs when we possess an entire Workset(s) and all the elements that are assigned to it. This will prevent anyone else from making changes to it or any elements assigned to that workset.

**Borrower**
This status occurs when we possess an individual element(s) that is/are assigned to a Workset, not the Workset itself.

**Editing Request**
A user can request to borrow elements from others. The person we request permission from can Grant or Deny our request. Granting a request can only be done if no changes have been made to the element. SWC is required if changes are to be kept.

**Make Elements Editable**
This is a command and it deals with individual elements, allow us to borrow just the selected element(s), like "borrowing a book".

**Make Workset Editable**
This is a command and it deals with the entire Workset (like a bookshelf) that an element is assigned to as well as the element(s), like borrowing a bookshelf and all of its books.

**Relinquish**
We do this to return the element(s) we have borrowed so that others may borrow them.

**Editable**
The element or workset belongs to you. You are free to make changes to them.

**Not Editable**
The element or workset does not belong to you. You can’t change it/them unless you borrow it/them.

**Viewing/Interaction:**

**Active Workset**
This is the workset that is displayed in the workset panel on the ribbon. New elements will become part of this workset automatically. Note this list does not display the workset name of a selected object nor does it change the workset of a selected object if you choose another while objects are selected.
Open
The workset is loaded into memory and Revit will display it in all views according to visibility/graphic settings of each view.

Closed
The workset is not loaded and Revit will not display it or process it in memory and any views regardless of visibility/graphic settings of each view.

Gray Inactive Worksets
This is a user specific setting that will alter a view to make inactive worksets gray, a lighter color, than the Active workset elements.

Maintenance/Special Features:

Compact Central File
This will condense the Revit database and result in a smaller file size. This is similar to the concept of defragmenting your computer’s hard drive.

Detach From Central
This allows you to open a central file (or local file) as a separate file and permanently sever its relationship to the original. It can never be synchronized with the original central file. Initially when you choose this option the file is a temporary file without a name. It is only when you choose Save that you get the opportunity to keep the file in its separate state.

Examples where this is useful are: Project Manager review or exploration, Plotting, Exporting, Troubleshooting and archiving. In each case this allows unfettered access to the project and does no harm to the active project. It also allows the project team to continue working while “snapshot” tasks like plotting, exporting or archiving are done.

IF THERE IS TIME

Revit Server
This feature is really outside the scope of this session, or more than I can practically demonstrate in this setting. I’ll just delve into concepts briefly.

Eye Tee
You aren’t going to get this up and running without working closely with the IT team in your office. It’s a much more demanding process than just installing Revit and changing a few settings.

Host
This is the master server for project files, you place projects on this host and it will be available to any part of your Domain.
Accelerator
This is very much like the central file you already connect to in your office now. Accelerators carry on a dialog with a Host, a middle man of sorts, and they resolve the ownership and element status as we work. The key is to ensure that no elements are being altered by more than one person at one time.

User
Your local file is essential the same as it ever was except to begin working you connect to via the Revit Server icon that appears as a shortcut on the left when you choose to open a Project (see Figure 45).

Locks
The most significant issue you’ll encounter with Revit Server is resolving locks that occur when something gets out of sync and Revit applies a lock to protect an element or elements. The Revit Server admin panel provides the control we need to deal with these.

Q&A