Autodesk University 2012

You Did What? AutoCAD® Revit® MEP and AutoCAD® P&ID? Amazing!

David Butts – Gannett Fleming, Inc

MP1461

Autodesk is constantly looking for ways to improve the user experience. Interoperability between programs is a hallmark of this policy, but sometimes they haven't quite gotten us there…yet. At Gannett Fleming, we wanted to start using AutoCAD P&ID software, but also wanted to find a way to link the data to AutoCAD Revit MEP software. And that's what this class is about. We will cover how to review and determine the need for Revit optimization, how to approach optimization from the proper perspective, what you need to know from the program developer to novice user, and how to get it implemented. As an example, we will review how we approached linking data between Revit MEP instrumentation and AutoCAD P&ID drawing data. You'll walk away with the added benefit of planning documents to help smooth the way. You will also learn what works and what doesn't, so be prepared to take your Revit to the next level!

Learning Objectives

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

- Plan for and review Revit programming capabilities
- Set up and manage a programming project
- Budget and plan for use and implementation
- Describe how the Gannett Fleming project data sync tool works

About the Speaker

David is a BIM Specialist for Gannett Fleming, a multi-discipline engineering firm based in Camp Hill, PA, with 60 offices in the US and overseas. Based in the Raleigh, NC office, he provides BIM Implementation and training for the firm's engineering design software, including Revit, Navisworks, AutoCAD MEP/P&ID and more. He has 27 years of experience in both the design and Autodesk VAR channel, spending 13 years working as an instructor and consultant for the Autodesk building design product line. David also worked as a training manager while in the channel, and was a member of the Autodesk ATC Advisory Board for 2009-10. He is a Revit Architecture Certified Professional, and also earned the MEP Implementation Certified Expert title.

David has spoken at AU for several years, and was named the Top Speaker for both labs and lectures at AU 2011. As an author, he also contributes to 4D Technology's CADLearning training programs and has written several training manuals on Revit MEP.

You can reach David via email at dabfvnc1@nc.rr.com.

Introduction

Why customize?

Why, indeed…and if there's anything the Autodesk user crowd knows how to do, is tweak, push, pull and convolute AutoCAD and other products to get what they need. Over the years, the biggest strength of AutoCAD (as well as its biggest weakness), is its ability to be customized. As Autodesk expanded their offerings into BIM applications such as Revit, this task became a little more difficult. Actually, it became a lot more difficult. For the layman user, you just didn't have access to the same programming environment. And…gasp…you actually had to learn how the program worked in much greater detail.

After many years of peddling and optimizing Autodesk products, I made the leap, a couple of years ago, back to the engineering channel. The first thing I learned is the biggest obstacle for most firms – is that they've become a dependent victim of our own success. Moving from a highly customized version of AutoCAD into the latest versions of software is difficult, especially if you're comfortable. Since the industry isn't static, we learned that we have to be able to look past what we made, and really dive into other methods, if we are to continue to be competitive on a national and international level.

So the change comes from looking at what BIM, or Building Information Modeling, offers a firm such as Gannett Fleming. While the advantages of modeling in 3D, and using that model to produce construction documents, was very clear, the "I" – or the data in the model – wasn't as easy to plug into a workflow. Hence, phase two of the BIM implementation was born.

For the past 18 months, we've been taking a hard look at how data is used on a project. In a traditional project, information as it applies to equipment and other parts of the design ~~is~~ are stored in a variety of places:

- Excel spreadsheets for equipment lists and key data;
- Word documents for specifications;
- CAD drawings for diagrams and details
- 3D models for design and documentation

The problem with this structure is the same problem that occurs when items on a plan are changed, and that change affects other plans, sections, elevations and more. For example, once you really start to get into BIM, there is still a disconnect between modeling data, such as the voltage assigned to an electrical connector on a ~~model~~piece of equipment, and the voltage assigned to the symbol in an instrumentation diagram. So that's where our project began – how do we link this data, and do it in an efficient way?

As we go through this class, we'll start at the beginning. We'll review how our team developed a plan to define this type of customization for AutoCAD, Revit and more. We also cove our~~Our~~ approach to managing the project ~~will also be covered~~. Understanding the financial impact on our firm, as well as determining the return on investment, is a critical step that most firms don't take. ~~And then seeing the real world results, and the success it brings,~~ Then we will ~~be~~ demonstrate~~d~~ ~~with~~ our own internally developed application and the real world results and success it brings. ~~Hopefully~~I hope that you'll take away a more effective ~~approach, that~~approach that helps you improve your bottom line.

So…let's get started!

Starting the Plan for Revit Programming

At Gannett Fleming, we always are looking for ways to gain a competitive advantage. And there's not another firm that doesn't do this, regardless of the market. As we look forward to what the next level of design brings, companies like ours really need to make sure we've got a great plan for what and how we want to optimize the applications we use.

When you begin looking into optimizing Autodesk products, you've got to start from some ground level areas. When you begin to plan for the optimization, understanding these issues can make or break your program. Let's look at several areas that can greatly impact your implementation plan.

Unknown Features?

The most common mistake made by users is not being aware of all the features in the applications they own. For example, I've worked with several companies that have written optimizations for placing symbols in drawings, from mechanical to instrumentation and more. Most of them use a lisp routine or other programming for simple tasks, such as setting a layer current before placing the symbol. This even occurs when users have vertical application such as AutoCAD P&ID, which do this automatically. Take the idea into Revit, and you've really got to have a good understanding of what you can and can't do.

A big issue with many companies occurs when they apply the drafting mentality ~~is applied~~ to the modeling world. We don't worry as much about layers and visibility as we do about data that is associated with a part. Since the task of altering how an object is displayed in a view is ~~fairly~~ simple, it's still restricted by the features of the program. There's only so much you can alter when it comes to the program's features; for example, understanding the difference between object styles and layers demonstrates "program awareness". The staff involved in planning these tasks must be literate in Revit features, or include someone who is, in order to be successful.

Knowing the Code

Revit has a more restricted methodology for custom code. We program or “code” using~~All programs have to be developed using a ~~.NET ~~API~~ or C# ~~code~~programming language, so expertise in these ~~programming ~~languages is required. An awareness of what items can be optimized in the Revit model is also critical. Some of the items that offer interesting opportunities include:

- Editing how commands are implemented, and how shortcuts, the Ribbon and ~~Qucik~~Quick Access toolbar can be modified;
- Editing how views are defined, so the items such as schedules are defined, the creation of the views improv~~id~~ed, and the manipulation of~~ng~~ objects placed in a view can be automated;
- Engineering tools to improve how the calculation tools are factored, and how routing preferences can be defined for a project;
- The import and export of data:~~,~~ IFC files, DGN files, DWG files, and DXF files. Revit has robust tools for sharing data with other applications, which is one area we are starting to dive into deeply.

Determining the Need

This area is by far the most critical. Too often, managers get motivated to push for optimization when they don’t know enough about the first two items. You can’t determine need without knowing the pain, and knowing what tools are available to solve the pain. As you work to determine need, you need to be asking these questions:

- Can the problem be clearly defined and described, down to the layman’s level?
- Can you describe an overall plan for how the programming should work?
- Can you determine a return on investment for the programming?
- Can you explain how the need was determined?
- Can you show how productivity, accuracy and/or coordination during the design process is improved?
- Can you develop an implementation plan the includes critical team members, and benchmarks for timeline and task completion
- Can you track the progress of the plan?
- Can you set a budget for the programming? Have you clearly defined the tasks~~Are the tasks clearly defined,~~ so you can assign a budget?~~ can be assigned?~~
- Can you clearly show benefit to clients and prospects?
- Can you describe other competitive tools that might be considered, as opposed to internally developing the programming?
- Can the programming help you win business?

Amazing enough, decisions on programming are not typically made at the management level, since most of these individuals in the engineering world don’t work with the applications to be fully informed. In most cases, money is being spent at the technical level to work on these tasks without any awareness of management, until after the fact.

In order to develop a solid plan about the customizations you want or need for your organization, you have to be able to address all of these situations – before a single dime is spent writing code.

Describing the Datasync Project

In our case, we knew before starting our project that sharing data between applications was the key goal. One of the advantages of Building Information Modeling (BIM) is the ability to store most data about equipment in a single file. Revit does this ~~nicely,~~nicely; allowing a user to define any type of data fields needed, or leverage the default parameters in a project. This includes common data such as equipment tag, manufacturer and model.

In order to justify the project, we had to determine need. A review of recent projects showed a common issue that occurred when work was being completed in several offices, using base AutoCAD for the drawings, Excel for the equipment lists, and Word for the specifications. The recurring problem was that data, such as an equipment tag, had to be manually synced between programs and documentation. Simple changes such as editing the tag number could occur in several places. If there were any communication issues between team members, the data could be incorrect in one or more location.

As we reviewed the problem, it was quickly determined that many more items needed to be coordinated during the course of the project. The first step was to review what data needed to be linked, and make sure that it could be easily shared. Once the required properties were identified, we could move into testing the export capabilities of Revit.

The second step was to review what applications were available, that could potentially solve our need. We reviewed two.~~:~~

During this process, we decided to move forward with the implementation of AutoCAD P&ID for our firm. While it’s great that Autodesk is developing the Plant Solutions platform, the path forward was a little different from where we were headed. Since we had already made the decision from the corporate level to pursue Revit as our primary design platform, we had to make sure that this data could be coordinated. The greatest appeal of AutoCAD P&ID was the instant link of data incorporated into a drawing to a project SQL database. We also were impressed with the Excel linking capabilities.

Since we already had a good, ISA standard symbol library, we determined that implementing these existing symbols into a P&ID project would be simple, especially since AutoCAD P&ID already had addressed most of the symbol content out of the box. The step up to using smart objects was an easy one for our instrumentation team.

During our initial review, we decided to pursue the Excel XLSX format, for a couple of reasons. While we were comfortable with the AutoCAD P&ID features and workflow, we also were aware of the capabilities in the Revit project database. For example, the ~~work flow~~workflow for exporting data from Revit to Excel was already established, and we had examples of Excel import and export applications we could refer to as an example, or use as a tool for our process. Our investigations into the Revit software development kit helped us realize that we could control what data could be exported, and how we could control the flow of information within our own code. Once we made this determination, it was an easy decision to move forward.

While our project eventually moved in a different direction for the coordinated data, ~~workgin~~working through these details with our internal programmers helped us to justify this project, and demonstrate how the application would benefit our design teams and our clients. In this first phase, we addressed all three of the major areas – reviewing the current programs to assure we were not duplicating existing features, understanding what the program's code would enable us to do, and determined the need for the application. Without a detailed review, the programming project would have never commenced.

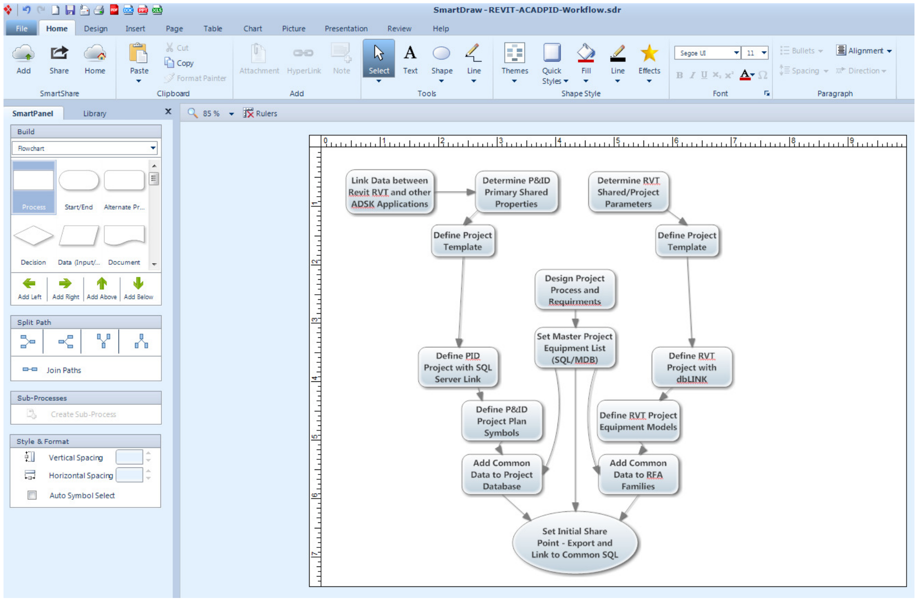
Setting up and Managing a Customization Project

Once you get an idea of the type of programming you're looking for, and what you want it to accomplish, you can never do enough planning. It's human nature to want to cover every detail, but one of the easiest way to get into trouble is by implementing the "Ready, Fire, Aim" philosophy. In this section, we'll look at a couple of tools you can use to help you get your plan together, and manage the workflow once it's in progress.

Describing the Work Process

For most people, writing out or diagramming a work process is as about as much fun as painting the toenails on a bull. There are a lot of different ways you can do this, and since I'm a "visual layman" kind of technician, I turn to programs like Visio and Smartdraw to help me lay this out. It's not necessary to do this for all projects, but anything that's going to take more than an hour to work on, you'd better be using it.

In this example, I'm starting from Smartdraw diagram. Smartdraw (www.smartdraw.com) is a diagramming application that runs from about $200-500 depending on the version. When it comes to planning, it's worth it to have a tool like this.


SmartDraw - REVIT-ACADPID-Workflow.sdr
File
Home
Design
Insert
Page
Table
Chart
Picture
Presentation
Review
Help
SmartShare
Clipboard
Add
Tools
Shape Style
Font
Paragraph
SmartPanel
Library
Build
Flowchart
Process
Start/End
Alternate Pr...
Decision
Data (Input/...
Document
Split Path
Join Paths
Sub-Processes
Create Sub-Process
Style & Format
Vertical Spacing
Horizontal Spacing
Auto Symbol Select
85 %
Rulers
Link Data between Revit RVT and other ADSK Applications
Determine P&ID Primary Shared Properties
Determine RVT Shared/Project Parameters
Define Project Template
Define Project Template
Design Project Process and Requirments
Set Master Project Equipment List (SQL/MDB)
Define PID Project with SQL Server Link
Define RVT Project with dbLINK
Define P&ID Project Plan Symbols
Define RVT Project Equipment Models
Add Common Data to Project Database
Add Common Data to RFA Families
Set Initial Share Point - Export and Link to Common SQL

Example 1 - The Smartdraw© Interface

In the previous example, you can start with a diagram to help you understand what it is that you want to accomplish. Leverage this tool to help you define how you expect the tool to work. The beauty of diagramming software like this is that it helps you figure out what the workflow should be, and where actions should be taking place. You can also set where expect results should be, and work to see if there are any influences that can affect the workflow.

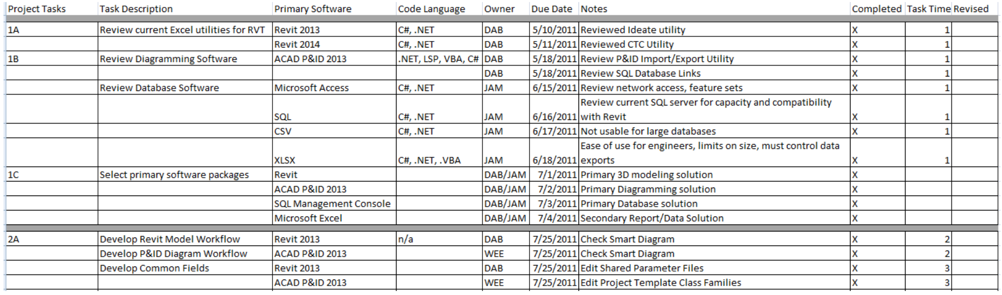
Since most process diagrams are a living document, it helps to understand that the first pass usually doesn't take everything into account. That's what I like about small packages like this – they can be edited to add new workflows, or reworked to adjust where items work better in the process. Another advantage to using these tools is that you can add the specific tools needed at each task, so the programmer knows what to expect at that point.

Managing the Tasks

I like Excel – I really do – and I use it for a lot of different tasks, But there are many different ways you can track and manage tasks. The key is to keep up with everything that happens in the course of the project. From an ownership standpoint, it's ~~not~~no different than keeping up with tasks that occur on a design project. By documenting each step, you get a better understanding of the true costs and effort involved.

For example, you can create a simple task tracking list. Here's one that I keep for typical customization projects. It describes:

- The overall name and description
- The participants and their roles
- The project tasks in sequence
- Description of the tasks
- What software the task applies to, or is used in that task
- The coding language that will be used in the final application
- Who owns the task
- Due dates, completion, task time and revision notes
- And notes or comments about the task



Project Tasks	Task Description	Primary Software	Code Language	Owner	Due Date	Notes	Completed	Task Time	Revised
1A	Review current Excel utilities for RVT	Revit 2013	C#, .NET	DAB	5/10/2011	Reviewed Ideate utility	X	1	
		Revit 2014	C#, .NET	DAB	5/11/2011	Reviewed CTC Utility	X	1	
1B	Review Diagramming Software	ACAD P&ID 2013	.NET, LSP, VBA, C#	DAB	5/18/2011	Review P&ID Import/Export Utility	X	1	
				DAB	5/18/2011	Review SQL Database Links	X	1	
	Review Database Software	Microsoft Access	C#, .NET	JAM	6/15/2011	Review network access, feature sets	X	1	
		SQL	C#, .NET	JAM	6/16/2011	Review current SQL server for capacity and compatibility with Revit	X	1	
		CSV	C#, .NET	JAM	6/17/2011	Not usable for large databases	X	1	
		XLSX	C#, .NET, .VBA	JAM	6/18/2011	Ease of use for engineers, limits on size, must control data exports	X	1	
1C	Select primary software packages	Revit		DAB/JAM	7/1/2011	Primary 3D modeling solution	X		
		ACAD P&ID 2013		DAB/JAM	7/2/2011	Primary Diagramming solution	X		
		SQL Management Console		DAB/JAM	7/3/2011	Primary Database solution	X		
		Microsoft Excel		DAB/JAM	7/4/2011	Secondary Report/Data Solution	X		
2A	Develop Revit Model Workflow	Revit 2013	n/a	DAB	7/25/2011	Check Smart Diagram	X	2	
	Develop P&ID Diagram Workflow	ACAD P&ID 2013		WEE	7/25/2011	Check Smart Diagram	X	2	
	Develop Common Fields	Revit 2013		DAB	7/25/2011	Edit Shared Parameter Files	X	3	
		ACAD P&ID 2013		WEE	7/25/2011	Edit Project Template Class Families	X	3	

It doesn't matter what you use to track the tasks, but you've got to at least keep up with them. Many project management applications can link to your email calendars, so you can integrate all of this together into a single interface as needed.

During the Project – Testing and Other ~~tasks~~Tasks

Formatted: Heading 3

Once the project gets started, you need to plan to include these tasks:

- *Progress meetings* – depending on the length or duration of a job, these should be held from a weekly basis up to monthly. Each team member, from the developer to the tester, should be involved.
- *Portion testing* – rather than dumping the entire program into a user's lap, consider sending out the code in smaller batches, to get feedback on the specific feature. With longer applications, this helps iron out the kinks, and makes review much easier on the tester.
- *Summary testing* – Summary testing should occur at key benchmarks , and include programming up to specific points or features. For example, we completed a summary test of the export feature prior to deploying any of the import tools, so the user could focus on just that tool.

Formatted: Font: Italic

Formatted: Bulleted + Level: 1 + Aligned at: 0.75" + Indent at: 1"

Formatted: Font: Italic

Formatted: Font: Italic

- *Interface development* – don’t overlook the importance of how the interface looks. Don’t get caught up in the opposite trap – pretty doesn’t always translate to “efficient”.
- *Sample Projects* – the K.I.S.S “keep it simple axiom applies for early testing. Use small project files to get the workflow down. Longer files that generate larger results can cause a user to become bored, or ignore intent.
- *Live Projects* – once sample projects are complete, the last stage is to deploy on a live project. The biggest rule in this case is to roll the program out to experienced, qualified users. In other words, you’ll get better feedback from a strong Revit or AutoCAD P&ID user that can grasp “big-picture” concepts that a user that has difficulty retaining more than 5 minutes worth of instruction. Be selective!

Formatted: Font: Italic

Formatted: Font: Italic

Formatted: Font: Italic

Staying on Budget and Getting the Tools to the User

Once the project is going, making sure that you’re not spending an overwhelming amount of time and resources in the wrong place is essential to gaining your return on investment. You also need to develop a simple way to get files to the users for testing and implementation. During this phase of the project, there are a few ~~couple of~~ items you need to be aware of ~~when it comes~~ to stay~~ing~~ on budget.

Communicate!

Meet regularly with the development team. For our project, we meet on a biweekly basis online, and reviewed the progress of the project. Since we were getting portions of the program to test during development, it made it easier to make adjustments in the workflow. Too few meetings could result in major rework in the coding, and cause unnecessary costs and changes. Too many, and you are wasting money on meetings that should be devoted to code.

Follow Benchmarks for Programming

As our team progressed through the project, a test version ~~would be~~was provided to the team as each portion was completed. During this phase partial runs gave the testing team the opportunity to provide feedback. As the tools became relevant to other teams, they were brought into testing as well. Better known as “release engineering”, the programming team used this to accomplish several goals:

- Identify problems with the code
- Identify the environment for the tools for better utilization
- Define reproducible results, based on the source application, any third party components that could have compatibility issues, and test the data being manipulated
- Test for consistency in the results, assuring the correct behavior for the users
- Check the agility of the program, and its ability to withstand change in the code

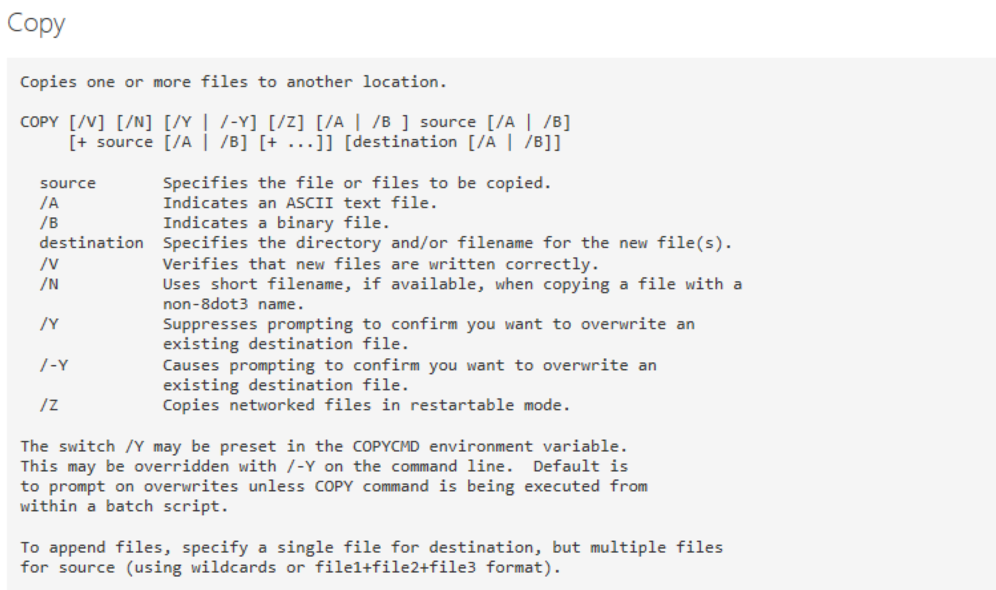
Every benchmark should have a deliverable date. With most programming jobs in AutoCAD, this protocol was rarely followed. The code would be delivered when it's done, and then countless changes would take place, as the production team would provide feedback. The benchmark date is also part of the revision numbering scheme. This helps the programmers and testers keep up with changes to the application.

Once advantage to benchmark programming is that is allows more flexibility in the scope of work. If you get to step 2 of 9, and you can see where more issues are arising, you can back up and re-~~examing~~examine the process or tools. It's better to do this in phases than as one entire shot.

Testing and Deployment

During the testing phase, we did not utilize an MSI or other means of deployment. Original files, ~~,~~such as program DLL's or XML/text files were manually copied to the relevant folders for Revit. For our case, we created a custom folder under the C:\ProgramData\Autodesk\Revit\Addins location, where other tools (such as the Revit extensions) were stored. Keeping these separated makes migration to newer releases much easier, as we only needed to copy the custom folder contents to the new addins location for Revit.

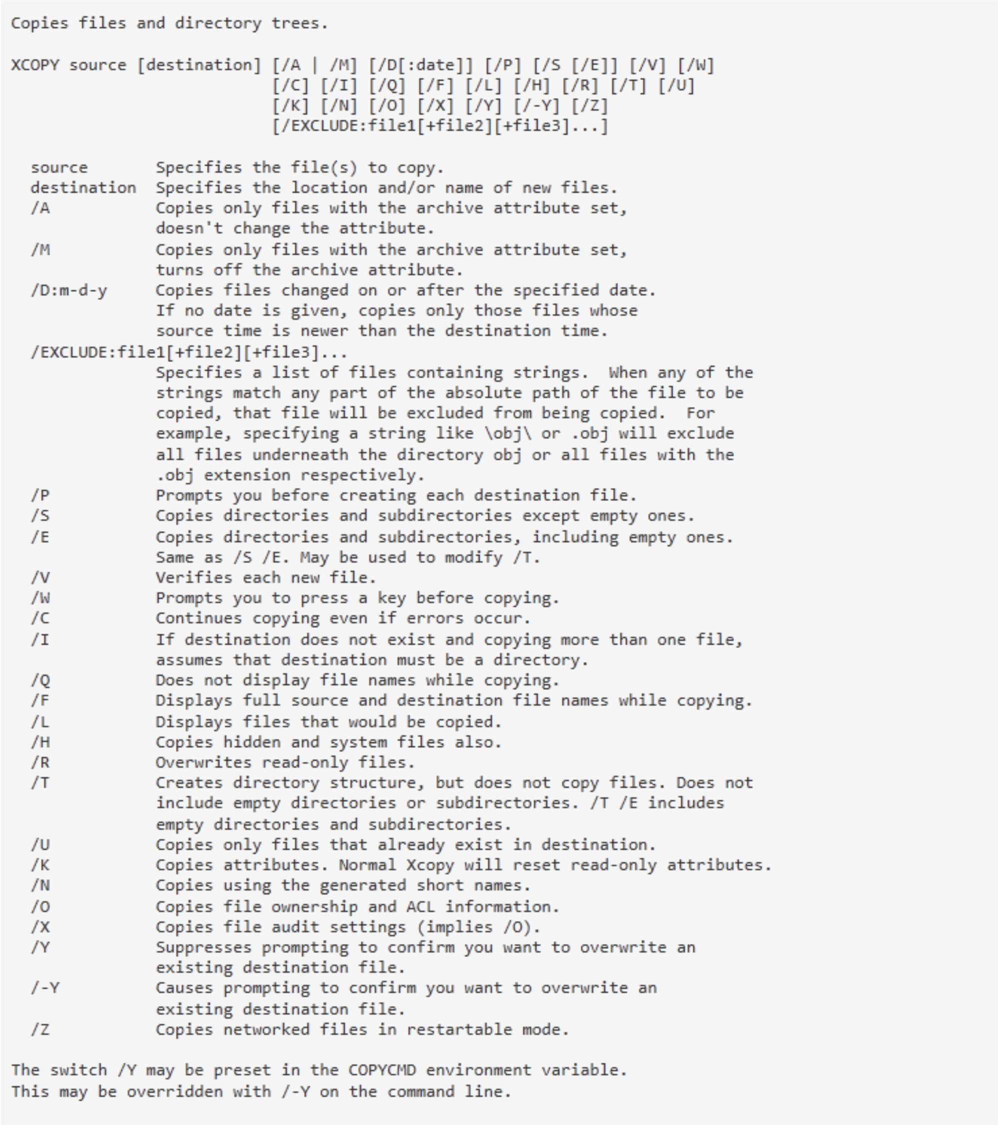
For full deployment, you have a variety of options. For Revit, it can be a variety of simple steps. One of the simplest is to copy the files to the addins location using a batch routine. Microsoft still runs basic DOS commands from the command line. Using a simple script file, you can tell Windows to copy files from one location (such as a network drive) to another location (such as local drive. The options for the Copy command include:



Copy

```
Copies one or more files to another location.

COPY [/V] [/N] [/Y | /-Y] [/Z] [/A | /B ] source [/A | /B]
     [+ source [/A | /B] [+ ...]] [destination [/A | /B]]

  source       Specifies the file or files to be copied.
  /A           Indicates an ASCII text file.
  /B           Indicates a binary file.
  destination  Specifies the directory and/or filename for the new file(s).
  /V           Verifies that new files are written correctly.
  /N           Uses short filename, if available, when copying a file with a
               non-8dot3 name.
  /Y           Suppresses prompting to confirm you want to overwrite an
               existing destination file.
  /-Y          Causes prompting to confirm you want to overwrite an
               existing destination file.
  /Z           Copies networked files in restartable mode.

The switch /Y may be preset in the COPYCMD environment variable.
This may be overridden with /-Y on the command line.  Default is
to prompt on overwrites unless COPY command is being executed from
within a batch script.

To append files, specify a single file for destination, but multiple files
for source (using wildcards or file1+file2+file3 format).
```

The more detailed XCOPY command allows more detailed commands, including copy subfolders and more:



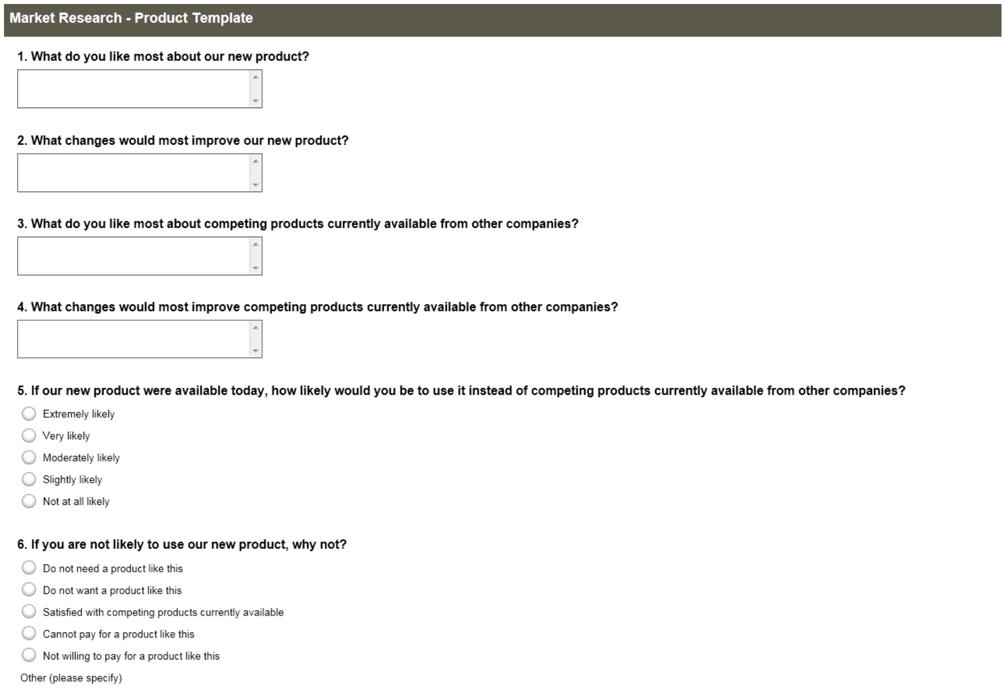
```
Copies files and directory trees.

XCOPY source [destination] [/A | /M] [/D[:date]] [/P] [/S [/E]] [/V] [/W]
                           [/C] [/I] [/Q] [/F] [/L] [/H] [/R] [/T] [/U]
                           [/K] [/N] [/O] [/X] [/Y] [/-Y] [/Z]
                           [/EXCLUDE:file1[+file2][+file3]...]

  source       Specifies the file(s) to copy.
  destination  Specifies the location and/or name of new files.
  /A           Copies only files with the archive attribute set,
               doesn't change the attribute.
  /M           Copies only files with the archive attribute set,
               turns off the archive attribute.
  /D:m-d-y     Copies files changed on or after the specified date.
               If no date is given, copies only those files whose
               source time is newer than the destination time.
  /EXCLUDE:file1[+file2][+file3]...
               Specifies a list of files containing strings.  When any of the
               strings match any part of the absolute path of the file to be
               copied, that file will be excluded from being copied.  For
               example, specifying a string like \obj\ or .obj will exclude
               all files underneath the directory obj or all files with the
               .obj extension respectively.
  /P           Prompts you before creating each destination file.
  /S           Copies directories and subdirectories except empty ones.
  /E           Copies directories and subdirectories, including empty ones.
               Same as /S /E. May be used to modify /T.
  /V           Verifies each new file.
  /W           Prompts you to press a key before copying.
  /C           Continues copying even if errors occur.
  /I           If destination does not exist and copying more than one file,
               assumes that destination must be a directory.
  /Q           Does not display file names while copying.
  /F           Displays full source and destination file names while copying.
  /L           Displays files that would be copied.
  /H           Copies hidden and system files also.
  /R           Overwrites read-only files.
  /T           Creates directory structure, but does not copy files. Does not
               include empty directories or subdirectories. /T /E includes
               empty directories and subdirectories.
  /U           Copies only files that already exist in destination.
  /K           Copies attributes. Normal Xcopy will reset read-only attributes.
  /N           Copies using the generated short names.
  /O           Copies file ownership and ACL information.
  /X           Copies file audit settings (implies /O).
  /Y           Suppresses prompting to confirm you want to overwrite an
               existing destination file.
  /-Y          Causes prompting to confirm you want to overwrite an
               existing destination file.
  /Z           Copies networked files in restartable mode.

The switch /Y may be preset in the COPYCMD environment variable.
This may be overridden with /-Y on the command line.
```

But one of the simplest steps is to use Active Directory. Your IT department can set up a script that runs when a user logs in. The script will copy files, install programs and more, by checking a system's registry at login for programs files and more. If the files exist, Windows moves on. If they don't exist, it copies the file to a predefined location. Active directory accomplishes this by searching for registry keys.

You can also edit the Revit executable configuration file to include a path to addins on a server location rather than copying to a local hard drive. While this has the advantage in simplicity, in a firm that leverages laptops for production work would have to have some form of access to a server whenever Revit is running. And since a majority of firms such as ours are moving to this model, it makes more sense to maintain a local copy.

During the testing period, we solicited feedback via email, but another valuable tool for larger projects is the Survey Monkey tool (http://www.surveymonkey.com).

Survey Monkey lets you build surveys and questionnaires that allows users to select a range of answers, or type in descriptions and comments:



Market Research - Product Template

1. What do you like most about our new product?

2. What changes would most improve our new product?

3. What do you like most about competing products currently available from other companies?

4. What changes would most improve competing products currently available from other companies?

5. If our new product were available today, how likely would you be to use it instead of competing products currently available from other companies?

○ Extremely likely
○ Very likely
○ Moderately likely
○ Slightly likely
○ Not at all likely

6. If you are not likely to use our new product, why not?

○ Do not need a product like this
○ Do not want a product like this
○ Satisfied with competing products currently available
○ Cannot pay for a product like this
○ Not willing to pay for a product like this

Other (please specify)

The results can then be compiled and used to make changes, improve features and more. By documenting the feedback, it makes justification of changes and improvements easier.

~~WORK IN PROGRESS~~

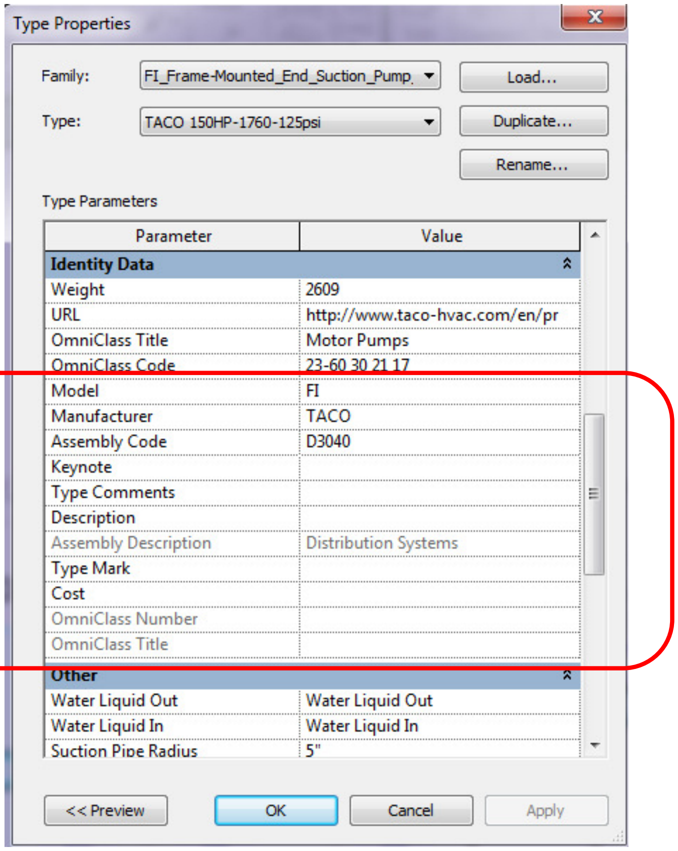
Formatted: Heading 1

The GF ProjectDataSync ~~t~~Tool – the Results!

So let's talk about our tool. Going back to the beginning, we needed a tool that would allow us to share data between a Revit MEP model and an AutoCAD P&ID diagram. In this case, we didn't want to share everything. We primarily wanted to share a few fields, such as the equipment tag, manufacturer, model number, and specific electrical data. In the next few paragraphs, we'll ~~ook~~look at a few of the key features that make this work.

Define the Data!

The first step is getting the templates in Revit and the default project in AutoCAD P&ID to have the same fields. For most users, the common identity data is already embedded in a Revit model, and is based on a type property:



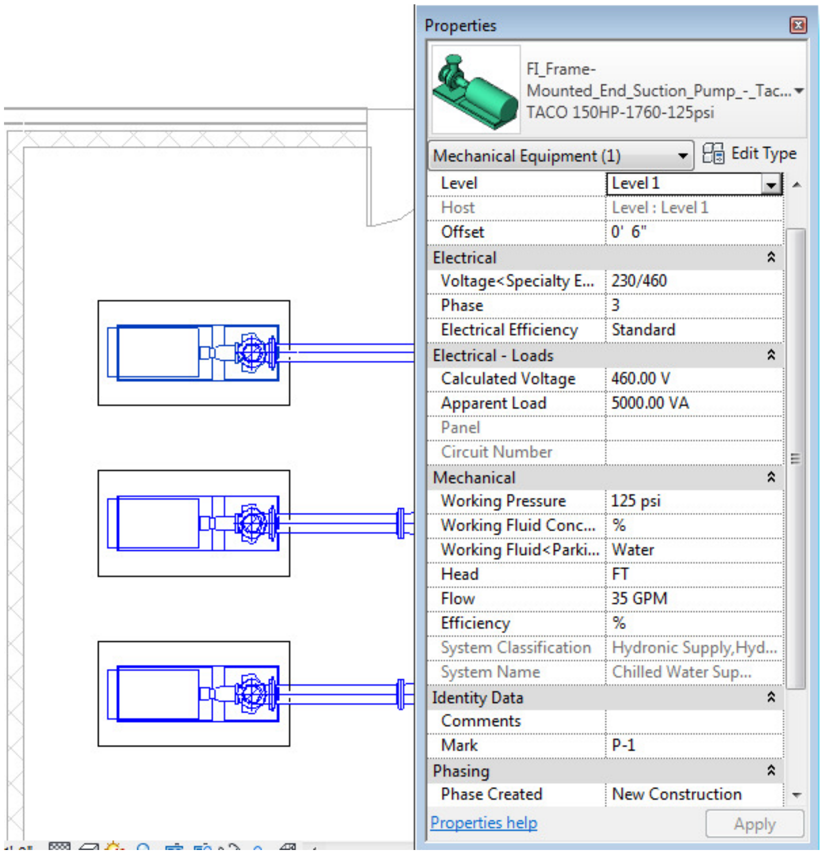
Family: FI_Frame-Mounted_End_Suction_Pump_

Type: TACO 150HP-1760-125psi

Type Parameters

Parameter	Value
Identity Data	
Weight	2609
URL	http://www.taco-hvac.com/en/pr
OmniClass Title	Motor Pumps
OmniClass Code	23-60 30 21 17
Model	FI
Manufacturer	TACO
Assembly Code	D3040
Keynote	
Type Comments	
Description	
Assembly Description	Distribution Systems
Type Mark	
Cost	
OmniClass Number	
OmniClass Title	
Other	
Water Liquid Out	Water Liquid Out
Water Liquid In	Water Liquid In
Suction Pipe Radius	5"

Since this is type based, we had to make sure that any exported data would be locked~~,~~ and editable only at a single location. Other information, such as the equipment tag, utilizes an instance mark value, so it can be unique to each piece of equipment. Potential data that is subject to change in either the Revit model or the P&ID diagram would be instance based. ~~Most data that could potentially be changed in either the Revit Model or P&ID diagram would be instance based.~~



Properties

FI_Frame-Mounted_End_Suction_Pump_-_Tac...
TACO 150HP-1760-125psi

Mechanical Equipment (1) — Edit Type

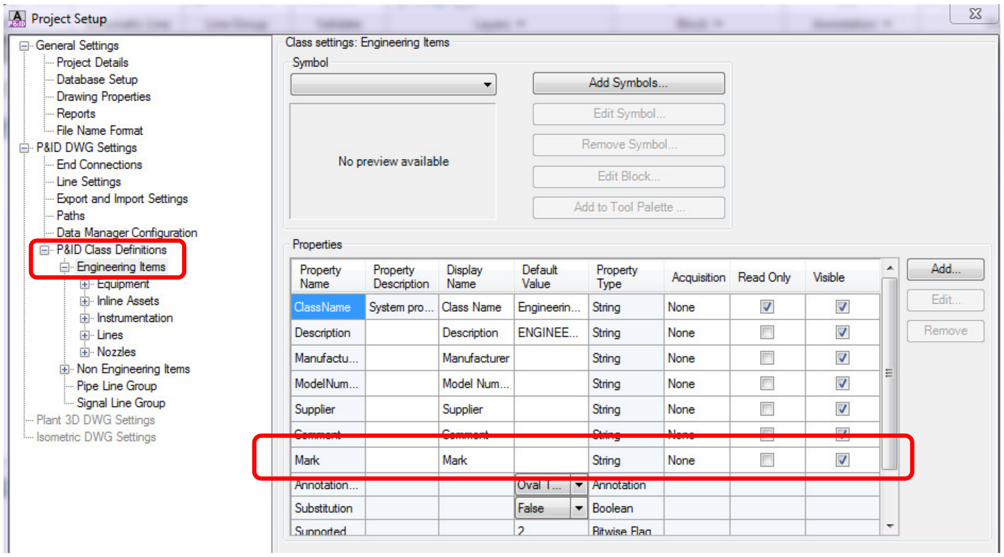
Level	Level 1
Host	Level : Level 1
Offset	0' 6"
Electrical	
Voltage<Specialty E...	230/460
Phase	3
Electrical Efficiency	Standard
Electrical - Loads	
Calculated Voltage	460.00 V
Apparent Load	5000.00 VA
Panel	
Circuit Number	
Mechanical	
Working Pressure	125 psi
Working Fluid Conc...	%
Working Fluid<Parki...	Water
Head	FT
Flow	35 GPM
Efficiency	%
System Classification	Hydronic Supply,Hyd...
System Name	Chilled Water Sup...
Identity Data	
Comments	
Mark	P-1
Phasing	
Phase Created	New Construction

Additional data was added as a shared parameter to families, so that it would be available when needed for export.

Here's a tip – use the shared parameter exporter add-in that is available as a subscription tool. This lets you add the parameters to several families at once, assuring that they are all using the same GUID identifier.

Once the parameters were defined in Revit, the default project in AutoCAD P&ID should also bear the same properties. Naming convention for these items is critical – it's a good idea that the name of the parameter matches the properties and location for AutoCAD P&ID objects,

For example, an instance mark is defined at a class definition level, which is a top level in the data hierarchy of an AutoCAD P&ID project (similar to the instance mark in Revit). Since all equipment must have a mark, the property is defined at the engineering item category:



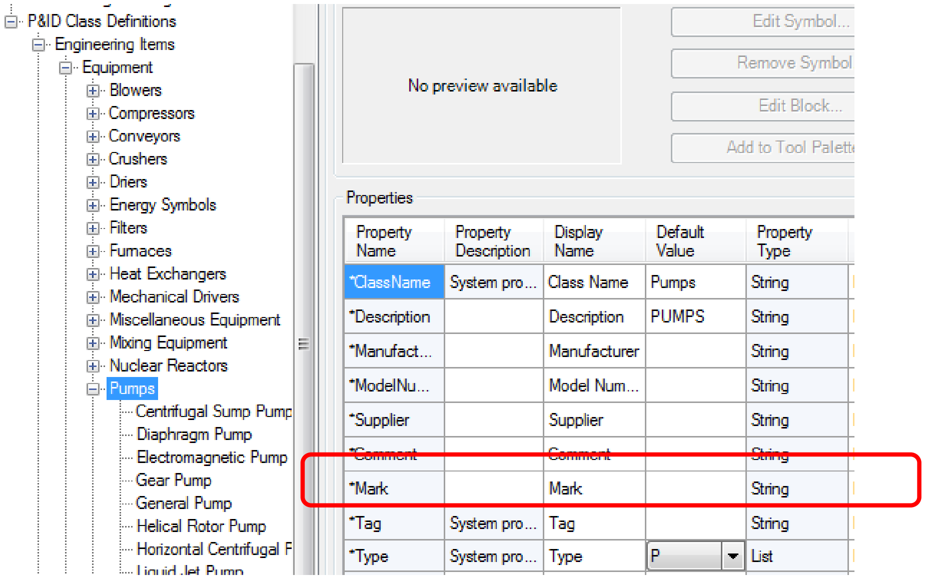
- General Settings
  - Project Details
  - Database Setup
  - Drawing Properties
  - Reports
  - File Name Format
- P&ID DWG Settings
  - End Connections
  - Line Settings
  - Export and Import Settings
  - Paths
  - Data Manager Configuration
  - P&ID Class Definitions
    - Engineering Items
      - Equipment
      - Inline Assets
      - Instrumentation
      - Lines
      - Nozzles
    - Non Engineering Items
    - Pipe Line Group
    - Signal Line Group
- Plant 3D DWG Settings
- Isometric DWG Settings

Class settings: Engineering Items

Properties

Property Name	Property Description	Display Name	Default Value	Property Type	Acquisition	Read Only	Visible
ClassName	System pro...	Class Name	Engineerin...	String	None	☑	☑
Description		Description	ENGINEE...	String	None	☐	☑
Manufactu...		Manufacturer		String	None	☐	☑
ModelNum...		Model Num...		String	None	☐	☑
Supplier		Supplier		String	None	☐	☑
Comment		Comment		String	None	☐	☑
Mark		Mark		String	None	☐	☑
Annotation...			Oval T...	Annotation			
Substitution			False	Boolean			
Supported			2	Bitwise Flag			

The property is then pushed to all sub-classes, but it still a unique property, all the way down to pump symbols in a project:



P&ID Class Definitions
- Engineering Items
  - Equipment
    - Blowers
    - Compressors
    - Conveyors
    - Crushers
    - Driers
    - Energy Symbols
    - Filters
    - Furnaces
    - Heat Exchangers
    - Mechanical Drivers
    - Miscellaneous Equipment
    - Mixing Equipment
    - Nuclear Reactors
    - Pumps
      - Centrifugal Sump Pump
      - Diaphragm Pump
      - Electromagnetic Pump
      - Gear Pump
      - General Pump
      - Helical Rotor Pump
      - Horizontal Centrifugal F
      - Liquid Jet Pump

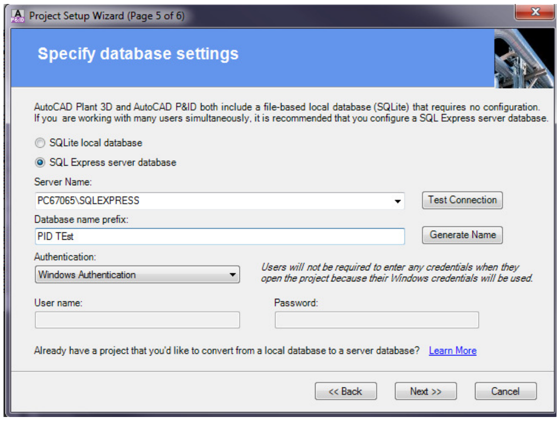
Properties

Property Name	Property Description	Display Name	Default Value	Property Type
*ClassName	System pro...	Class Name	Pumps	String
*Description		Description	PUMPS	String
*Manufact...		Manufacturer		String
*ModelNu...		Model Num...		String
*Supplier		Supplier		String
*Comment		Comment		String
*Mark		Mark		String
*Tag	System pro...	Tag		String
*Type	System pro...	Type	P	List

In order for the data sync to work correctly in this case, it's a best practice to make sure this exists in both areas before starting a project.

Understand the Shared Database

One big advantage to using both AutoCAD P&ID and Revit is their ability to communication with SQL databases, which is the primary database type we decided to use. AutoCAD P&ID is designed to work within a project environment. Each project has a link defined to one of two type types of databases:



Specify database settings

AutoCAD Plant 3D and AutoCAD P&ID both include a file-based local database (SQLite) that requires no configuration. If you are working with many users simultaneously, it is recommended that you configure a SQL Express server database.

○ SQLite local database

● SQL Express server database

Server Name: PC67065\SQLEXPRESS

Database name prefix: PID TEst

Authentication: Windows Authentication

Users will not be required to enter any credentials when they open the project because their Windows credentials will be used.

User name:

Password:

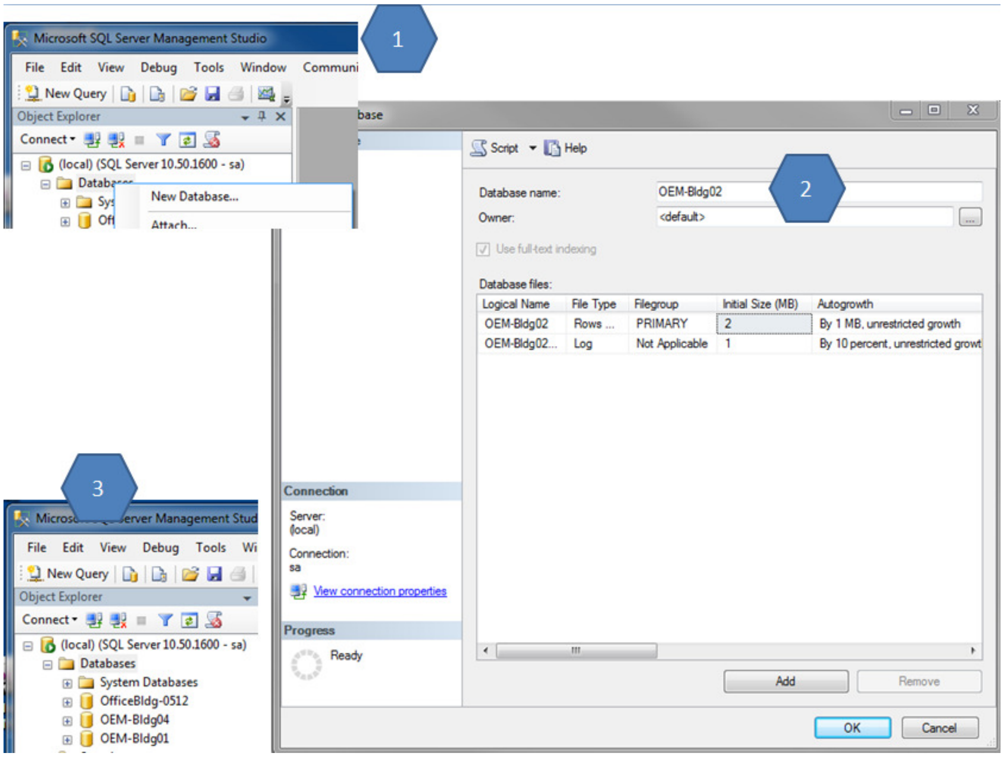
Already have a project that you'd like to convert from a local database to a server database? Learn More

AutoCAD P&ID creates an automatic association to a SQL Database, in both the form of a local, “lite” database, that gets its data strictly from what you add to the project class properties. You can also set up to use SQL Express, which is required when more than one user is assigned to work on a project. The full SQL server or express server resides at the server level in a network, but access is limited. Users can only access the database via their Windows Authentication, or via the SQL authentication. As with Revit, the SQL Express software has to be installed prior to using this option.

The other side of the data coin is the Revit database. Autodesk release the Revit DB Link utility a few releases ago, and at first blush, it takes a little bit to get going. In our initial tests, this utility allows a user to create a connection to a SQL database directly, and create a bi-directional link. At first blush, it’s seems pretty simple, but then there’s the rub – it’s not. Unlike AutoCAD P&ID, Revit doesn’t support a “lite” version. While you can create links to more types of databases, these still have to be defined before you can create the link. From a layman’s standpoint, you’d better have a couple of things straight.

Microsoft has three primary versions of SQL Express you can work with – 2005, 2008 and now, 2012. There are older versions, but the 2008 R2 release is the most common. The 2008 R2 release is the latest currently supported by Revit, as of this current date.

As with any other software package, SQL Express and the SQL Management Studio must be installed, in order for this to be used with Revit or AutoCAD P&ID. You have the option to install the database locally or on a network. After Express is installed, the management studio is used to create the local or server database:



Database name: OEM-Bldg02

Owner: <default>

Database files:

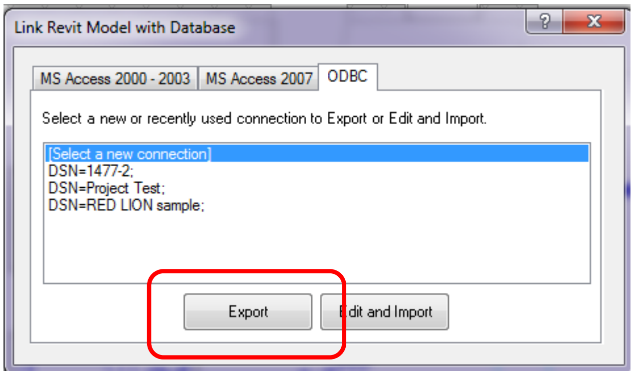
Logical Name	File Type	Filegroup	Initial Size (MB)	Autogrowth
OEM-Bldg02	Rows ...	PRIMARY	2	By 1 MB, unrestricted growth
OEM-Bldg02...	Log	Not Applicable	1	By 10 percent, unrestricted growt

Note: Be aware that you must have full administrative rights to the computer where the database will be stored. I would recommend using Windows Authentication, since this use's a ~~persons~~person's Windows login credentials to provide the shortest access.

Once the server database is defined as a named instance on your local hard drive, you can start Revit DB Link. In the application, you establish a link to your SQL database.

For new projects, do this at the beginning of the project. Existing projects that have large amounts of data can take a long time to export.

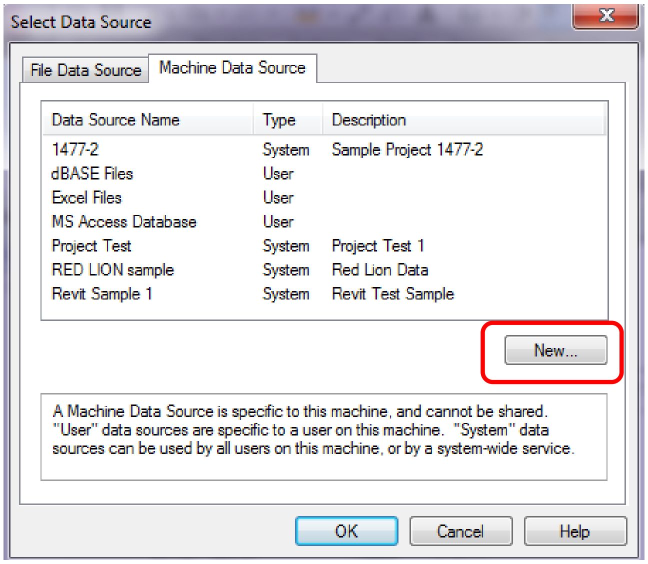
You can create database links to Access or an ODBC database such as SQL. From the ODBC tab, start by selecting a new connection, and then choosing Export:



Select a new or recently used connection to Export or Edit and Import.

[Select a new connection]
DSN=1477-2;
DSN=Project Test;
DSN=RED LION sample;

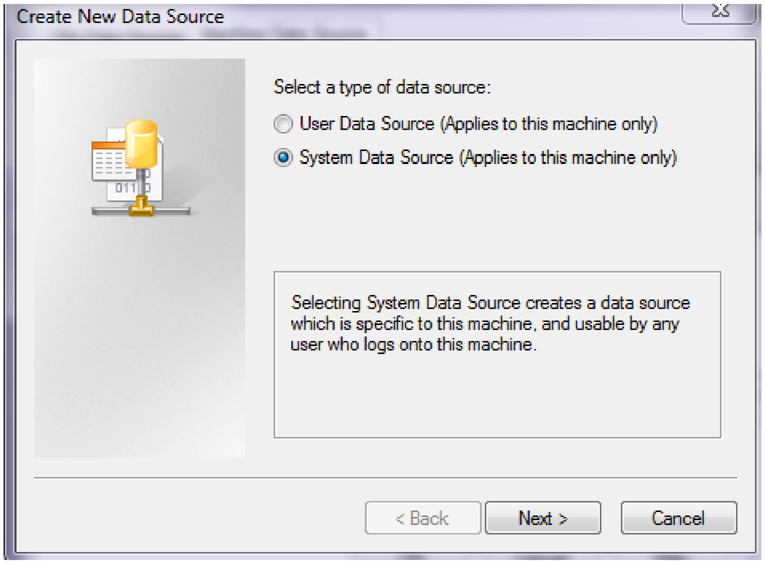
Once the export is started, there are two options – file data source, and machine data source. Since Machine data sources are specific to a machine, access is limited by the system or user. Servers are an ~~a~~example of a system wide machine, so if you are sharing data, this is the best option. You can select existing data sources such as the ones I have below, or create a new data source. For example, I would use existing data sources for multiple RVT files on one project. For a new project, select New:



Data Source Name	Type	Description
1477-2	System	Sample Project 1477-2
dBASE Files	User	
Excel Files	User	
MS Access Database	User	
Project Test	System	Project Test 1
RED LION sample	System	Red Lion Data
Revit Sample 1	System	Revit Test Sample

A Machine Data Source is specific to this machine, and cannot be shared. "User" data sources are specific to a user on this machine. "System" data sources can be used by all users on this machine, or by a system-wide service.

The sequence is a little backwards here, but the next dialog allows you to choose between the user and system datasource. If the data is to be accessed by multiple users, pick the System option.



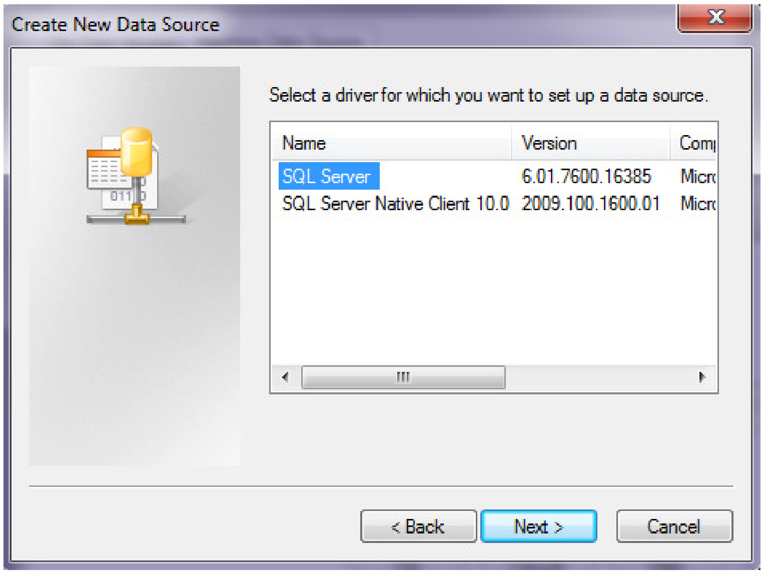
Select a type of data source:

○ User Data Source (Applies to this machine only)

● System Data Source (Applies to this machine only)

Selecting System Data Source creates a data source which is specific to this machine, and usable by any user who logs onto this machine.

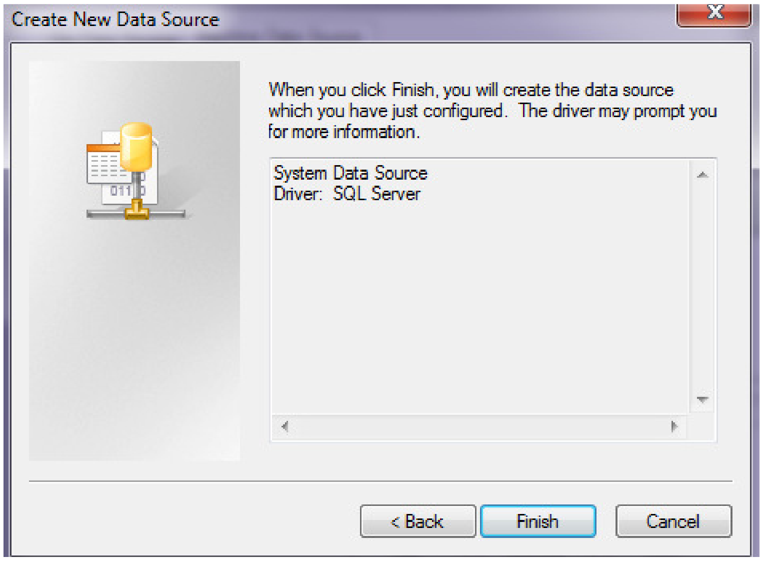
After selecting the source, select the database type – in our case, SQL Server:



Select a driver for which you want to set up a data source.

Name	Version	Comp
SQL Server	6.01.7600.16385	Micro
SQL Server Native Client 10.0	2009.100.1600.01	Micro

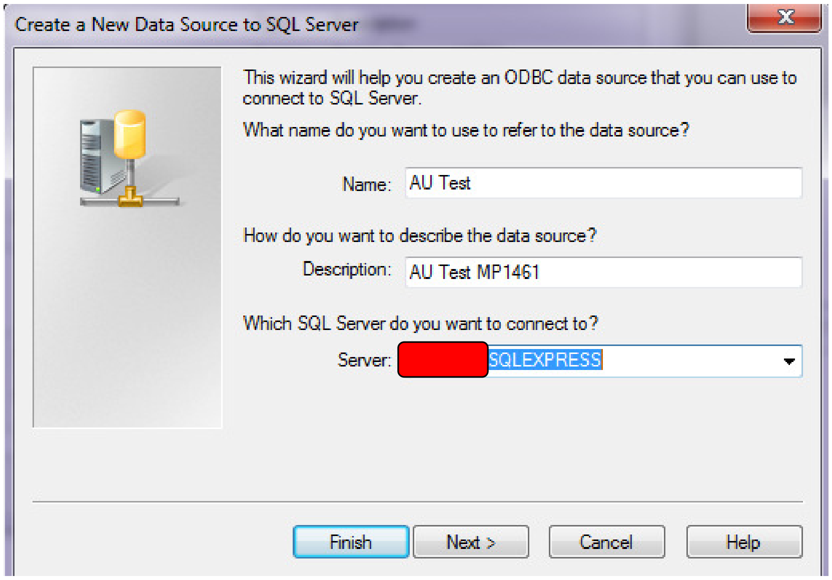
The last dialog in this step reviews the configuration – you can go back if needed to make changes. Otherwise, select Finish:



When you click Finish, you will create the data source which you have just configured. The driver may prompt you for more information.

System Data Source
Driver: SQL Server

Next up is the data source wizard, which first lets you add a name and ~~descrption~~description to the specific database you are creating. You can also select the location of the SQL server. In this case, I defined the SQL database to reside on my computer:



Create a New Data Source to SQL Server

This wizard will help you create an ODBC data source that you can use to connect to SQL Server.

What name do you want to use to refer to the data source?

Name: AU Test

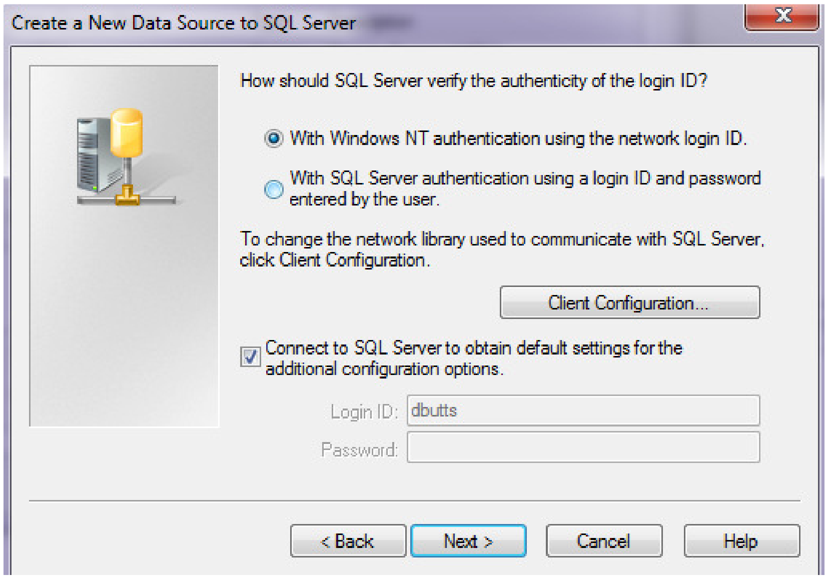
How do you want to describe the data source?

Description: AU Test MP1461

Which SQL Server do you want to connect to?

Server: [redacted]SQLEXPRESS

The following step lets you set up the login:



Create a New Data Source to SQL Server

How should SQL Server verify the authenticity of the login ID?

● With Windows NT authentication using the network login ID.

○ With SQL Server authentication using a login ID and password entered by the user.

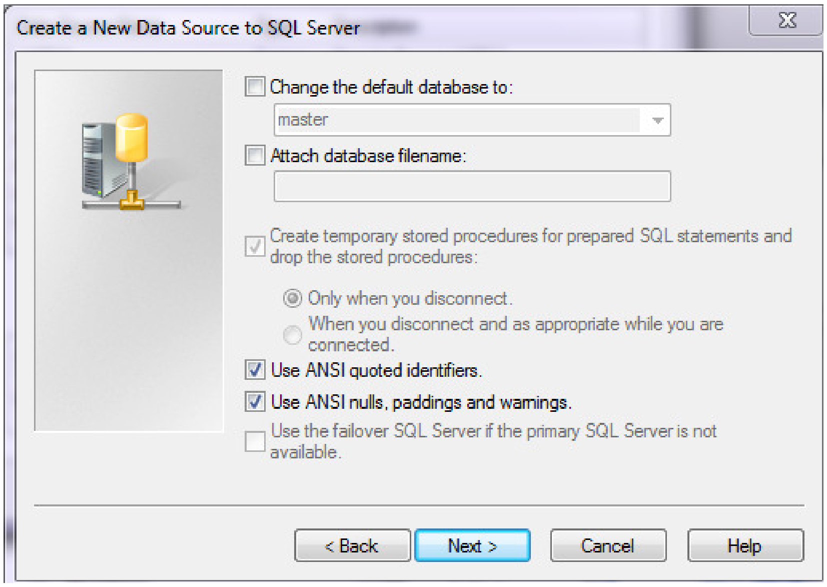
To change the network library used to communicate with SQL Server, click Client Configuration.

☑ Connect to SQL Server to obtain default settings for the additional configuration options.

Login ID: dbutts

Password:

The third step lets you change the default database, attach a database filename, and define how identifiers, nulls, paddings and warnings are used:



Create a New Data Source to SQL Server

□ Change the default database to:

master

□ Attach database filename:

☑ Create temporary stored procedures for prepared SQL statements and drop the stored procedures:

● Only when you disconnect.

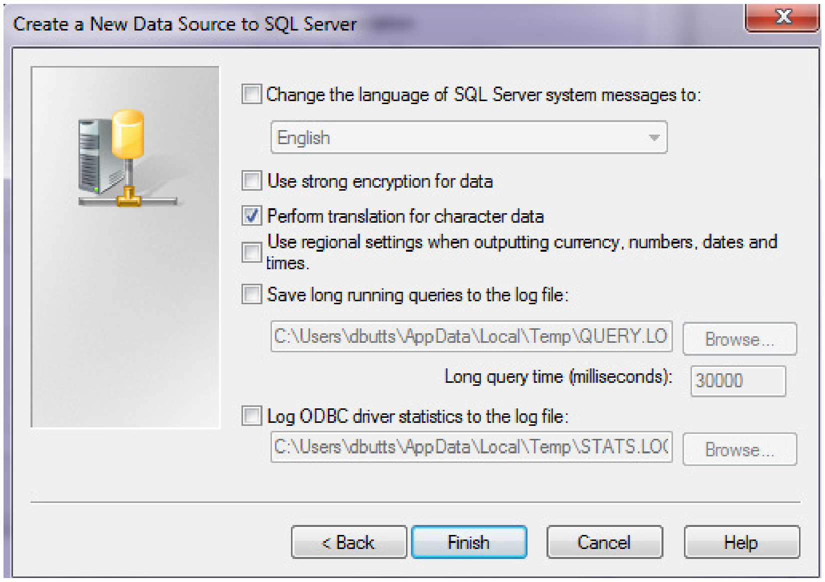
○ When you disconnect and as appropriate while you are connected.

☑ Use ANSI quoted identifiers.

☑ Use ANSI nulls, paddings and warnings.

□ Use the failover SQL Server if the primary SQL Server is not available.

And the last step lets you set the language, encryption strength, character translations, regional settings and log files:



Create a New Data Source to SQL Server

□ Change the language of SQL Server system messages to:

English

□ Use strong encryption for data

☑ Perform translation for character data

□ Use regional settings when outputting currency, numbers, dates and times.

□ Save long running queries to the log file:

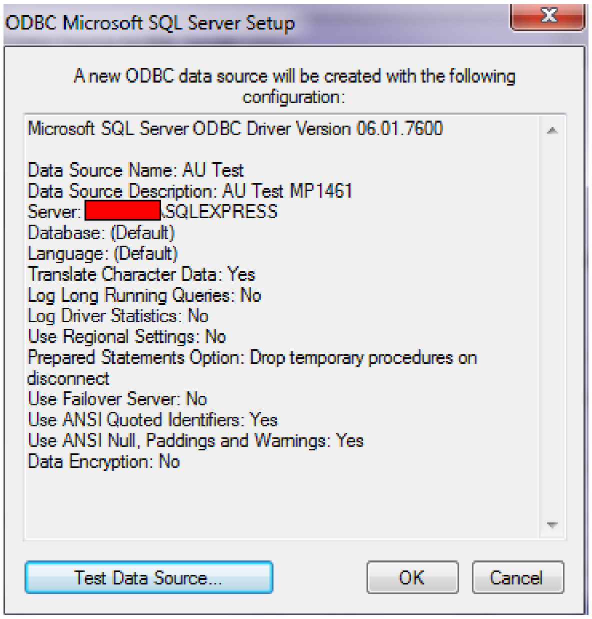
C:\Users\dbutts\AppData\Local\Temp\QUERY.LO

Long query time (milliseconds): 30000

□ Log ODBC driver statistics to the log file:

C:\Users\dbutts\AppData\Local\Temp\STATS.LO(

Once you've finish, you'll get a review dialog:

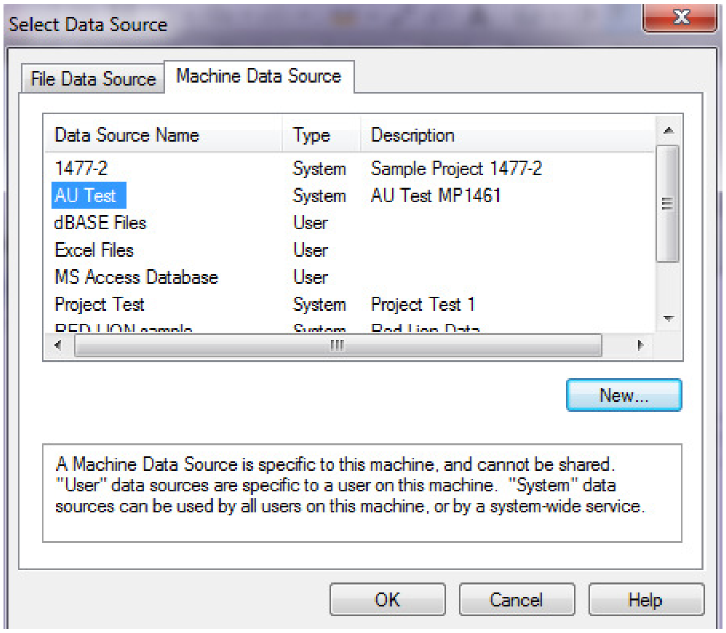


A new ODBC data source will be created with the following configuration:

Microsoft SQL Server ODBC Driver Version 06.01.7600

Data Source Name: AU Test
Data Source Description: AU Test MP1461
Server: [redacted]SQLEXPRESS
Database: (Default)
Language: (Default)
Translate Character Data: Yes
Log Long Running Queries: No
Log Driver Statistics: No
Use Regional Settings: No
Prepared Statements Option: Drop temporary procedures on disconnect
Use Failover Server: No
Use ANSI Quoted Identifiers: Yes
Use ANSI Null, Paddings and Warnings: Yes
Data Encryption: No

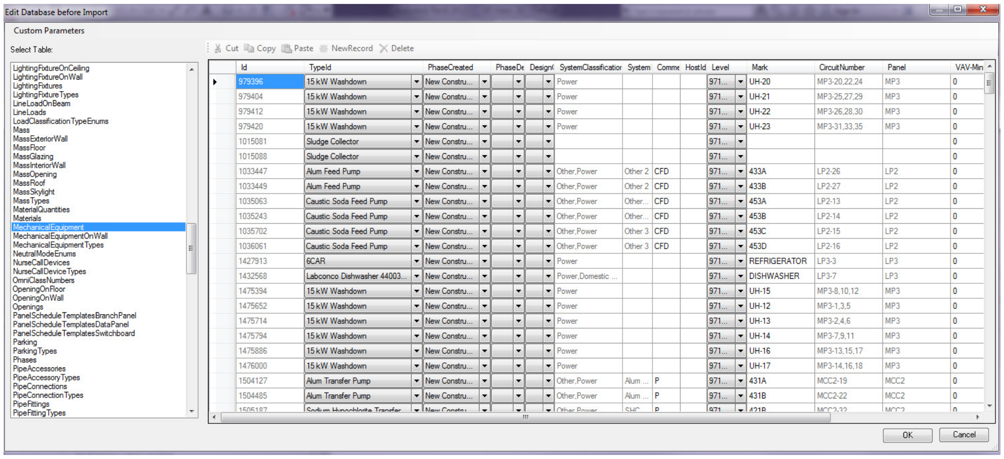
After selecting OK, the new database will appear in the machine data source list:



Data Source Name	Type	Description
1477-2	System	Sample Project 1477-2
AU Test	System	AU Test MP1461
dBASE Files	User	
Excel Files	User	
MS Access Database	User	
Project Test	System	Project Test 1
RED LION sample	System	Red Lion Data

A Machine Data Source is specific to this machine, and cannot be shared. "User" data sources are specific to a user on this machine. "System" data sources can be used by all users on this machine, or by a system-wide service.

Select OK, and you've created your link. Revit will export ALL data from the model – architectural, structural and MEP object data, for instance and types.



LightingFixtureOnCeiling
LightingFixtureOnWall
LightingFixtures
LightingFixtureTypes
LineLoadOnBeam
LineLoads
LoadClassificationTypeEnums
Mass
MassExteriorWall
MassFloor
MassGlazing
MassInteriorWall
MassOpening
MassRoof
MassSkylight
MassTypes
MaterialQuantities
Materials
MechanicalEquipment
MechanicalEquipmentOnWall
MechanicalEquipmentTypes
NeutralModeEnums
NurseCallDevices
NurseCallDeviceTypes
OmniClassNumbers
OpeningOnFloor
OpeningOnWall
Openings
PanelScheduleTemplatesBranchPanel
PanelScheduleTemplatesDataPanel
PanelScheduleTemplatesSwitchboard
Parking
ParkingTypes
Phases
PipeAccessories
PipeAccessoryTypes
PipeConnections
PipeConnectionTypes
PipeFittings
PipeFittingTypes

Id	TypeId	PhaseCreated	PhaseDe	Design(	SystemClassificatior	System	Comme	HostId	Level	Mark	CircuitNumber	Panel	VAV-Min
979396	15 kW Washdown	New Constru...			Power				971...	UH-20	MP3-20,22,24	MP3	0
979404	15 kW Washdown	New Constru...			Power				971...	UH-21	MP3-25,27,29	MP3	0
979412	15 kW Washdown	New Constru...			Power				971...	UH-22	MP3-26,28,30	MP3	0
979420	15 kW Washdown	New Constru...			Power				971...	UH-23	MP3-31,33,35	MP3	0
1015081	Sludge Collector	New Constru...							971...				0
1015088	Sludge Collector	New Constru...							971...				0
1033447	Alum Feed Pump	New Constru...			Other,Power	Other 2	CFD		971...	433A	LP2-26	LP2	0
1033449	Alum Feed Pump	New Constru...			Other,Power	Other 2	CFD		971...	433B	LP2-27	LP2	0
1035063	Caustic Soda Feed Pump	New Constru...			Other,Power	Other...	CFD		971...	453A	LP2-13	LP2	0
1035243	Caustic Soda Feed Pump	New Constru...			Other,Power	Other...	CFD		971...	453B	LP2-14	LP2	0
1035702	Caustic Soda Feed Pump	New Constru...			Other,Power	Other 3	CFD		971...	453C	LP2-15	LP2	0
1036061	Caustic Soda Feed Pump	New Constru...			Other,Power	Other 3	CFD		971...	453D	LP2-16	LP2	0
1427913	6CAR	New Constru...			Power				971...	REFRIGERATOR	LP3-3	LP3	0
1432568	Labconco Dishwasher 44003...	New Constru...			Power,Domestic ...				971...	DISHWASHER	LP3-7	LP3	0
1475394	15 kW Washdown	New Constru...			Power				971...	UH-15	MP3-8,10,12	MP3	0
1475652	15 kW Washdown	New Constru...			Power				971...	UH-12	MP3-1,3,5	MP3	0
1475714	15 kW Washdown	New Constru...			Power				971...	UH-13	MP3-2,4,6	MP3	0
1475794	15 kW Washdown	New Constru...			Power				971...	UH-14	MP3-7,9,11	MP3	0
1475886	15 kW Washdown	New Constru...			Power				971...	UH-16	MP3-13,15,17	MP3	0
1476000	15 kW Washdown	New Constru...			Power				971...	UH-17	MP3-14,16,18	MP3	0
1504127	Alum Transfer Pump	New Constru...			Other,Power	Alum ...	P		971...	431A	MCC2-19	MCC2	0
1504485	Alum Transfer Pump	New Constru...			Other,Power	Alum ...	P		971...	431B	MCC2-22	MCC2	0
1505187	Sodium Hypochlorite Transfer	New Constru...			Other,Power	SHC...	P		971...	421B	MCC2-32	MCC2	0

Revit runs through an initial export – once this is complete, you can use the edit and import features to review the data. One item I noticed right out of the box is that all data doesn't automatically flow from the project to the database, such as custom shared parameters

In order to share the data, it's not required to have the same names, but it is preferred. At certain points, we were looking at linking data by Revit element ID to AutoCAD handle, but found it wasn't necessary.

You can now go into the back of the house and do your programming to link the P&ID database to the Revit database.

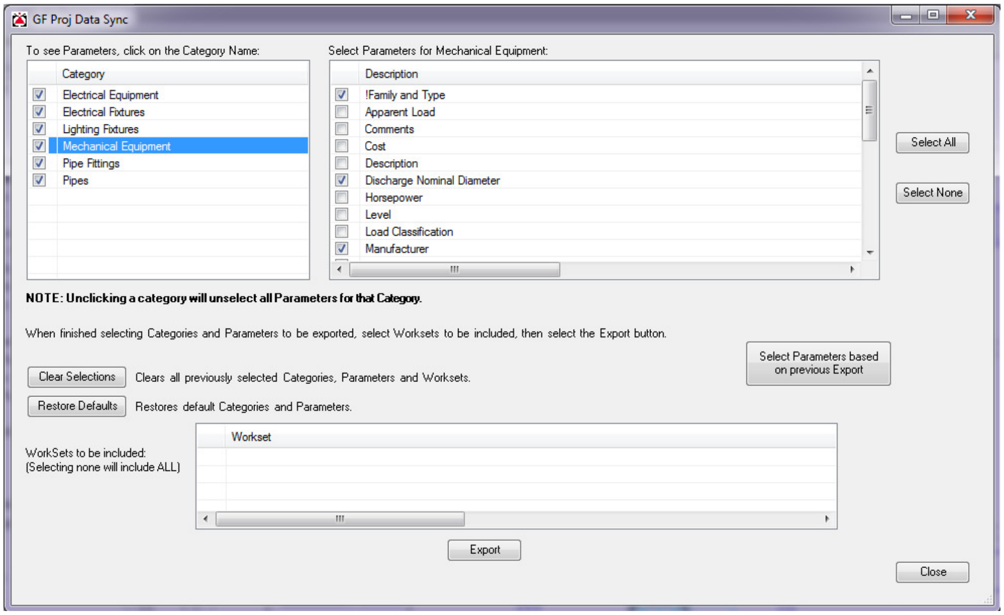
A Kindler, Simpler Data Solution

Our Project Data Sync project wasn't about exporting out all the data, but instead about the user having control over the specific data. In this case, we had a phase that addresses the export and import to Excel. Earlier, we referred to other applications, but decided to do our own in house for several reasons:

- Intellectual property control – by owning the application, we could modify and expand based on our needs and the client's needs
- Cost control – by using in house resources, we can develop parts of code that could be used in other applications, reducing their future development costs
- Portability – while developing the programming for Revit, the common code items could also be structured for other BIM applications we might be required to use, such as Bentley BIM applications.

- Keeping it simple – for the Excel portions, we could alter the interface and other user features without having to depend on external developers to do the work.

So we developed the application that allows us to generate linked Excel files, which could also be used with old-fashioned OLE links to share the data. The data can also be locked, so that items in the Excel file could not be changed, if they needed to be only changed in the Revit model. An example of this would be an electrical connector's values, or circuiting information.



To see Parameters, click on the Category Name:

	Category
☑	Electrical Equipment
☑	Electrical Fixtures
☑	Lighting Fixtures
☑	Mechanical Equipment
☑	Pipe Fittings
☑	Pipes

Select Parameters for Mechanical Equipment:

	Description
☑	!Family and Type
☐	Apparent Load
☐	Comments
☐	Cost
☐	Description
☑	Discharge Nominal Diameter
☐	Horsepower
☐	Level
☐	Load Classification
☑	Manufacturer

NOTE: Unclicking a category will unselect all Parameters for that Category.

When finished selecting Categories and Parameters to be exported, select Worksets to be included, then select the Export button.

Clear Selections Clears all previously selected Categories, Parameters and Worksets.

Restore Defaults Restores default Categories and Parameters.

WorkSets to be included:
(Selecting none will include ALL)

Workset

We made sure that several key features were included:

- The Ability to lock cells and the entire sheet
- Workset Export for sorting
- Saved settings so that the same group of parameters could be reused, and expedite updates to existing spreadsheets
- Warnings for excessive missing parameters, along with email notifications to BIM leads and project managers
- Verification for import/export data that can be easily documented

Formatted: List Paragraph, Bulleted + Level: 1 + Aligned at: 0.75" + Indent at: 1"

The add-in has been deployed to users on several projects, and the team has provided great input that should help improve the end result. We're also starting to see a good ROI, as more project teams are leveraging the data to create more coordinated document sets.

Formatted: Heading 1

Conclusion

So you've had a chance to learn how we approached customizing AutoCAD P&ID and Revit 2013 to share information in a more consistent manner, and cut down on errors and omissions. By working to minimize manual tasks and reduce repetitive steps, you can deliver a better quality project to the client – and really give a bump to your bottom line!

For more tips and trick, refer to my blog, The MEP CAD Engineer, at http://mep-cad.blogspot.com.

Thanks for attending!