AUTODESK. UNIVERSITY

CES224424

How to Build Effective Assemblies

Russ Nicloy
MACER Technologies, Inc.

Learning Objectives

- Understand why connecting subassemblies is crucial.
- Know how to reuse design options to reduce duplicate work
- Use tracking techniques to aid in navigating the design later in the project.
- Understand the strengths and weaknesses of Conditional Subassemblies.

Description

Assemblies figure prominently in the building of Civil 3D corridors. They provide the basic information that can be used in many ways as your design takes shape. There is the potential for a lot of data, and since wrangling that data later could get confusing, having a strategy to manage the building blocks of corridors can make the long-term maintenance pay off. Save time and effort by using processes that standardize your corridor assemblies.

Speaker

Russ Nicloy has been in the civil/infrastructure industry for over 23 years. For 10 years, he was in production drafting and GIS roles for gas and water utilities and site design, and for the last 13, he worked for an Autodesk reseller, providing subject-matter expertise in Civil 3D, Infraworks, Map 3D, Land Desktop, and AutoCAD. He has worked with municipalities, utilities, site designers, environmental engineering firms, and the Wisconsin DOT on transportation projects, helping firms implement and train their staff for their specific use of their software. He also taught Civil 3D Basic and Civil 3D Advanced courses for several semesters at Milwaukee Area Technical College. His company, MACER Technologies, Inc., provides drafting services to the civil/infrastructure industry, in the Autodesk Infrastructure Collection of products.

Why Connecting Subassemblies is Critical

Assemblies are made up of a series of subassembly parts that are connected together. If these subassemblies are not properly connected data from the individual subassemblies will not be read by the assembly. This can cause various issues, all of them negative for your design. Here are a couple of important tips to use while working with subassemblies.

Connect using the Point Code symbols

When connecting a subassembly make sure to attach to the point code of the already connected subassembly precisely. If you select on some other part of the subassembly (link or shape code) the subassembly may not connect where you expect. Usually this is immediately obvious and easily fixed.

Don’t “place” or OSNAP

The subassembly attachment process is specialized to occur when you have select the point code to attach to. If you use some other method, such as AutoCAD OSNAPS or just moving something in a location that looks correct, the data flow will not be complete. It would be as if you had two electrical lines that you laid over one another. They might accidentally work for a little while, but it would not be a proper connection for the long term.

Don’t use the AutoCAD Move, Copy or Mirror

The subassemblies have a very specific Move, copy, or mirror command function that is NOT the same as the AutoCAD commands of the same names. Because of the way the subassemblies are programmed they need these specialized tools. As in the case of OSNAPs, the subassemblies may look like the AutoCAD tools worked, but the data would not flow and eventually it will cause problems.

Reuse Design, Reduce Work

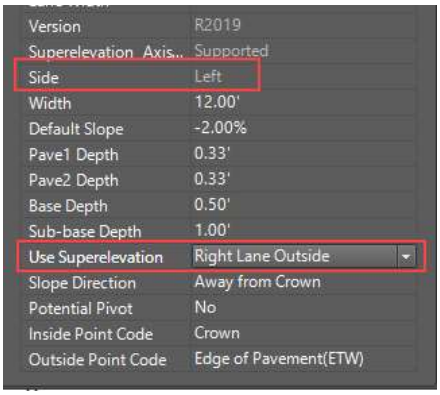
Many designs have similarities. Perhaps not always consistent, but there are enough similarities that reusing parts of a design will save time and typing.

Subassembly MOVE, COPY, OR MIRROR

It is important to remember the AutoCAD MOVE, COPY, and MIRROR commands do not help with subassemblies. There are specialized versions of these commands to help with subassemblies. They are similar, but under the hood they are carefully adjusting parameters, so the coding does not break. These versions can be accessed by selecting the subassembly part you need to act on, then from the green context ribbon select the appropriate command. It won’t feel very different, but the result will be better.

The Mirror Gotcha

Something to be mindful of is that the subassembly MIRROR command only changes the side of the new version of the subassembly. Other fields are not changed. For example; a lane has a side, but it also has fields for which superelevation column to use. If you mirror a right-sided lane that is following the Right Outside superelevation column, you will end up with a left-sided lane that is following the Right Outside superelevation column. So, go ahead and mirror, but give a once over on the parameters when you are done.



AN EXAMPLE OF THE MIRROR GOTCHA

Copying Assemblies

While the AutoCAD MOVE, COPY, and MIRROR commands do not help with subassemblies, but they do work on the assemblies. Sometimes there will be enough similarities between assemblies that copying the whole assembly and then tweaking or replacing the parts that aren't the same is much easier.

Even if you are using the exact same assembly in a different region of your corridor, you might want to copy the whole assembly, rename it, and assign it separately. First, if a small change occurs in one region, and you edit the assembly, if they are separate copies, you haven't accidentally propagated that change where you didn't intend. Second, name it with the region stations in the name. That helps you and others, remember where the assemblies are in use. It also helps when those edits do come around. It is also helpful when in the corridor if all of the parts have a similar name, so you know they are related.

Unique names

In Civil 3D some objects must have a unique name. When you copy an object, if a unique name is necessary, it will default to the source's name with a parenthetical number behind it (Example: (1)) behind it. Depending on how many copies there are this number gets larger. You should go in and change these names immediately after a copy. It is easily overlooked until later, and it may be found by someone who doesn't understand your methodology.

Techniques for Tracking Assemblies Through the Project

Depending on the size of the project you will probably be getting into some complicated design situations. Multiple regions call for multiple assemblies. Add the small transition assembly needs into this mix and you could have a pretty scattered list of corridor design elements to track down.

Naming Techniques

There is no “right” way to name your assemblies other than to do it. Your organization should agree upon a naming convention so that all of the varied issues can be addressed. Here are some suggestions. First, include the stationing where the assembly is assigned somewhere in the name. Second, have the name tell the users what it is. Whether you abbreviate these (Example: Sidewalk is SDWLK) or spell them out, this will help in targeting or other issues down the road. Third, try to keep the names short enough that they fit in dialog boxes. If you put the best information at the end no one will see them when their dialogs are too thin. This is tricky due to the perceived need for more specific data, but that’s where the abbreviations come in.

Name them Right Away

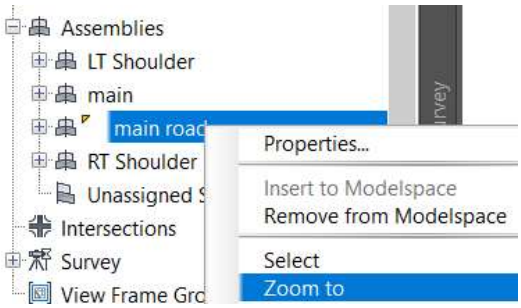
It’s worth doing the naming as soon as the assembly is created (or *while* you are creating it). The best of intentions is you will name it in a minute after the other three assemblies you are building. If you get distract and forget it is just that much more annoying looking through all the other ones to find the one you need. Don’t forget, this also helps others on your team, or your customer, decipher the design later.

Use the ‘Zoom To’ Options

Not all Civil 3D objects have a ‘Zoom To’ option. Assemblies are ones that do. Instead of hunting around the drawing for the assembly you want, go to the Toolspace > Prospector > Assemblies (its just below Corridors) and find the right name. Even if you are slightly off on the name, you are probably closer then if you pan through all of them.

Location of assemblies in drawing

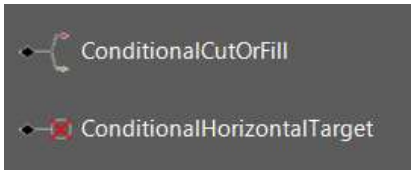
It can be helpful to stack the assemblies together by some common thread. Station location, or type of assembly are popular for good reason. If you do zoom to the wrong assembly, you are more likely to be close to the one you need anyway.



IN THE TOOL SPACE PROSPECTOR, EXPAND ASSEMBLIES, RIGHT-CLICK ON THE ASSEMBLY YOU ARE LOOKING FOR.

Conditional Subassemblies: Strengths and Weaknesses

Conditional subassemblies are triggers that allow the assembly to react to the conditions that the corridor is being created in. There are two types of conditional subassemblies, CutOrFill and HorizontalTargets. At each frequency line of a corridor the conditional subassembly will look for the “found” or “not found” conditions. If found, it will apply the rest of the subassemblies that are in that branch, and if not found, it will apply the subassemblies in that branch. This allows the conditional subassemblies to build multiple assemblies through one region where otherwise multiple regions and assembly designs would be needed.



Strengths

Conditional subassemblies accel at creating the conditions that one assembly can cover the design needs that would take several assemblies in regions to complete. Conditionals operate exclusively to a side, left or right, so the conditions that trigger responses can be unique to a side. Because of their “if/then” nature, very different design needs can be set up and executed throughout the course of a corridor design. A user can be very creative in how they lay out the triggers for the conditionals, so they can be useful in many non-standard design situations.

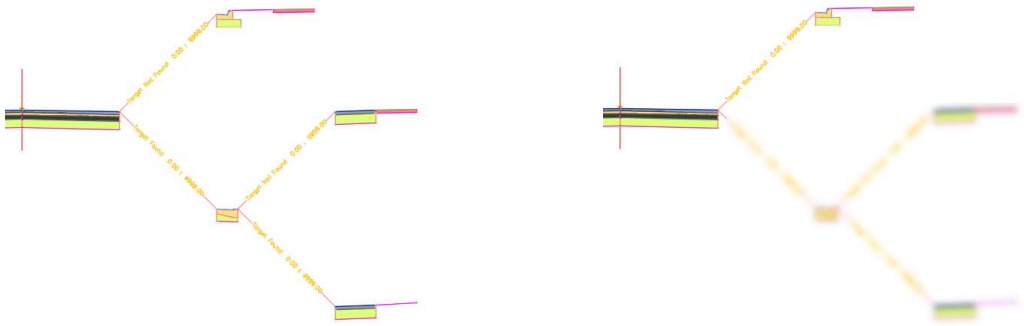
Weaknesses

The number one weakness of conditional subassemblies is their tendency to complicate the assembly used in the corridor. For every conditional assembly, a target must be assigned. If there is a not found condition along with a found condition you NEED to assign the not found condition to what it isn’t finding. Many users miss this important assignment. This is discussed more below.

Finding the unfound condition

As mentioned above, one of the strange complications is a user not targeting their not found condition. The result of this oversite is both the found and the not found condition will appear at the same time. This can be difficult to see where there are a lot of corridor lines in the area of need. Usually dropping into Section Editor for the corridor it is easier to identify where this has occurred.

The other complication is a user will try to “nest” conditional subassemblies inside one another. The issue here is that the first conditional is total, and if there are elements down the second conditional link that the user needs, that needs to be recreated for the both full paths. Many users are confused when something in their design works for several stations but then disappears because one of their conditional situations temporarily crosses the parameters they had set, wiping out a whole branch of subassemblies.



AN ASSEMBLY WITH “NESTED” CONDITIONAL SUBASSEMBLIES VS WHAT A “NOT FOUND” CONDITION SEES

Don’t forget, as great as conditional subassemblies can be, they aren’t “Superman.” Sometimes its better to give in to separate regions rather than overcomplicate a design.

Cut or Fill?

One thing to be aware of regarding the Conditional Cut Or Fill subassembly is that it is dealing in depth not distance of cut or fill. It measures from its attachment point and that is where the condition is assumed to be at. This isn’t good or bad, but you need to know what you are using.

Please feel free to contact me with questions regarding the processes found in this document.

Russ Nicloy
MACER Technologies, Inc.
Russ.nicloy@macertechnologies.com