



Las Vegas, Nevada, December 3–6, 2002

Speaker Names: Brian M. Wildt (wildtb@weasler.com)
Phil A. Leverault
Dave Espinosa-Aguilar

Course Title: Developing a FM Automation Program
for AutoCAD - A Real World Example!

Course ID: FM23-1

Course Description:

In this course we'll show you how we developed an automation program from concept to implementation. We'll discuss how to get started, what things to keep in mind as the concept gets off the ground and a few problems that you may run into. We know there are many companies out there looking for ways to save time and money in their drafting departments. We want to share our experience with you and show you just what can be accomplished and maybe give you some ideas to get started with a program of your own.

Hi, my name is Brian Wildt. I am a CAD Manager for Weasler Engineering Inc. in West Bend Wisconsin. Weasler designs and manufactures drivelines (PTO shafts) for the agricultural and lawn & garden markets. Our products are completely custom to the applications that they are used. They do use many common components however the combination of those components is what makes each driveline custom. Because of the nature of our product, we have developed numerous LISP routines and toolbars for automating many of our daily repetitive tasks. Over the years they have proven to save much time per drawing. However, we are in need of more timesaving, as customers demand shorter lead-times and lower costs.

About 2 years ago the executives at Weasler wanted management to find ways to reduce cost in our quoting process and take time out of the drafting lead-time. At that time, it would take roughly 1-3 weeks for a complete quote with drawings. They wanted that time to be reduced to 1-3 days for any type of quote. This posed a problem for us in engineering as at that time it took approximately 1-3 days to complete a driveline assembly drawing from design to customer quote drawings.

Simplifying our drawings would help but we knew that the only way to really save time was automation. Our president came back from a trip with a copy of an automation program that used pictures of components and assembled them in order to make the finished product. This program was from a company that had standard products and we are a custom product company.

Taking the concept of that program we found a way to use our existing drawings adding some extended entity data and use them directly in our assembly drawings as external references.

The following is an example of the outline we used to develop our application. It is basically a brainstorm of questions and information that can be used as a guide to develop your own program. This is just a guideline. You may have more or less requirements for your program.

PROGRAM DEVELOPMENT OUTLINE:

- **Scope of the program:**
 - This is where you try to get your ideas written for what you want your program to do. How long should it work? Who will use it? Etc. Nothing is out of the question here. This is brainstorming. Come up with all the crazy ideas that you would like to see the program do. The next section is where you can pare down the list to things that make sense.
 - What do you want it to do?
 - Build a drawing
 - Assembly
 - Sub-Assembly
 - Detail
 - Build from right to left
 - Create a Bill of material
 - Balloons
 - Parts lists
 - Dimensions
 - Titleblock
 - Scale
 - XREF Drawings
 - Insert Drawings
 - How long should/can this program function without updates?
 - In terms of years and releases of AutoCAD
 - How do you want it to work?
 - Within AutoCAD
 - Outside of AutoCAD

- Who will be using the program?
 - Drafters
 - Engineers
 - Sales
- Is there existing data to use?
 - MRP
 - Part Databases
 - Quoting Documents
 - Bills of Material
- Where should this program info be saved?
 - What program/software should be used?
 - Database
 - Spreadsheet
 - Other
- How can we get that info into AutoCAD?
 - VB
 - VBA
 - LISP
- **Feasibility Study -- Is it possible to do what you need?**
 - There are many places to get help in the feasibility process. The AUGI guilds are great. However, you may consider the help of someone builds these types of applications on a regular basis. That is generally the best and fastest way to get answers to the "what if" and "how do we do this" questions. Not only that but it is possible that they may have done what you are trying to do and can give you some help in your process. You may find that some of your wild ideas are not too difficult to include.
 - Try to think of all possible drawing configurations
 - Assemblies
 - Varieties of assemblies
 - Sub-assemblies
 - Varieties of sub-assemblies
 - Details
 - Varieties of details

- What drawing configurations lend themselves to be automated?
 - Focus on those types
- Is there a certain set of events that need to happen?
 - Gather information to one location
 - Import info to AutoCAD
 - Build Drawing
- Document the process of building the drawing
 - This will help find any potential problems
- Step through the process with actual drawings
 - Make several drawings
 - Try to duplicate process manually
 - This will help find any potential problems
- Decide on the drawing format and program process
 - Order of production
 - Where information comes from
 - How does the information get to AutoCAD
 - What type drawings are we going to build
 - Drawing layout
 - Xrefs
 - Parts List Methods
 - Ballooning Methods
 - Paper Space and Model Space Interface
 - Database structure
 - Field naming and Implementation
 - Dialog Box Design
- What to do with existing drawings
 - Modify to use in new format
 - Leave drawings as they are and start with new drawings in new format
- Develop Template Drawings
 - Set-up drawings that have all layers, dimstyles and textstyles set

- **Program Architecture**

- This is where the decisions are made about the program itself. How will you interface with it? Who will maintain it? Where will it run? What different methods can be used? These are all things that will determine how the program works and what language it gets written in. Decisions here are set in stone so that code writing can begin.
- Decide if any utilities need to be made for existing drawings
 - Adding extended entity data
 - Process for strapping extended entity data
 - Drawing set-up

- **Develop a time line**

- This will help to keep track of all the steps that need to be completed
- Set-up a chart to track progress
 - This will help keep the time line moving

- **Develop a productivity chart**

- You'll need this for justification of your program. Be sure to include all estimates of time and cost of your current process. Add an estimate of how much time you will save and translate that to cost. Include the time it will take for the return on your investment cost. This will be very helpful in getting the buy-in from those that will give the OK to go ahead. Once the program is in place for awhile track the actual savings time and compare that to your estimates for an actual payback time.

Once the program architecture is complete coding can begin. You should keep in mind that any changes to your program spec at this point will add time to the coding. Try to have the spec nailed down and set in stone before you start. This will save any future headaches. There will always be things that have been overlooked so make sure to add a little to your time line for that.

Some things to keep in mind...

If you hire a programmer to do this work for you be sure to keep open communication with them. The last thing you need is some miscommunication to add time to your project. Keep in mind the more time it takes to complete the project the more it will cost. Most consultants will give you a quote on their time for coding. You also must keep in mind though that if you make any changes once the coding process has begun that will undoubtedly add cost to their estimate.

As the programmer gets code written ask them to share it with you so that you can test it as it is written. This will save time in the beta testing process. That is where much time is spent trying to make the program "break". The earlier you get the code the earlier problems can be resolved.

There are many problems that can pop up while your program is being built. Most of them are oversights from the Feasibility stage. Things that you didn't realize that you needed or because of the way the code needs to be written some things may need to be changed or eliminated in your program. A good programmer will figure for these kinds of things in their quote for coding. But in any case make sure that they have allowed for it.

As our program was being coded we realized several things that we did not think of earlier. Most of them were related to how the drawings were shown. Which dimensions are needed, how each piece is related to the others. Things of that nature. The order of how the part are brought in was another issue. Inevitably these things happen.

Because many times there isn't an existing program that you can buy there may be certain issues with AutoCAD or Excell or Access that haven't been seen before. These are the scary things that can pop up. We ran into this situation. The way we wanted to use xrefs was not supported by AutoCAD. In fact it was a known flaw with Xrefs. We actually had to contact a programmer at Autodesk to get the workaround because no-one that we could ask knew how to get around it.

The problem specifically was as the program assembled the drawing with xrefs it would fail if there was another xref with the same name in the drawing. We have some products that use multiple identical components. This posed a problem for us. We needed that capability. The time consuming solution was to rename the xrefs as they were brought in.

Barring any of the above issues Beta testing should go fine. Beta testing is the time when the code is basically done and now it's time to figure out what makes it tick, or not tick. Once those issues are resolved you have a working program.

The reason I bring those issues up is that it is very important to keep these things in mind. They are the things that mostly can be avoided if you spend a little more time in the scope and feasibility stages.

Hopefully you can get enough from this to get started with your own program. It is a long process. From the time we started with the concept to the time we had a program implemented was about 2 1/2 years. Our program is fairly complex. Of course there was the usual maintenance of drawings and everyday business that doesn't go away. So in actuality it didn't take that long.

Thanks for taking the time today. I hope this information will be valuable.

Brian M. Wildt
CAD Applications Administrator
Weasler Engineering, INC.
7801 HWY 45 North
West Bend, WI 53095
262-338-5494
wildtb@weasler.com
www.weasler.com