

VBA - Project Prep

I have conducted several seminars on VBA over the past couple of months and in each I open with an interactive discussion intended to help the audience set some expectations. I want the audience to leave understanding how to outline the scope of a VBA development project and how to make a development project successful.

In my seminars I break these down into five questions, which I've decided to share with you here in this article. Each is designed to test the concept, or idea. These five questions will tell you whether or not your software will be implemented by end-users and embraced by management.

What do you want to build?
Does the solution already exist?
How do I find out if the solution already exists?
Who is the Program for?
How do I sell the project?

I often wonder why we repeat the same things over and over again when we know it's not the best approach. Maybe we do so because it's easy and sometimes it works. Perhaps we have trouble convincing management that we have a problem. Whatever the reason I know most of us have had similar experiences using MicroStation. It might go something like this:

You're the CAD lead for your discipline and there is a 50 % submittal due. At some time before the daily courier deadline, and by the way it's Friday, you notice that your entire package says 25% submittal instead of 50%. And, you notice that the string exists in every file instead of in a common reference file. Yikes!

What's the fix? Well when there's a deadline you just pop the string into the border file since every file references the same border, you open every file and delete the submittal stamp, re-plot and make 12 copies.

So you start, you make the change to the border. Yes that step was very easy. Now you begin opening each file – open, delete, fit, save settings, compress - repeat. 250 files later there's not much time left for plotting, not to mention you need 12 copies. Good thing you came in early today, too bad you'll have to deliver the drawings yourself.

You know there is a better way and that's why you are interested in MicroStation VBA.

MicroStation VBA is an extremely powerful tool that can make use of MDL code, macro code, windows functions, and of course visual basic code. Developing a VBA can eliminate many repetitive nightmares, but where's the best place to start?

In our example above we recognized the problem right away. We need a tool that will automatically process a set of design files removing the unwanted piece of text. Now, not

all problems have such clear solutions. So before we write any lines of code, we attempt to define the project by asking our questions.

Question 1 - What do you want to build?

The question, “What do you want to build”, forces you to clearly define the problem and the solution.

“We have to remove the same text string from every file, compress, fit, save settings, fence and re-plot. There are 250 files and it will take one person eight hours to complete the job. We can create a program in 2 hours that will automatically delete the string and submit the files for plotting. The program will take five minutes to run and the files will finish plotting on our high speed plotter in 3 hours.”

A more complicated program requires a more sophisticated statement possibly describing application features and expectations. By asking this question however, you start to realize the scope and effectiveness of the project. Consider what would have happened if our answer to the submittal problem had been an estimated development time of 8 hours. Unless you plan to make similar mistakes again the usefulness of this application is no longer clear.

When I conducted my first seminar at the New England MicroStation User Group in Portsmouth, New Hampshire I asked the attendees what they wanted to build, and one response stands out.

The problem required that the reference file dialog be opened and closed by a function key. Their workflow required constant interaction with the reference dialog and they decided that there would be significant productivity gains by being able to open, and close the reference file dialog using a function key.

On the surface the solution appeared simple enough. Just record the sequence of opening and closing the reference dialog using the recording features of VBA and then apply the appropriate key-ins to a function key. Although the open feature recorded, the close event did not. We ended up solving the problem with a combination of MDL and VBA all contained within the VBA application. Here’s the solution.

Using MDL in our VBA application we can access programming tools that are not available in Visual Basic. In this case we are talking about unloading the reference dialog box. Bentley recommends this method as a last resort, but I have found it to work extremely well.

Option Explicit

```
Declare Function mdlDialog_closeCommandQueue Lib "stdmdlbtin.dll" (ByVal dbP As Long) As Long
Declare Function mdlDialog_find Lib "stdmdlbtin.dll" (ByVal dialogId As Long, ByRef ownerId As Integer) As Long
```

```
Sub openRefDlg()
    CadInputQueue.SendCommand "MDL KEYIN REF DIALOG REFERENCE"
End Sub
Sub closeRefDlg()
```

```
'DIALOGID_ReferenceFile (-7) ::: this is from mdl\include\dlogids.h
mdlDialog_closeCommandQueue mdlDialog_find(-7, 0)
CommandState.StartDefaultCommand
End Sub
```

The complete application is contained in a single module. The interface is the keyboard, so a user form is not required.

Line 1 we have seen in previous articles. “Option Explicit” tells VBA that all variables in this module must be declared.

Lines 3 and 4 tell visual basic how to interpret the MDL functions. In other words, where to find the function, and how to define and pass variables.

Mr. John Gooding of Bentley posted this response regarding MDL in VBA, noting that this technique only works in MicroStation VBA.

“Yes, it is now possible to declare MicroStation API functions like you do with Win32 API functions. However, your Visual Basic program has to be running in MicroStation's address space. Of course, the VBA programs running under MicroStation's control do always run in MicroStation's address space. However, standalone VB programs and VBA programs running under the control of VBA in other applications cannot use these functions.”

Bentley has not provided a complete template library for use of MDL functions in VBA as of the writing of this article, but with a few rules you can declare most mdl functions within your modules.

1) Find the mdl function declaration by searching the MDL reference guide. Check the return type, if it is void then the function should be defined as a sub routine:

```
Declare Sub mdlMline_getInfo Lib "stdmdlbtin.dll" (...)
```

Functions are declared like the example shown above. Here's another look.

```
Declare Function mdlDialog_closeCommandQueue Lib "stdmdlbtin.dll" (...) as Long
```

2) Figure out how to pass the variables in each function. If the variable is a pointer, indicated by an asterisk next to type label, then in basic it should be preceded with a ByRef statement in the VBA declaration.

If on the other hand the variable is not a pointer (no asterisks) then the variable is passed as ByVal.

Searching the MDL reference for mdlDialog_find returns the following function declaration.

```

BoolInt    mdlDialog_find
(
int        dialogId,
void*      ownerMD
);

```

The type void is followed by an asterisks indicating that the associated variable, ownerMD, is a pointer and requires a ByRef designation.

3) There are exceptions to rule 2. Elements, and dialog box pointers for example need to be passed as long. There are a few examples on news.viecon.com in the v8.vba discussion group. If you don't find what you are looking for here, then post a new thread.

4) Finally, the variable types in Basic are not the same in MDL. For example, Integers in MDL must be passed as long. You can find a mapping of variable types in the MicroStation VBA Help under the topic heading, "Interacting with MDL". You can navigate the help file via the contents tab; Overview > Developing Code in VBA > Interacting with MDL.

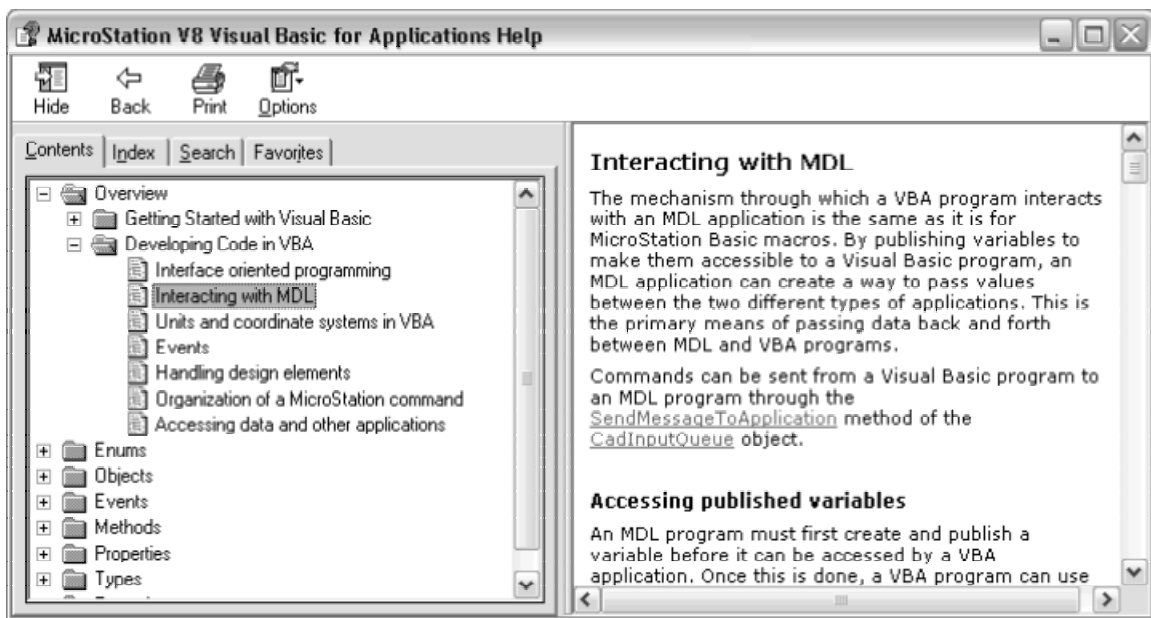


Figure 1: VBA Help Interacting with MDL

The rest of the code is just two function calls. The first is the open reference dialog sub routine created using the record features. The second routine calls the close request for the reference dialog using the MDL functions.

In this case the solution wasn't very difficult so we went ahead and created the program. A few minutes of programming and you're done. Now email the solution to everyone that has the same problem along with a help file so that they can set up the function keys. And be sure to tell the boss so that you can get a bonus for saving the company several hours of production time per year, per week, per day!

Question 2: Does the solution already exist?

In other words, is there a software application commercially available that does exactly what you want to do? If the answer is yes, then buy it. Don't waste money on software development if there is an off the shelf solution that will do the things you need to do.

Whether done in house or outsourced, developing a new software application will always cost more than buying a packaged solution. Any questions?

And, remove the negative. It is amazing how many reasons we can find for not purchasing software that will improve some part of our design workflow. Here are some examples.

My boss won't let me buy anything. My company will never do that. My company doesn't listen to me. I can never get time for development. Or some other statement along the same vein... Change your attitude, and change your mind because the problem may not be where you think it is... Then review this article.

And then there is the ultimate, "We can't buy it because it's too expensive."

In this case you don't have a problem. Think about it. If it is too expensive to buy the software then it will be much more costly to develop it yourselves.

Question 3: How do I find out if the solution already exists?

Currently the best search tool available is www.google.com. The search tool indexes almost 2 billion web pages and a search on the word MicroStation returned about 178,000 pages. Check out comp.cad.microstation, also available at www.google.com.

If you still haven't found solution, then post a message on one of the active news groups, like CCM - comp.cad.microstation, or bentley.microstation.v8.vba on news.viecon.com.

Extend your research further by visiting vendor sites, ask your MicroStation resellers, and ask your colleagues at user group meetings.

Question 4: Who is the Program for?

Will you be the only user? Or, will everyone else in your office be able to benefit from this project?

If you are the only one who will use the program then ask another question, "Will the time and quality improvements pay for the cost of development? Or, will you be able to do something, a calculation, a design that you haven't been able to do before, something

that will land you new customers and projects. If it doesn't, maybe this isn't the correct solution for your problem.

On the other hand, the VBA application you propose may decrease the time it takes to draw HVAC duct, and there are 10 HVAC designers in your office. Drawing elbows you estimate takes five minutes and you can reduce that to almost zero time. You have just paid for a couple of salaries.

There's a pattern here and it is directly related to the return on investment. You wouldn't pay for MicroStation if you didn't make or save money using it. And the same applies to new software you want to develop.

To summarize – your application must improve quality, save money, or make money – or all of the above.

Question 5: How do I sell the project?

What? I'm not a sales person I am a designer, an architect, an engineer. Well, yes you are. However, while your professional label indicates your primary role, you have a secondary responsibility to sell the design you're working on and your outstanding abilities that show your boss you deserve a raise.

But seriously, if you have done your homework and answered the first four questions then this step is easy. You know what the problem is, you understand the solution, and you also know how much money it will save and/or make.

Here are a couple of strategies that will help in the internal sales process.

First, make a features list. These are the solutions to the problem(s) you are solving with your application. A ten point bullet item list works well. Keep it simple. Interface development is easy in VBA so go ahead and layout what you think the interface will look like. (Use Alt-Print Screen to capture any "Active" windows dialog. Then use ctrl-v to paste the dialog into a word document.)

Now, go tell everyone about it. Ask their opinion. Modify your feature sheet to match their feedback. Ask them if they will use it. Finally, you have everything you need to approach the boss and you will be programming in no time. Good luck!