CI500019

Underground Utility GIS Features to Civil 3D Pipe Networks with Dynamo

Jae Kwon
SolidCAD

Maxime Carrier
SolidCAD

Learning Objectives

- Import GIS object into Civil 3D
- Create Civil 3D pipes and structures from electric utility GIS objects using Dynamo for Civil 3D
- Transfer attributes between Civil 3D objects and GIS objects
- Create MAPEXPORT friendly objects ready to be consumed by GIS

Description

Local governments are gradually more involved with 3D infrastructure designs: on one hand, they are managing their existing infrastructure, like storm/sanitary sewers and aqueducts networks, through georeferenced databases, and the other hand designing their proposed infrastructures through Civil 3D objects.

Civil 3D never had a perfect fit to connect with GIS data, from recreating existing 3D infrastructures to better connect them with proposed designs. Until Dynamo came into the picture!

This presentation covers efficient workflows using Dynamo for Civil 3D, mapping electric utility GIS data to automatically create Civil 3D pipe network objects, using custom pipe catalogs. It also covers attribute transfers involving both object data and property sets.

Speaker(s)

Jae Kwon is a professional technologist in New Brunswick, with a focus on highway and municipal civil engineering technology. After joining Crandall Engineering, Jae took lead in a wide range of projects, including highway design, GIS implementation, Lumion rendering and CAD standard management. Notable past projects include Irving Oil rail terminal expansions, Cabot Trail highway realignment, Ritchie Lake trunk sewer in Quispamsis, Shediac/GSSC GIS database, and Perth/Andover Redevelopment Design. He has also developed a full suite of standards, templates, scripts, and a knowledge base system with for Crandall that greatly improved efficiency. Jae provides training and support on Autodesk civil engineering products

with a focus on AutoCAD and Civil 3D. He is a strong advocate for practices that emphasize project efficiency, elimination of error, strong organization, and responsive feedback.

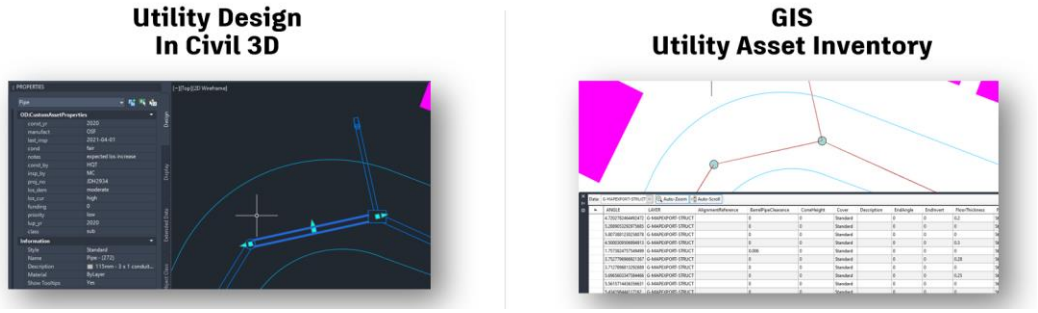
Maxime Carrier is a geomatics engineer, member of the "*Ordre des Ingénieurs du Québec*" and has worked since 2008 on a wide variety of projects in cartography and 3D modeling. Officially graduated from Laval University in 2013 and immediately hired as a technology integration specialist at Groupe VRSB inc., a survey firm, Maxime takes charge of innovative projects, such as the implementation of drones and LiDAR scanning in surveying methods, the use of aerial imagery at different scales in the planning of a natural gas distribution network and the "as-built" 3D modeling of sites upstream of BIM projects. A Cansel employee since June 2018, Maxime recently joined the SolidCAD team as a bilingual technical consultant in civil engineering and geomatics. He currently covers consulting, implementation, training and support services for civil engineering and geomatics.

Bridging the World of Civil 3D Design and GIS Asset Inventory

Introduction

The Problem

There is a gulf between the objects that Civil 3D designers work with and the objects that GIS technicians work with.



TYPICAL UTILITY OBJECTS IN CIVIL 3D AND GIS

Increasingly, Civil 3D designers must prepare their data in a form that is easily digestible by a GIS asset platform, as municipalities and corporations face more pressure to have a detailed asset management program for funding and subsidies.

GIS technicians, on the other hand, tend to receive request for information that is needed by designers to digitally reproduce existing assets, and often find that much of the needed information is lost.

Consequently, there is a need to accurately and comprehensively transfer data between the two worlds. However, the process to translate from one to the other is typically a manual, labor intensive process involving a lot of data entry, prone to human error. The laborious nature of the process places tremendous pressure on the technicians to make decisions of which data to keep - decisions that may not necessarily agree with their present or future colleagues.

Why Dynamo?

There are three types of solutions available to the problem.

At one end of the spectrum, are pre-made tools for the process. Some examples include the Export to SDF tool that can be installed as a Civil 3D add-on and the ArcGIS connector. These are easy to use, but rather inflexible when it comes to custom needs of an organization. The exported properties are few, may not be the ones needed, or may be exported in a way that it takes additional schema mapping work that adds to the labour.

At the other end of the spectrum is custom coding using one of the traditional languages such as Lisp, VBA, and .NET. This solution is the most powerful and customizable solution available, but it requires a significant time commitment to get started if one is

new to it. Few organizations have development teams available to take on the task in-house, and others find outside development a costly venture.

Somewhere in the middle is Dynamo, a visual scripting solution. It has a balance of flexibility and accessibility that many technicians find attractive. It is a custom solution with a great deal of power that is within immediate reach for most organizations.

AVAILABLE SOLUTIONS

Dynamo Packages Used

This project was only possible because of the packages that dedicated community leaders put together.

PACKAGES USED

The primary package that was used, of course, was the Civil 3D Toolkit by Paolo Serra and Safi Hage. This package contained the pipe network, GIS feature, OD table nodes and other nodes that formed most of the graphs we put together. We would like to thank Paolo especially for the timely updates to OD table nodes in response to our reports on Github.

For UI, we relied heavily on Data-Shapes by Mostafa. It allowed us to gather information about the drawing before presenting the user with UI choices; further, it allowed us to bypass the Dynamo player so that we could split the task into several smaller graphs and use them in various combinations via macros. Initially, we were simply looking for a way to present a drop-down. In the end it ended up restructuring our project altogether that gave us added flexibility.

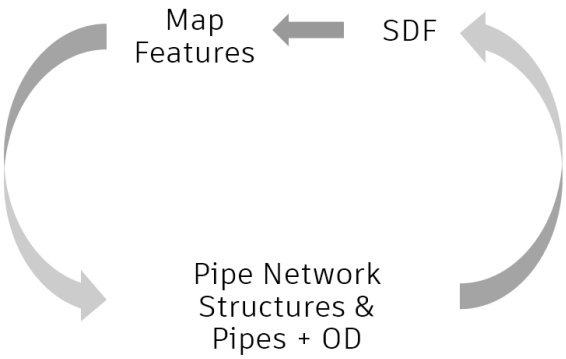
Finally, we used the Clockwork package by Andydandy to control the flow of data and measure the run times of various node groups to troubleshoot any performance issues.

Presentation Structure

The discussion of this topics is divided into two parts.

First, the GIS to Civil 3D data flow is presented. The task involves connecting to map features and creating pipe network objects based on the geometry and data tables of the map features. Further, additional asset data is transferred from the map feature data tables to the pipe network Object Table records.

Second, the Civil 3D to GIS data flow is presented. Information is gathered from pipe network objects, then AutoCAD objects are created with object data tables storing the additional pipe network object information. If there is any information stored as property sets, it is converted to object data. All the geometry and object data information are then exported to GIS-ready formats.



FLOW OF DATA

Downloadable Content

Samples of all graphs used in the presentation are offered here for free download:
https://files.solidcad.ca/index.php/s/KgTgHHpD4ZBgiZq

The following are the graphs available for download:

GIS to Civil 3D
pnet_fdo_to_c3d_DYNdata.dyn - Creates C3D pipe network objects from map feature layers
pnet_fdoData_to_c3dData.dyn - Transfers asset data from map features to C3D pipe network objects

Civil 3D to GIS
pnet_c3d_to_acad.dyn - Creates AutoCAD objects from C3D pipe network objects
pnet_ps_to_od_definition.dyn - Defines object data tables from property set definitions
pnet_ps_to_od_values.dyn - Transfers values from property sets to object data tables

pnet_transfer_od_to_cad.dyn - Transfers object data from pipe network objects to autocad objects

The above graphs will require minor modifications to function with your own drawings and catalogs (e.g. file paths, pipe network names, feature layer names).

While snippets of the graphs are included as screenshots in this document sometimes, they are only meant to help locate the relevant parts of the actual graph. Not much actual graph information will be shown in these screenshots.

The presentation is a high-level overview of the implementation that we show. This document aims to clarify some of the main issues for the task of converting between Civil 3D and GIS utility data, as well as some discussion on various details of the graphs.

This project has been very educational for us, and hopefully we can make the process easier for others who are considering jumping into a Dynamo solution to the data conversion task. We are sure that many of you have implemented similar Dynamo graphs to accomplish the task, and if you have any tips for improvement, we would be very grateful.

Transferring Data from GIS to Civil 3D

From GIS to Civil 3D

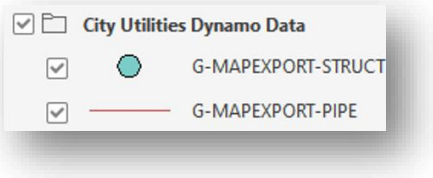
Typically, we pull information from GIS for Civil 3D design for context of existing conditions. Usually, it is not clear to the Civil 3D designer how the GIS information should be represented in the Civil 3D environment, as key information is often missing. But if the Civil 3D DWG and the GIS information is faithfully passed back and forth without loss, what used to be Civil 3D as-built objects stored in the GIS database can be easily reconstructed as pipe network objects.

Setup and Schema Considerations

Required Setup for the GIS-to-C3D Graphs (graph name)

A few preparations are needed to run this graph.

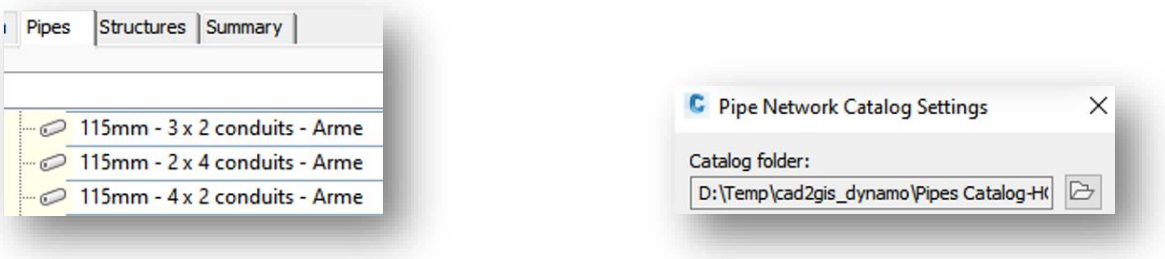
First, a connection needs to be made to the GIS data source and added to the current drawing as feature layers. The data source can be anything that Map 3D can handle; it may be a set of shape files, SDFs, WFS, and so on.



FEATURE LAYER CONNECTION IN MAP 3D

Second, a pipe network parts list needs to be created with all the families and part sizes required to recreate the objects that are in the GIS data source. Often, this will require a

custom part catalog to be loaded, since the parts used in utilities are typically different that what is available for gravity pipe networks in Civil 3D.



PARTS LIST AND MAP 3D CATALOG

Schema Considerations

One of the central considerations when converting Civil 3D and GIS data is the data schema. Civil 3D operators would know them as either object data definitions, property set definitions, or the list of built-in properties of pipes and structures. GIS technicians would know them as data headings, data types and their constraints.

We must decide what format we store the data in. When transferring data ready for GIS consumption, what property names and data types do we use? This will require a comparison of Civil 3D's object data capabilities and the constraints of the desired data format for importing into the GIS platform. Is the GIS database relatively new and allow for restructuring of the data? Or is it mature and everything needs to conform to the body of data that already exists?

In our sample implementation, we are dealing with a new GIS database with only a handful of existing data stored as SDFs, extracted by FME (Safe Software). The property names are uninformative and confusing; luckily, we are free to setup new property names and definitions. to keep things simple, we simply used the names that the Dynamo nodes use. Those property names will be preserved by exporting to either SQLite files, which seems to be the most compatible format in the current propriety and open source software.

A Dynamo graph was constructed to import the data from these existing files (pnet_fdo_to_c3d_FMEdata.dyn). Once the data is incorporated into the new schema, this conversion graph is no longer be needed.

```
num13 = Structure.RimElevation(t1);
num14 = Structure.RimToSumpHeight(t1);
num17 = Structure.Rotation(t1);
str8 = Structure.Shape(t1);
num10 = Structure.Station(t1);
num16 = Structure.SumpDepth(t1);
num20 = Structure.SumpElevation(t1);
```

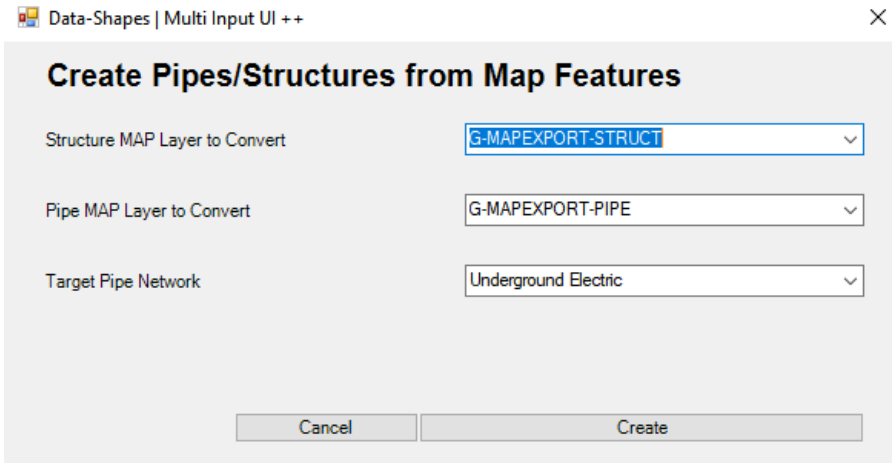
RimElevation	RimToSumpHeight	Rotation
160.11	1.75	4.7202782464492472
160.15284187406732	1.835	5.2889053292975685
160.81847603014248	1.835	5.8073881230258069
160.85307051445494	2.6	4.5000309506994913

aec_stru01	aec_stru02	aec_stru03	aec_stru04
163.491	<Null>	0	159.98031391888856
162.7703386986262	<Null>	0	160.11548429653047
162.11284370035472	<Null>	0	160.12420069677754
162	<Null>	0	160.47358307475224

FROM DYNAMO AND EXISTING SCHEMA TO NEW SCHEMA

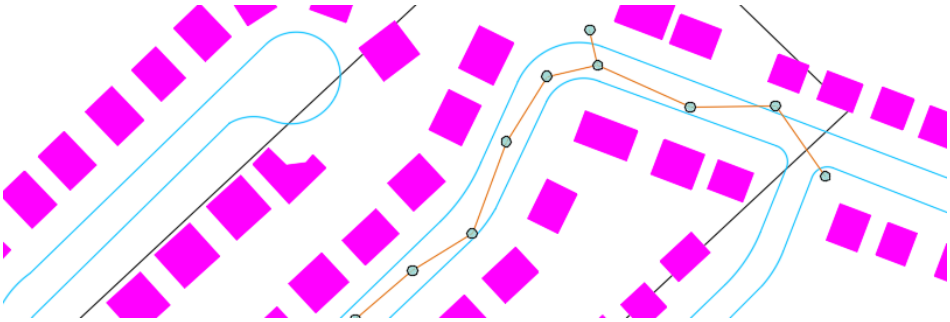
From GIS to Civil 3D

Step 1 - Convert Map Features to Civil 3D Pipe Network Objects



PNET_FDO_TO_C3D_DYNDATA.DYN

First step is to create the Civil 3D pipe network structures and pipes using the information from the map features. Only the core information is used to construct them; other data will be transferred over in a separate graph.



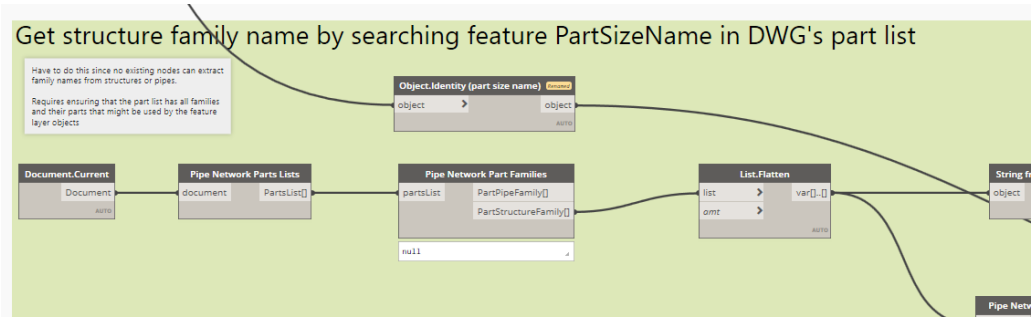
UNDERGROUND UTILITY MAP FEATURES

First, we get from the user three inputs: 1) the map feature layer that the structures are in, 2) the map feature layer that the pipes are in, and the target pipe network that pipe network objects will be created in. By leveraging Data Shapes UI, we can scan what map layers and pipe networks exist in the drawing and let the user make the selection with drop down menus. We have taken care to keep the structures and pipes on separate layers because eventually it will make the process a lot easier when the data is eventually ready to be exported back to GIS.

Next, we pull all the individual features from the feature layers. While we could have built in a location or text filter of some kind, we chose to leave any filtering for the map task pane, as Map 3D has an excellent filtering UI. Within the Dynamo script, we simply passed the feature layer through a text filter without any text input to get all features.

From those map features, key information is extracted. These include the geometries and spatial properties such as structure rotation, rim elevation and pipe start/end inverts. The extra spatial information is key to ensuring that structures and pipes are fitted together properly.

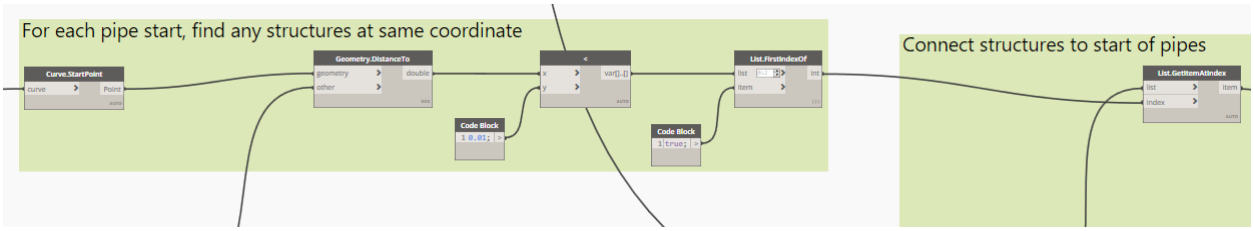
Perhaps the most important information is identifiers that tie the features to specific parts in the pipe catalog - The part size name and the family name. These names are required to find the correct part size object. But since family names cannot be extracted from pipes and structures, we simply gather all part size names for all families and do a search for what we need. This means that all needed part size names need to be loaded into the parts list.



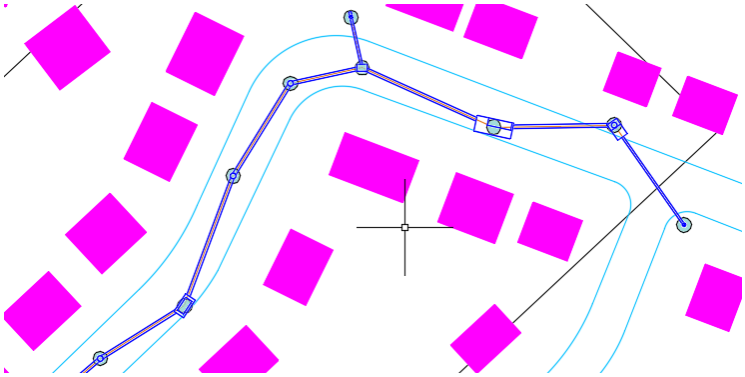
EXTRACTING THE CRITICAL FAMILY NAME AND PART SIZE NAME

Structures and pipes are then created using this information. For a mature GIS database, the part size name is most likely not stored in the GIS data, nor is there an option to add that to the database easily. A mechanism to identify the appropriate part size based on properties such as size and material might be needed in that case.

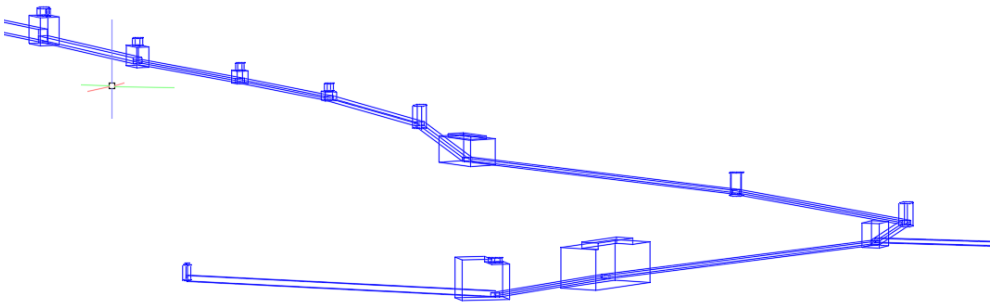
Finally, the coordinates of the structures are compared to the start/end coordinates of pipes; those that overlap are connected.



CONNECTING STRUCTURES TO PIPES



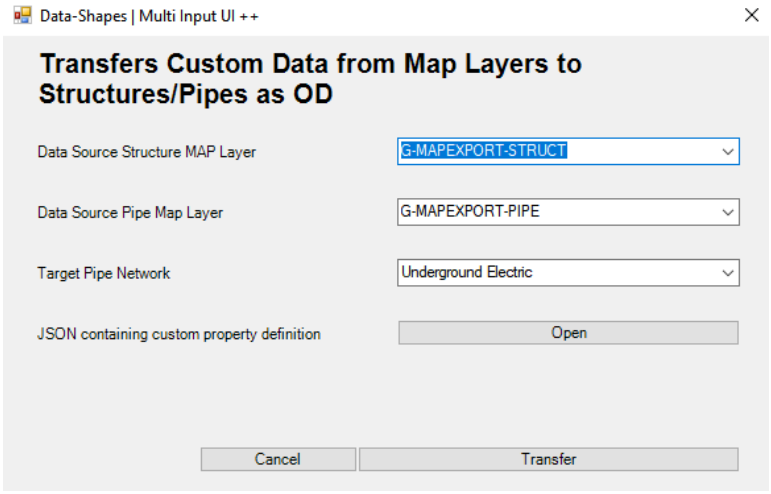
AFTER CONVERSION TO PIPE NETWORK PARTS, PLAN VIEW



AFTER CONVERSION TO PIPE NETWORK PARTS, 3D VIEW

Now that the pipes and structures have been created with the correct part definition, position, rotation, and elevation, we can transfer in other data, whether it's extra pipe/structure data or custom asset data about these objects.

Step 2 - Transferring Additional Data from Map Features to Civil 3D Pipe Network Objects



PNET_FDODATA_TO_C3DDATA.DYN

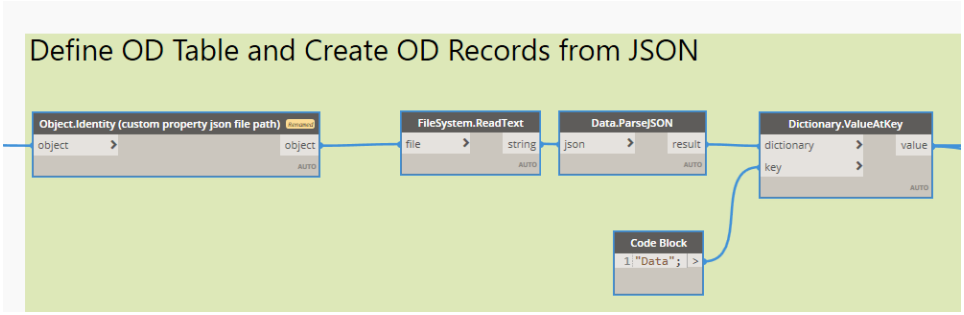
Let's look at the second step - that of transferring in extra data from the map features to the structures and pipes created by the last graph. Some of the extra data might include structure or pipe properties not reproduced by the correct selection of the part in the catalog, such as rim-to-sump height, sump depth and sump elevation. Other data include custom asset properties that will have to be attached as object data.

class	cond	const_by	const_yr	funding	insp_by	last_insp	los_cur	los_dem	lup_yr	manufact	notes	priority	proj_no	FeatId
sub	fair	HQT	2020	0	MC	2021-03-24	high	moderate	2020	OSF	expected los increase	low	JDH2934	1
sub	fair	HQT	2020	0	MC	2021-03-24	high	moderate	2020	OSF	expected los increase	low	JDH2934	2
sub	fair	HQT	2020	0	MC	2021-03-24	high	moderate	2020	OSF	expected los increase	low	JDH2934	3
sub	fair	HQT	2020	0	MC	2021-03-24	high	moderate	2020	OSF	expected los increase	low	JDH2934	4
sub	fair	HQT	2020	0	MC	2021-03-24	high	moderate	2020	OSF	expected los increase	low	JDH2934	5
sub	fair	HQT	2020	0	MC	2021-04-01	high	moderate	2020	OSF	expected los increase	low	JDH2934	6

ASSET DATA IN MAP FEATURE TABLE FOR CONVERSION TO OBJECT DATA

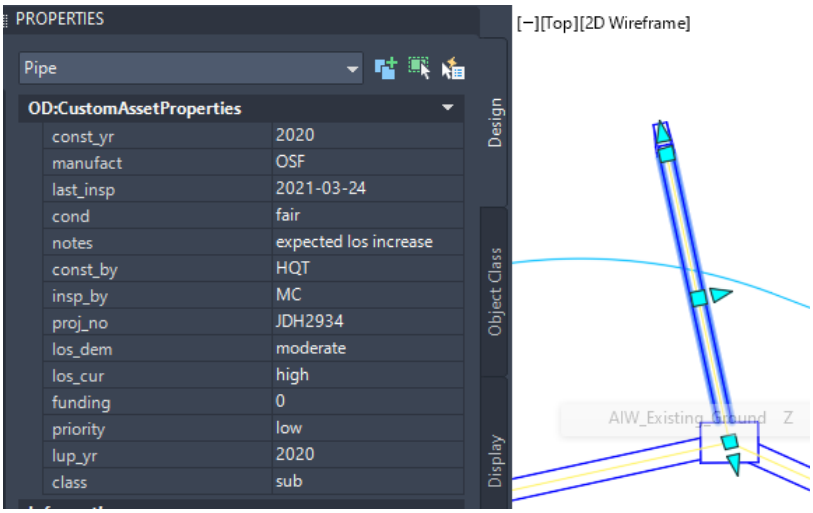
We ask the user for the same data as last time: 1) structure feature layer name, 2) pipe feature layer name, and 3) target pipe network name.

Next, object data tables need to be made that mirror the data structures of the map feature layers. The data structures have already been captured during converting data the other way (from GIS to Civil 3D) and saved as a JSON file. We can simply read this JSON file and unpack the property definitions. This is then used to construct the Object Data table.



UNPACKING PREVIOUSLY STORED SCHEMA FROM JSON TO OD

Once the Object Data tables are created, we attach the structures and pipes to these tables and transfer the values over from the map features.



ASSET DATA SUCCESSFULLY TRANSFERRED FROM MAP FEATURE DATA TABLE TO PIPE OBJECT DATA

Transfer attributes between Civil 3D objects and GIS objects

The journey back to GIS

Now, we need to transfer our final design or as-built Civil 3D .DWG plan back to our GIS database. The designer might have added and removed pieces in our network and updated the metadata on the remaining parts. To apply these alterations back to GIS, a GIS user or database administrator would run these automated actions :

- ***Read*** our proposed pipe network and any metadata attached to it.
- ***Keep*** intact in the database all untouched existing components
- ***Update*** all components that were identified as “modified”
- ***Remove*** all components that were marked “to be removed”
- ***Add*** all new components to our database

Currently, Dynamo does not handle these actions back into a GIS database. That’s because Dynamo, while it can totally take GIS information as an input, has a limited ability to write back

data to the database. It has nodes that can read and modify existing elements through the FDO connector, but it **cannot** create new elements or erase existing ones.

For that reason, the user will have to develop a workaround procedure to update their GIS database.

At this point, it could even be for the best, since most GIS administrators will work within their own GIS software with some automated routines that will validate any planned modifications and new data entry **prior to apply direct edits** to their GIS Database. The GIS database act as an archive for your utilities for years to come, so it needs to be protected, standardized, and properly maintained.

In that regard, we can still do some legwork with Dynamo.

From Civil 3D to AutoCAD

Most GIS software will read DWG files directly, but they won't detect nor extract any data from Civil 3D objects. With that said, some GIS software are compatible with Civil 3D (like **FME** from *Safe Software* or **ArcGIS** through *Civil 3D's ArcGIS connector*), but when they do, to our experience, they won't grab every info needed to fully recreate your pipe networks afterward and they won't reach to every possible place where you can put your utility metadata.

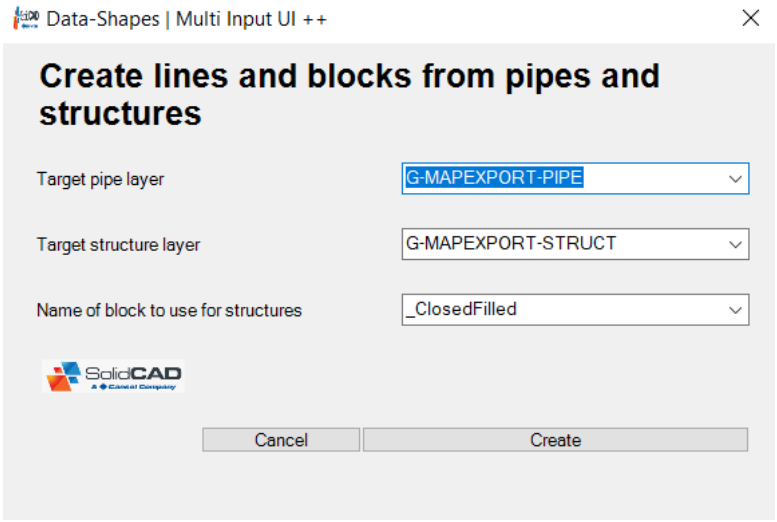
Here is the most "full-proof" way we've developed with Dynamo to grab everything that you'd need, not only to recreate a full 3D pipe network later, but to grab every single metadata you added to your pipes and structures, and "digest" all this information in AutoCAD objects that can be easily read back into your current GIS database.

Note: In this part of the demo, we've used customized buttons in our tool palette to launch our scripts. If you ever want to create your own buttons, see the *Launching Dynamo Scripts from the Tool Palettes* section of this document.

To ensure that we can gather information from every possible angle, **we split our procedure into 4 steps**.

Step 1 – Convert Civil 3D Pipe Networks to AutoCAD Objects

The first script reads all pipe networks in the current drawing, creates AutoCAD objects (lines and blocks) from them on target layers, and transfer the "standard" pipes and structures properties into them. You can even specify which block in your drawing will be used to represent your structures.



PNET_C3D_TO_ACAD.DYN

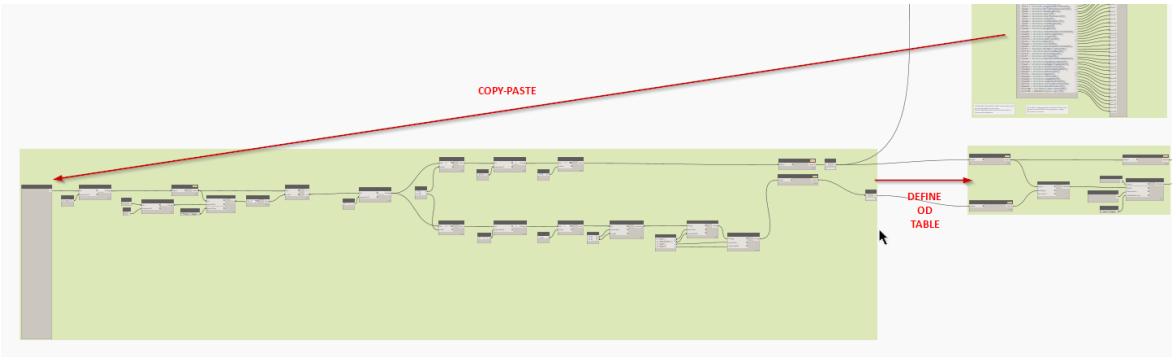
Note: After running a Dynamo script that creates either AutoCAD or Civil 3D objects, you will have to switch to a layout then back to model space to make the new objects appear. A simple REGEN command will not do the job.

If you select one of the newly created AutoCAD objects (either a line of a block) and look at its object properties, you will see every info required to eventually rebuild your pipe network, all stored within a Map 3D Object Data table that can be easily read by GIS software or exported with the MAPEXPORT command into GIS objects (like ESRI Shapefiles and Autodesk SDF files).

Since the script grabs info directly from your parts (and not the part catalog), and it should work with any custom part catalog you have used in your networks without extra adaptations of the script.

The way this script constructs the pipe/structure data schema might require some explanation. The property names come straight from the Dynamo node names for these properties. The following are the steps:

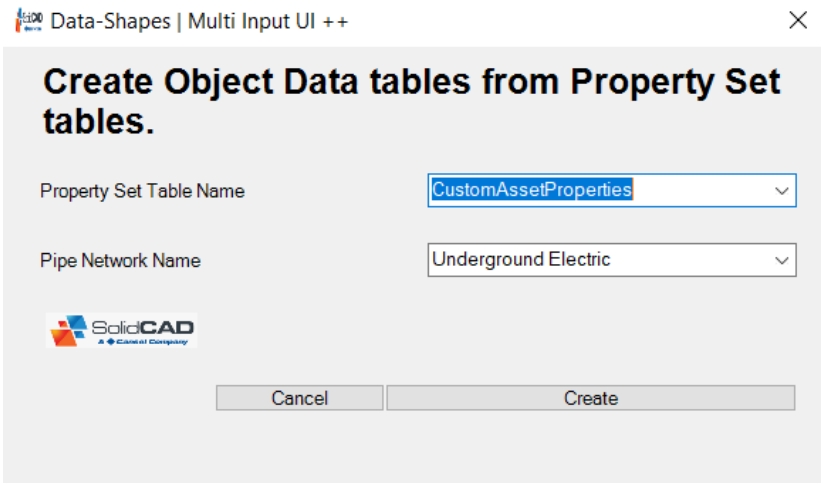
1. Get various structure or pipe properties with Dynamo structure/pipe nodes
2. Covert those nodes to a code block with node-to-code
3. Copy-paste the node-to-code into a string node
4. Manipulate the text to extract a list of property names and its corresponding data type (see "Generate OD field names and data types from copy-paste of node-to-code" group)
5. Use these lists to define an object data table.



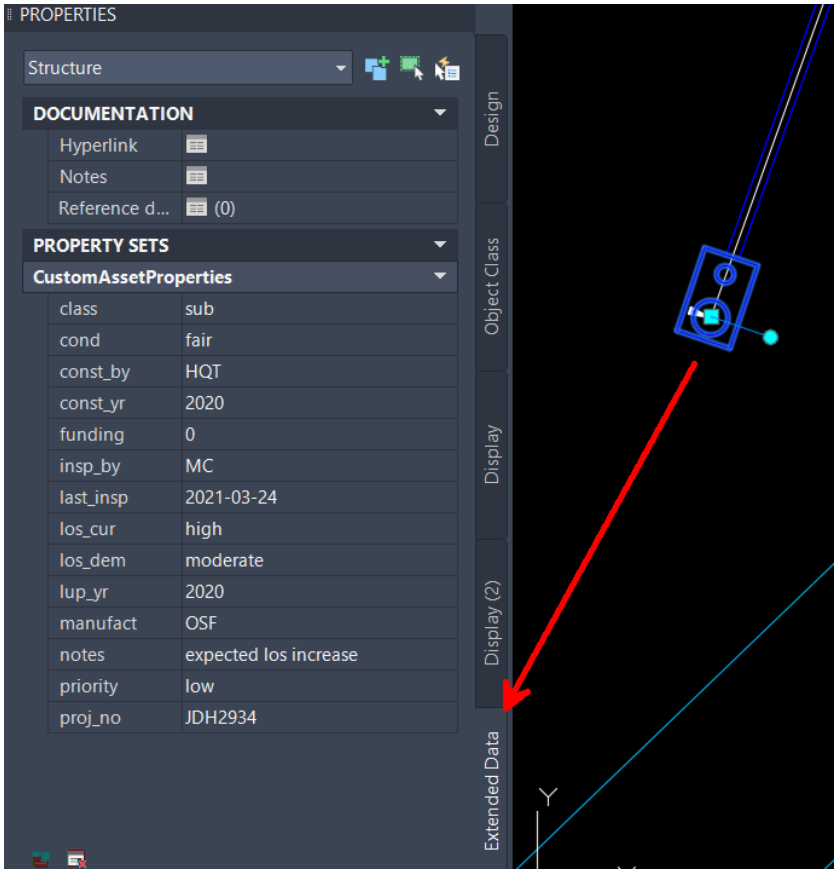
Step 2 – Create An Object Data Table Structure From Any Property Sets (*Extended Data* Tab)

The next two scripts will work within a target pipe networks to gather all your custom utility metadata into a more versatile Object Data table.

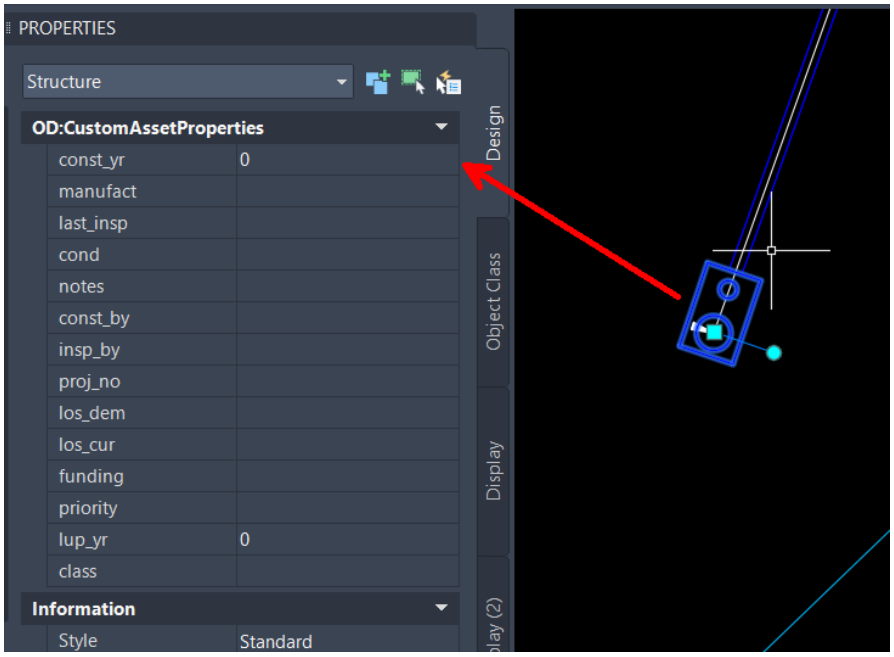
We first recreate the table structure of a target Property set assigned in the Extended data tab, respecting all preset data types for each data fields (*characters* into *characters*, *integers* into *integers*, *real* into *real*).



PNET_PS_TO_OD_DEFINITION.DYN



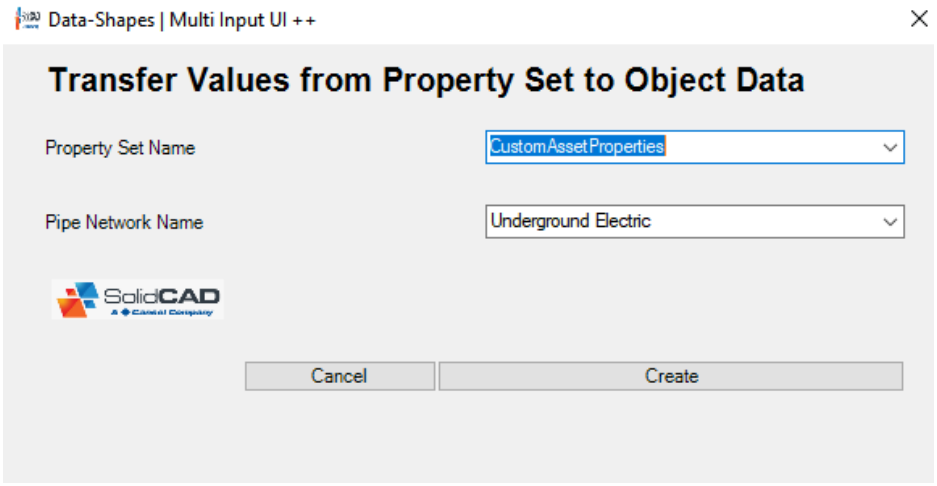
PROPERTY SET AVAILABLE UNDER THE EXTENDED DATA TAB, USED TO STORE OBJECTS METADATA



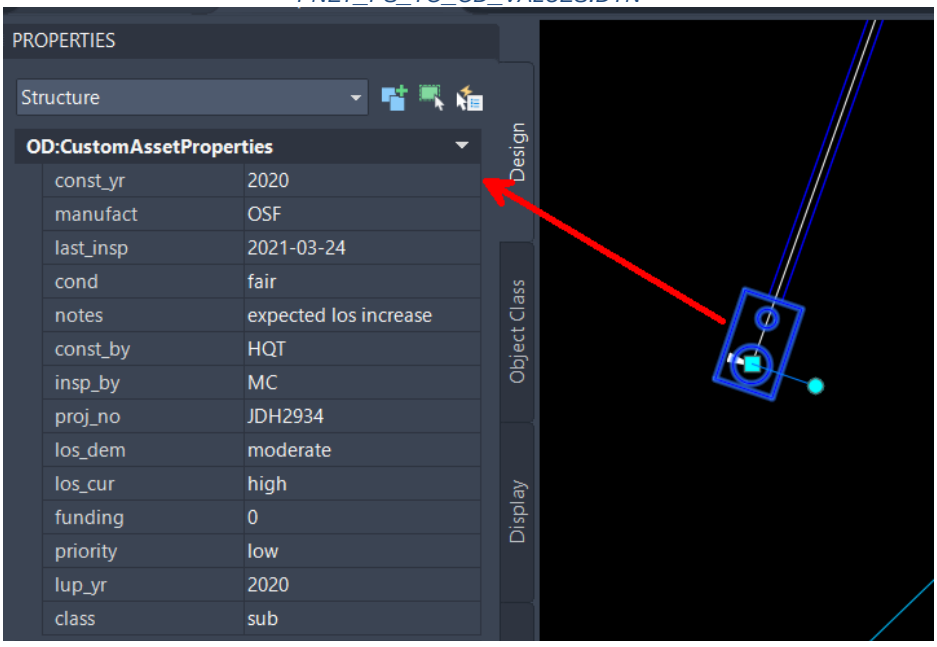
OBJECT DATA TABLE BUILT BASED ON A PROPERTY SET AND ATTACHED TO A STRUCTURE

Step 3 – Transfer Property Set Values to Object Data Tables

The third step will transfer the target Property Set content to the target Object Data table (newly created). It was much easier for us to separate the table creation from the content transfer into two separate scripts, but you can easily launch them back-to-back under a single Tool Palette button (see the *Launching Dynamo Scripts from the Tool Palettes* section of this document).



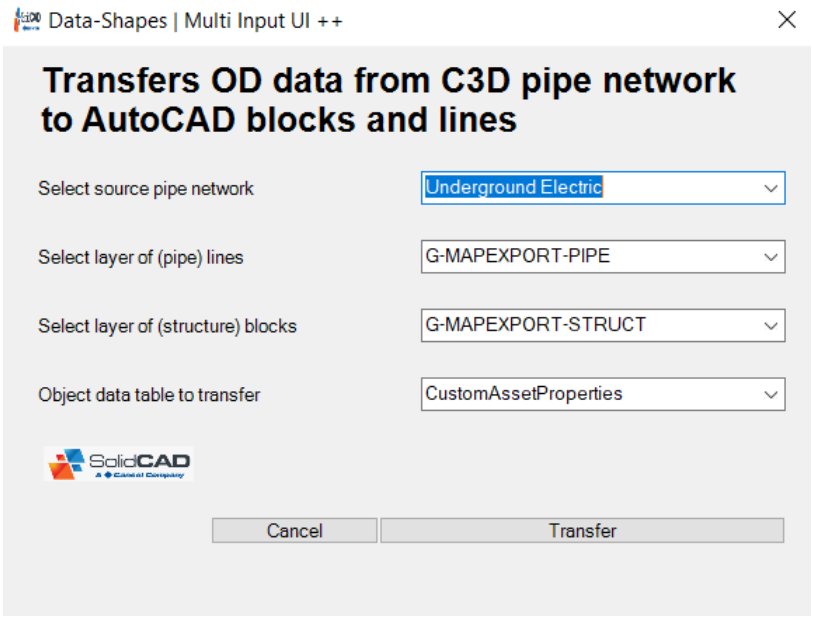
PNET_PS_TO_OD_VALUES.DYN



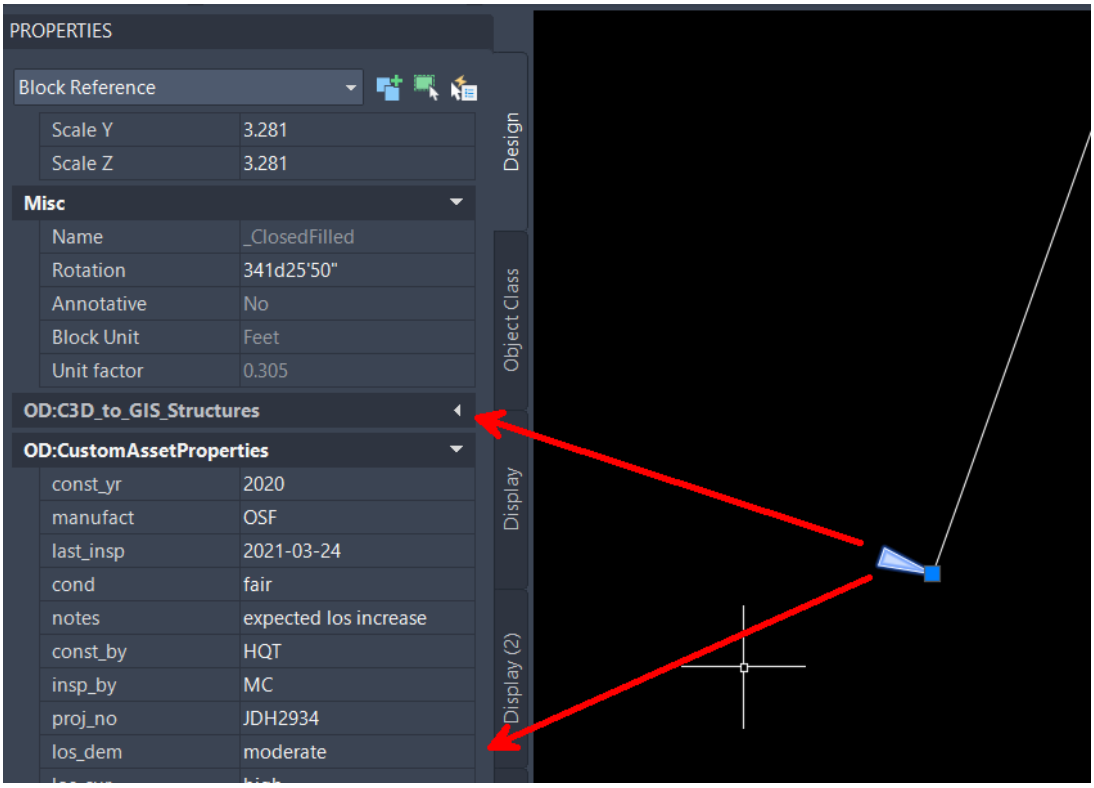
FILLED OBJECT DATA TABLE IN THE STRUCTURE PROPERTIES

Step 4 – Copy The Object Data Table From Pipe Network Parts To AutoCAD Objects

The fourth step will transfer the target Object Data table from a target pipe network to correspondent AutoCAD objects (line segments and blocks) located on target layers. Our script identifies correspondent “target” blocks and lines based on common coordinates of insertion points (for structure) or mid-points (for pipes).



PNET_TRANSFER_OD_TO_CAD.DYN



BLOCK PROPERTIES WITH THE STRUCTURE OBJECT DATA TABLES (METADATA AND PART PROPERTIES)

A GIS administrator could easily use one of your many custom properties to automatically assign an action back to the GIS database, like using the original component layer name to split your design by unchanged parts, modified parts, added parts and parts to be removed.

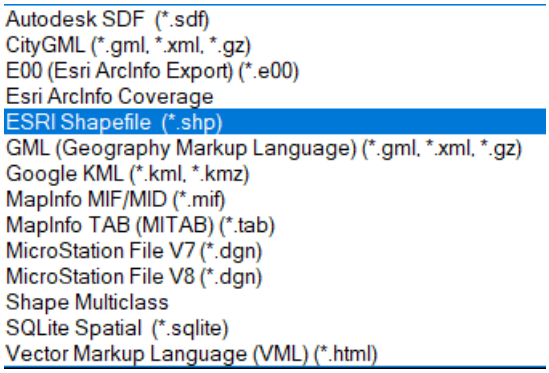
From AutoCAD to GIS

While it was discussed in fewer details in the demo, there is a couple of scenarios to send back your pipe network as AutoCAD objects back to GIS. Let's explore 2 of them:

1. Using the **MAPEXPORT** command to create GIS files.
2. Using a 3rd party GIS integration software, in this case ***FME*** from *Safe Software*.

Using the MAPEXPORT Command

The MAPEXPORT command is a powerful feature within Map 3D/Civil 3D. It allows the user to export AutoCAD objects into a GIS friendly format, like these:



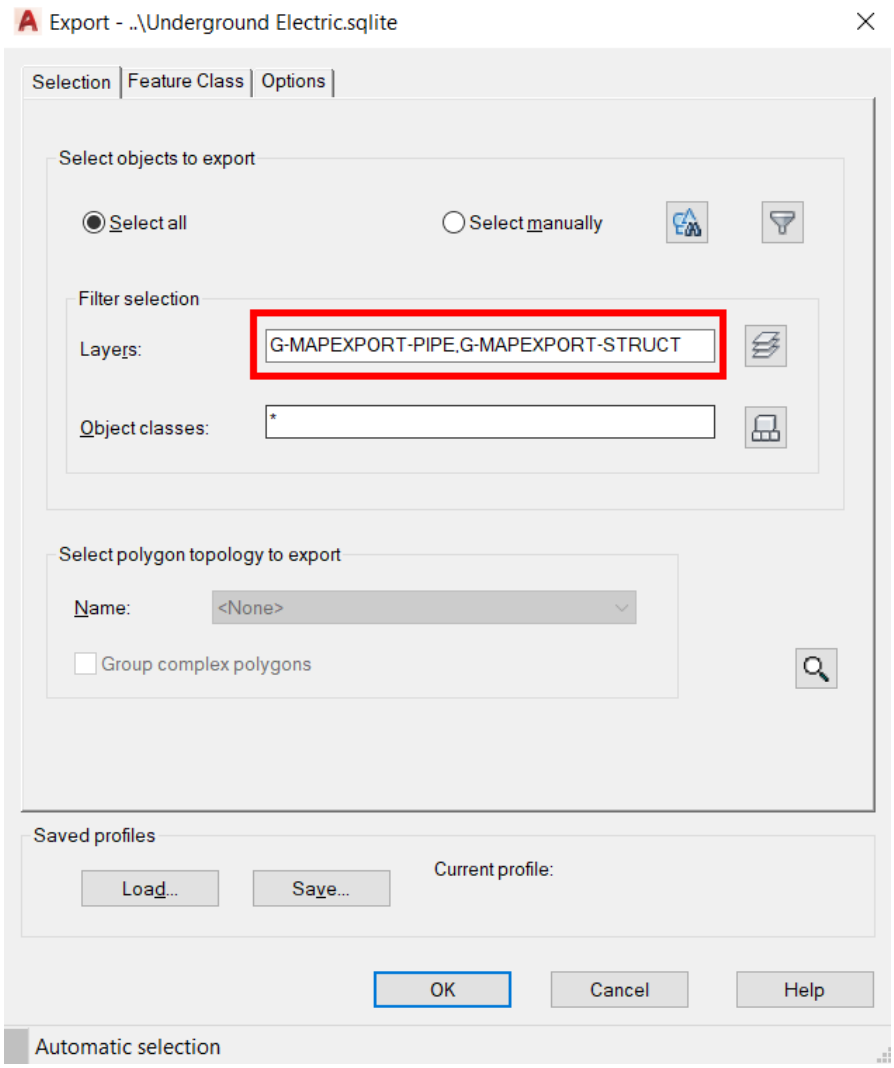
MAPEXPORT EXPORTABLE FILE FORMATS

The most commonly used/known format in that list would be Shapefiles, but keep in mind that Shapefile is an old format that has limitations in characters and content (i.e.: the length of property/column name, limited to 8 characters) that could lead in a loss of data, as you export your pipe network.

While the demonstration showed briefly how to export your pipe network in a SDF file, it is a proprietary format to Autodesk rarely compatible with GIS software.

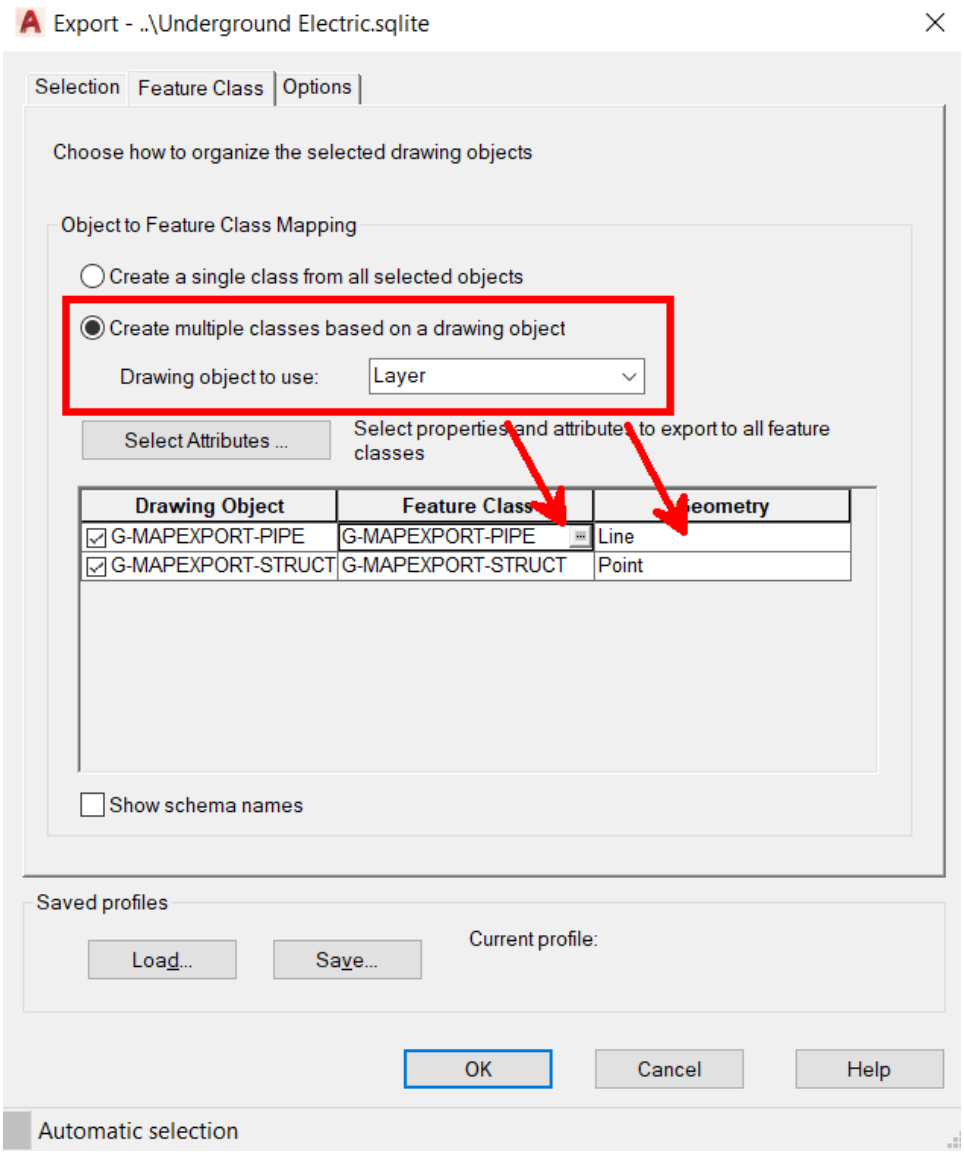
We would suggest, like in the following procedure, that you export your objects as in SQLite Spatial database (a more open, versatile, and superior option to Shapefiles).

Launch MAPEXPORT and select all objects on the chosen layers:



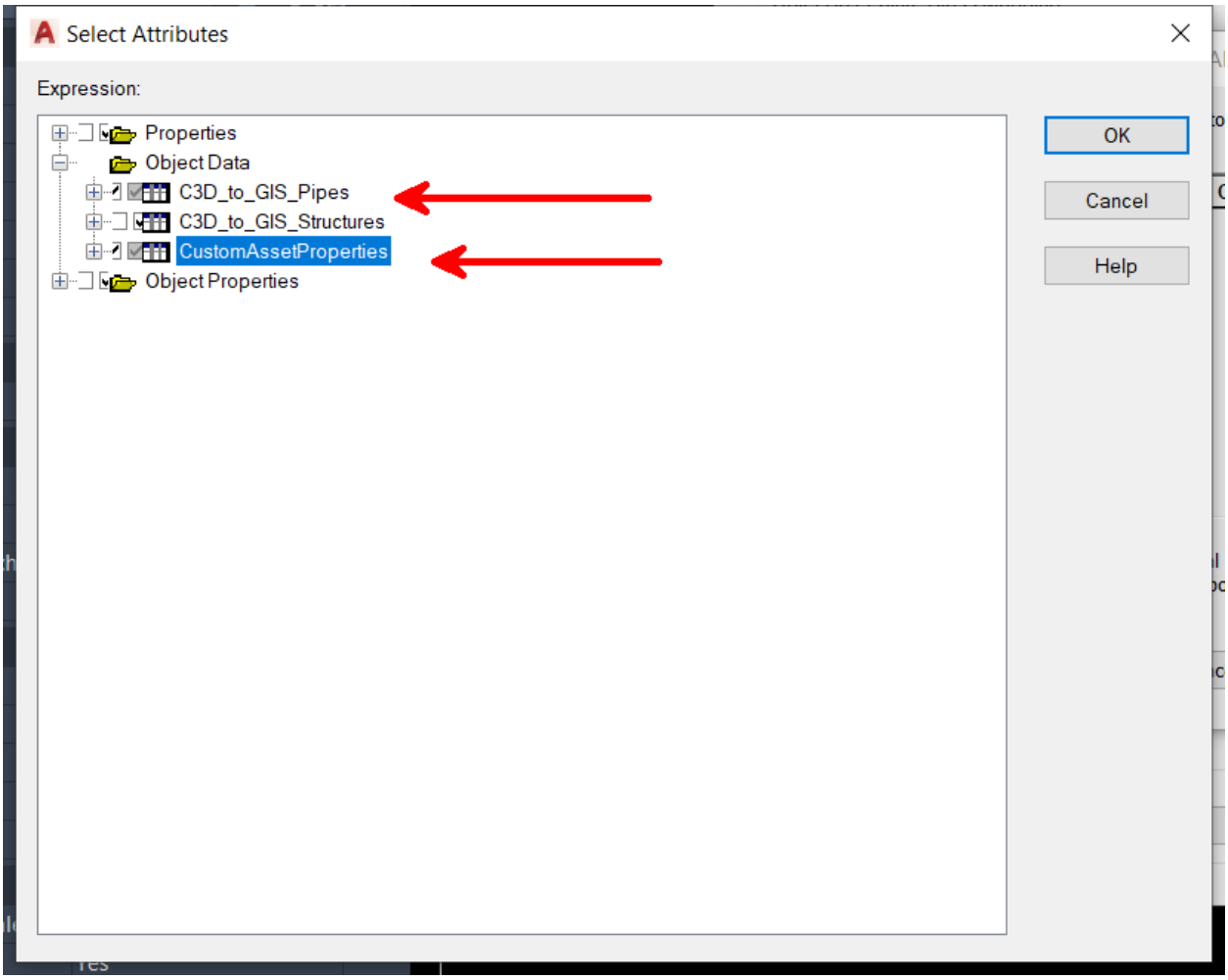
MAPEXPORT MENU, ONCE YOU HAVE CHOSEN A FILE TYPE TO EXPORT

Under the Feature Class tab, select the Create multiple classes […] option, base on Layers, then specify the target geometry type for your structures and pipes, and then select the "…" button for each feature type to select which attributes to export:



MAPEXPORT FEATURE CLASS TAB, ALLOWING YOU TO STRUCTURE THE OUTPUTTED DATA

For pipes and structures, click the checkboxes of the appropriate Object Data tables to export. In this case, we will merge part properties and metadata into a single data table.



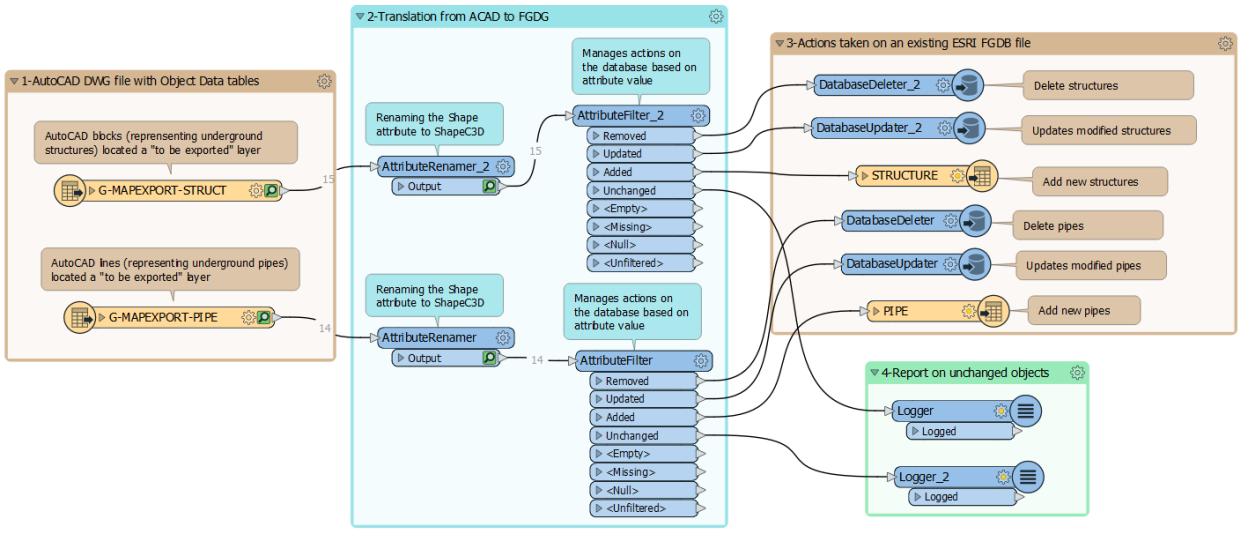
THE SELECT ATTRIBUTES MENU, ALLOWING YOU TO BUILD A DATA TABLE BASE ON AUTOCAD OBJECT PROPERTIES

Once this is fixed for both your pipes et structures, press the Save... button to create an Exportation Profile that will streamline this whole setup process next time your launch MAPEXPORT for this purpose.

Using FME To Translate CAD Data Back To GIS

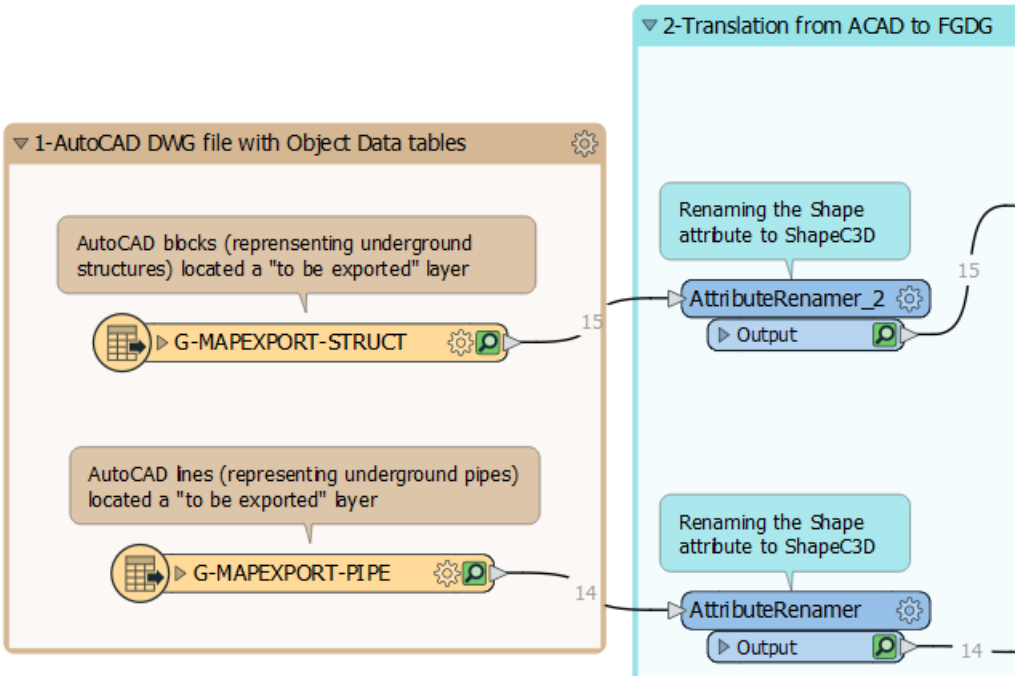
FME is a visual scripting environment meant to do data translations and quality control “from point A to point B” for all your spatial and non-spatial data. While it can read Civil 3D objects directly within your DWG file, it reads a limited amount of information, so we would suggest that you read only the AutoCAD objects previously created with Dynamo.

Here is a simple translation (called “workspace”) from our previous AutoCAD DWG file, that will split up our pipe network parts into *Removed*, *Updated*, *Added* and *Unchanged* categories (based on an object attribute value), then will take specific actions based on these categories directly in an existing ESRI FGDB database.



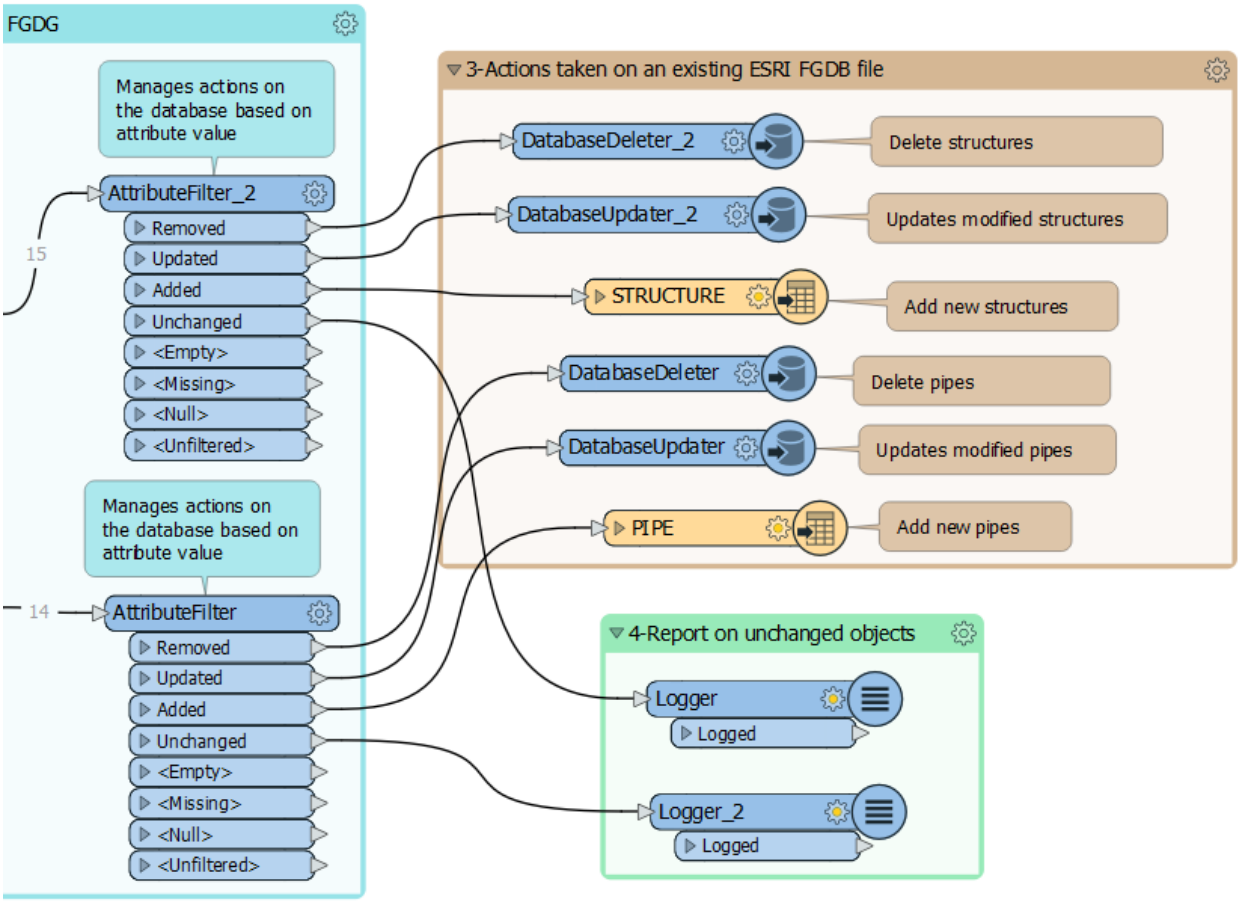
OVERVIEW OF THE WORKSPACE BUILT IN FME WORKBENCH

On the left side, our “exportable” layers from our DWG file are read directly, and attributes (like the *Shape* one) will be renamed to dodge a naming convention conflict as we write into our ESRI FGDB database.



OVERVIEW OF THE WORKSPACE BUILT IN FME WORKBENCH

On the right side, we split objects based on attributes (as “actions to be applied”) and branch them in different actions in our ESRI FGDB database.

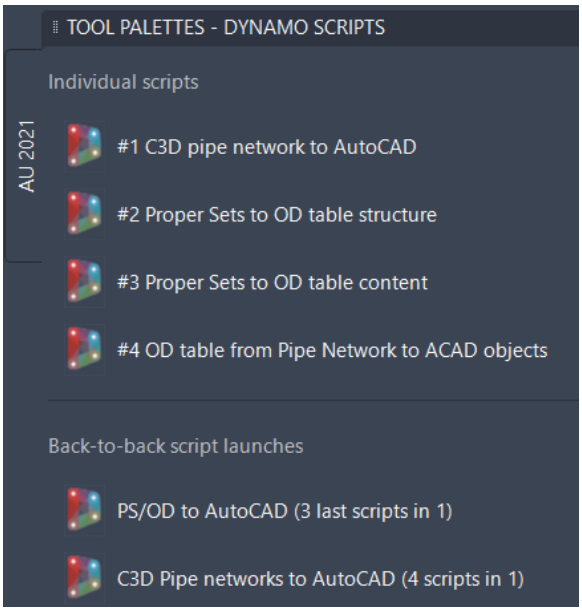


OVERVIEW OF THE WORKSPACE BUILT IN FME WORKBENCH

This workspace was oversimplified to display the potential of automated GIS actions but could easily be improved with extra quality control features and create all sorts of automatic reports and quantity take-off documents with the filtered objects.

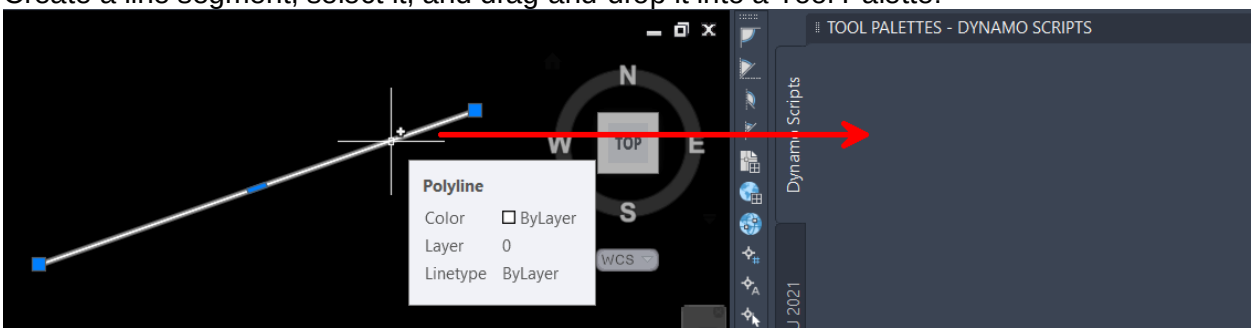
Launching Dynamo Scripts From The Tool Palettes

Here is the procedure to launch our Dynamo scripts (or any Dynamo scripts) from your *Tool Palettes*:

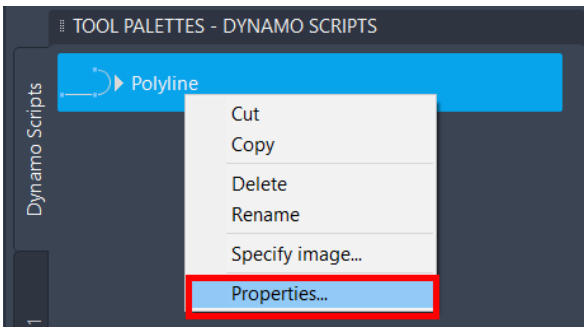


FILLED OBJECT DATA TABLE

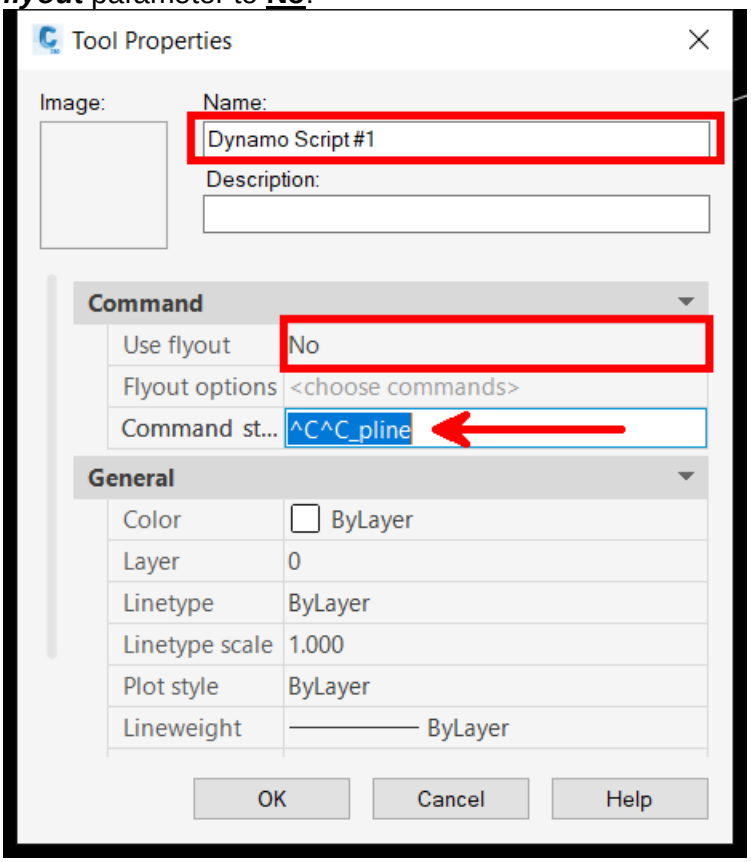
Create a line segment, select it, and drag-and-drop it into a Tool Palette:



Right-click on the new button and select *Properties…*



In the Tool Properties menu, assign a proper name for your new button and change the ***Use flyout*** parameter to **No**:



Then, in the ***Command string*** input, copy-paste the following command line (where the full path name can be changed to your own dynamo script):

```
^C^Caeccrundynamoscript
"C:/AutodeskUniversity/Temp/cad2gis_dynamo/2_pnet_c3d_to_acad/pnet_c3d_to_acad.dyn"
```

Be aware that copying paths from *Windows Explorer* creates backslashes [\] that needs to be replaced by forward slashes [/].

Through the ***Command string*** input, you can load back-to-back as many scripts as you want like this:

```
^C^Caeccrundynamoscript
"C:/AutodeskUniversity/Temp/cad2gis_dynamo/2_pnet_c3d_to_acad/pnet_c3d_to_acad.dyn";
aeccrundynamoscript
"C:/AutodeskUniversity/Temp/cad2gis_dynamo/2B_pnet_c3d_data_xfer/1-
pnet_ps_to_od_definition.dyn";aeccrundynamoscript
"C:/AutodeskUniversity/Temp/cad2gis_dynamo/2B_pnet_c3d_data_xfer/2-
pnet_ps_to_od_values.dyn";aeccrundynamoscript
"C:/AutodeskUniversity/Temp/cad2gis_dynamo/2B_pnet_c3d_data_xfer/3-
pnet_transfer_od_to_cad.dyn"
```

[^C] means pressing the <Escape> key once.
[aeccrundynamoscript] is the command to launch a script at a specified file path in “quotes”.
[;] means pressing the <Enter> key once.

This way, each following script will wait for the previous script to finish before launching itself, making it an easy way to divide your scripts for clarity’s sake while streamlining a set of actions under a single button.

See the following page to explore more “special control characters” available to build your own Command string:
https://knowledge.autodesk.com/support/autocad-lt/learn-explore/caas/CloudHelp/cloudhelp/2019/ENU/AutoCAD-LT/files/GUID-DDDB6E26-75E1-4643-8C6A-BEAEBA83A424-htm.html

Conclusion

The potential connectivity between GIS and Civil 3D through Dynamo scripts was shown, using complex Civil 3D objects like pipe networks. Further, we could adapt this workflow without any problems to other Civil 3D and AutoCAD object.

We hope that this demonstration will lead to further conversations in the Autodesk community to expand these kinds of applications, connecting Map 3D and Civil 3D through Dynamo for Civil 3D.