Programming is No Longer Back to Basics

Donald F. Schneider

Stratus Engineering, Inc.
PMB 339
2951 Marina Bay Drive #130
League City, Texas 77573
(281) 335-7138
Fax: (281) 335-8116
e-mail: dfsstratus@aol.com

In the early days, less than 10 years ago but several generations in the computer business, writing a program on your desktop computer to take care of the odd engineering calculation was tedious, but not difficult. Most major Universities assured through many curriculums that you at least had the opportunity to learn the skills necessary to write simple code. Operating system vendors made it easy by including with every system a version of the BASIC language and a line-editor to create the code.

Those were the days of :

```
Print "Please type the letter of the option you wish to select"
Print ""
Print "A - Flow in lbs/hr"
Print "B - Flow in gpm"
Print "C - Flow in ft3/min"
Print ""
Input "Hit return after entering ( or type Q to quit)", LETR$
```

Compilers for other languages were also available, and worked in much the same way. The tools were readily available when you needed them. It wasn't unusual to try your hand at programming to solve a problem, or perhaps write a simple game.

But, as the personal computer evolved, the expectations of the user, and amount of work required to meet them, greatly increased. Writing your own software started to go the way of working on your own car: it's complicated enough that you let someone else do it! Just as computer programming began getting too complex even for computer programmers, tools were developed to ease the way. These tools can be used by those with very little computer knowledge with great success.

Programming for You & Your Job

Today's software provides an enormous reservoir of application capability and flexibility to help you work. Sophisticated word processor, spreadsheet, presentation graphics, drafting, database, and many other application types are available. However, even with the great number of software tools available, you may find opportunities to apply computer power where existing applications are inadequate or too costly. Or you might utilize a piece of software more fully by employing its built-in programming capabilities.

Once you begin to consider the possibilities, you may find the potential to improve your productivitiy through writing your own programs surrounds you. Examples of areas for which you might consider writing your own program include:

- Repetitive computer tasks such as editing, searches
- Unusual or tedious calculations
- Computer techniques which you'd like to distribute to others
- Data entry, evaluation, reivew, or organization
- Job or task training and documentation

After you have seen the need for a custom program of your own construction, the next step begins by using a specialized type of software.

Software for Software

Ironically, or perhaps fittingly, in order to write a program in the graphical environment, you *need* a software package. You need a software development program. Typically these applications consist of editors and pre-built tools to step you through the development of your program. Of course you have to provide the idea, but the software development program provides the framework. A whole new generation of languages has been created to interface with the modern computer. Languages are now development systems that support the operating system along with the program. Visual BASIC, Visual C++, even Visual COBOL have been created.

These applications bring programming for the graphical environment to the masses. Before they were developed, writing even the simplest 'Visual' code was an ordeal. Of course the hard part was not manipulating the information. The hard part was generating the graphical interface for the user. The Visual languages have automated much of this tedium. These software development packages are priceless. In summary, they provide a set of standard methods and objects for creating a display and its underlying code structure. You don't have to create a drop down or check box - that's already done. You simply have to tell it what to do. Figure 1 illustrates a screen that you might construct and typical objects that available for you to add to your program.

Spreadsheets & Databases

Spreadsheets and databases have evolved into program development systems all their own. These environments provide quick ad-hoc methods for program development. They often use languages that are also available as stand-alone development tools. In many you can create graphical objects and dialog boxes for user interaction. You can also use a programming language, not just equations in a cell or a macro language, to make calculations or manipulate information. Essentially, you write your own program within a program. The power and flexibility available to the user is tremendous.

Spreadsheet and database data entry and analysis can be greatly improved by providing your own custom software within the application to enter or examine the information. Even something as simple as programming a data entry form can reduce errors, provide a logical entry format, and present a display that is much easier with which to interface.

Call the Exterminator

Programming bugs are the bane of users and programmers alike. With the growing complexity of PC systems, it is a wonder they work at all. Is it the hardware, the hardware driver, the software, or the operating system that has caused the unexpected glitch? Indeed, one side benefit of writing your own programs is that you will, by necessity, learn more about the machine's hardware and software.

One of the most difficult potential 'bug' areas to manage is operating system limitations. Sometimes as a result of our increasing demands on the interface, the system runs out of room. Predicting this is difficult. Rigorous testing of your program under numerous situations is the best hope for reducing the risk of bumping against operating system, and other, limitations. Newer operating systems are becoming more robust, but they are also becoming more complex. True multitasking environments allow programs to branch during execution thereby performing more than one operation 'simultaneously'. This desirable capability also makes system response non-linear because the same action, done in the same sequence, may not cause the same electrons to move the same direction at the same time within the machine. You can see how this might complicate trouble-shooting. Perhaps the machines are becoming more human after all.

Much like beauty, many bugs are in the eye of the user. One user's bug is a programmer's intended result. However, even for problems we all agree are code errors, we seem to have acquiesced to a culture of accepting software that includes sometimes startling glitches. Funny how our dollars aren't defective when we pay for the product... Program bugs that are bad enough when they affect your word processor, become intolerable when an engineering calculation is involved. We simply cannot afford to arrive at and promulgate incorrect answers. Computer programs that involve scientific analysis are often extremely complex. Even thorough testing may not identify unusual scenarios that render the computer method invalid. That is why the normal systems of checks that are typical for hand calculations must remain for computer generated data. In addition, documenting the basis of the method used, much like you do for hand calculations, helps avoid pitfalls. A result is only valid within the context of its basis. Again, the information you provide in program comments or on-line help may not cover all cases, but it might remind you in the future why you did something, or why you shouldn't have.

Help!

There are two primary kinds of programming help: the kind you as a programmer need, and the kind the user needs. Let's talk about the programmer first.

Going on-line is one of the best resources for those needing programming help. This is increasingly true for all computer related software and hardware. On-line services and the Internet have sites where programmers and programming system vendors share information and updates. You may have stumbled upon a bug that is well known with a simple solution. Consulting an on-line group may provide this information with the least difficulty. Or, you may see someone's comment asking about a technique that you have mastered. Lending a hand is rewarding and increases your programming confidence. Additionally, software vendors of all types routinely post bug patches and upgrades on their corporate electronic bulletin boards or on the Internet or other on-line service. Downloading these files keeps your software up to date. However, be sure to save a copy of the old version of any upgrade files you install. The new software may not be

compatible with your particular program, or the intended upgrade may actually downgrade performance.

Besides on-line programming help, there are numerous books, videos, and CD-ROM's covering various languages and techniques. Most Visual program development packages come with substantial support documentation. Give them a try first before deciding what other help you need.

To help the user, help systems built into your program are a basic requirement of modern software. Figures 2 & 3 illustrate typical help file screen features. Most help files are at least an adjunct to an operating manual, if not a verbatim copy. Even if you write a small program for your own use, you may want to build a help file to store information about the software. If the program includes engineering calculations, a help file is a handy place to store assumptions, equations, and references. Like your code, help files can be printed for documentation purposes. Note, however, that a whole genre of software has cropped up for use in creating help files. The good old days are still gone: the tools needed to build a help file are usually not included with modern operating systems. Even the software development system you purchase may not include adequate help file authoring tools. But, creating a simple one is fairly easy and adds a lot of value to a piece of software.

In fact, help files are themselves programs that can stand alone. Help files can appear as icons on a desktop. Help files possess rudimentary search capabilities and can support information distribution, decision making, and training among other things. Recent additions to the power of the help file include the ability to launch programs and open documents, make full help file text searches, and allow multi-level keyword searches. PC based help files offer a powerful opportunity for any individual or organization to more efficiently use their computer information resource.

Rewards

For those that enjoy it, programming continues to be a rewarding experience. It may not be art, but it is creation, and it is yours. Today's computer systems have added a great deal of complexity to the work of programming. But that complexity pays dividends in added user power and flexibility. Staying in step with modern programming techniques keeps you up-to-date with hardware and its limitations which will make you popular around the office and help you keep your personal machine operating smoothly. It also allows you to utilize the programming functions available in common software packages such as spreadsheets and database engines. Even if you never find a reason to write a program for yourself, the computer revolution continues to push these machines deeper into our lives. It can only help to know something about how they do their amazing things while we watch.

Author biography:

Donald F. Schneider is a consultant with Stratus Engineering, Inc., of Houston, Texas. Previously he was a senior engineer for Stone & Webster Engineering, and worked as an operating and project engineer for Shell Oil Co. Don holds a BS in Chemical Engineering from the University of Missouri-Rolla and an MS degree in Chemical Engineering from Texas A&M University. He is a registered Professional Engineer in Texas. Stratus has recently completed development of a Windows based set of engineering tools.

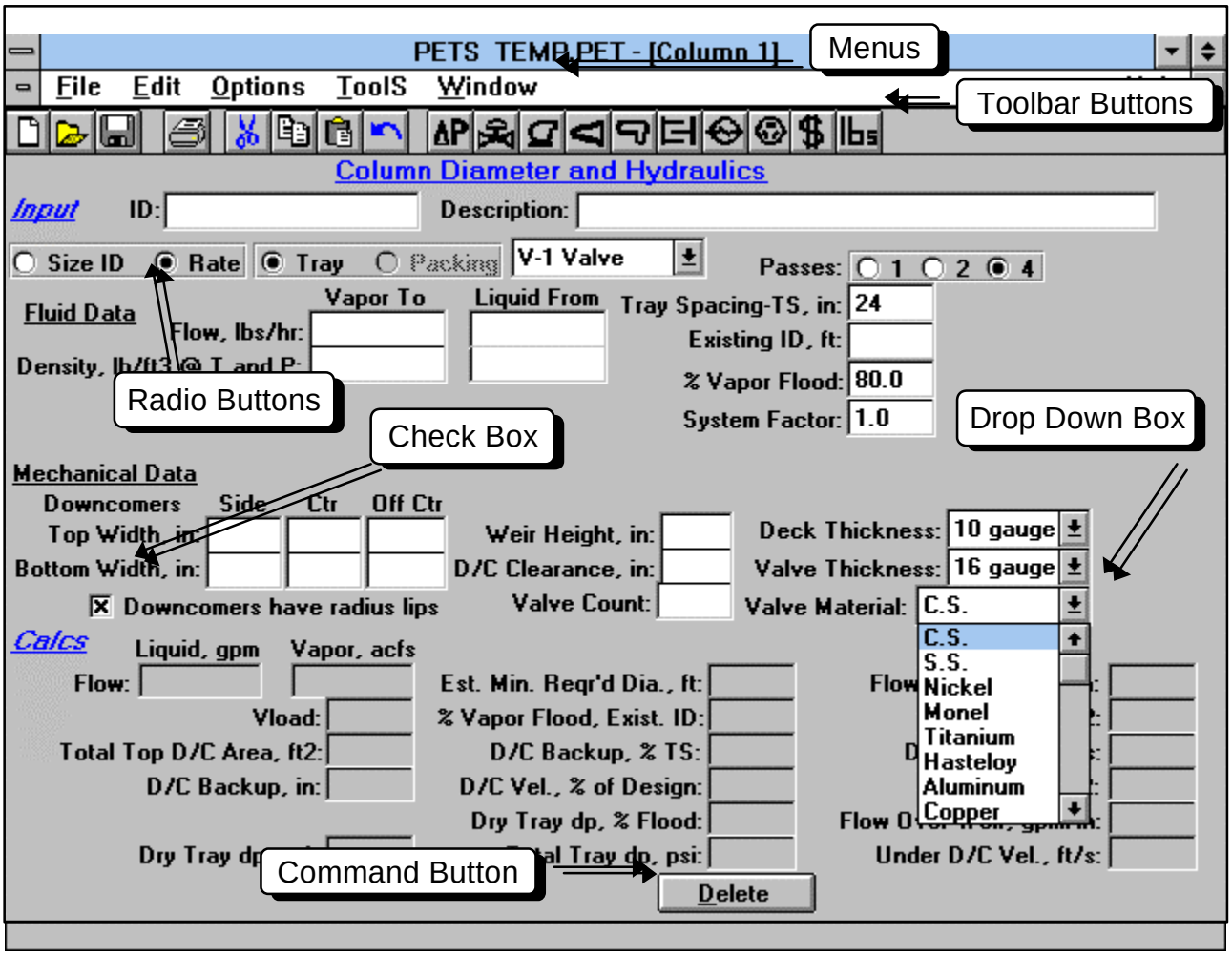

Figure 1 - Programming Objects

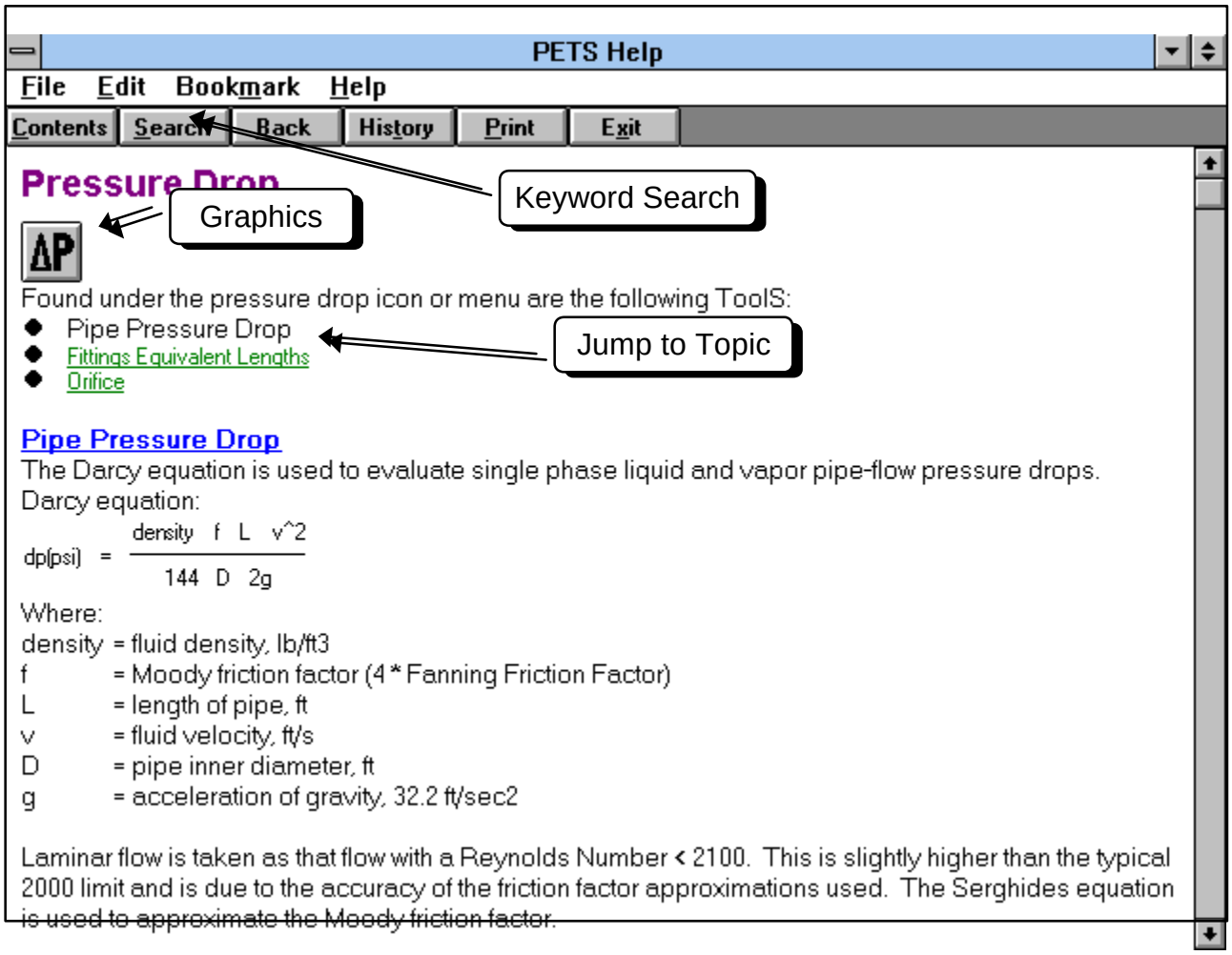

Figure 2 - Help Screen Example

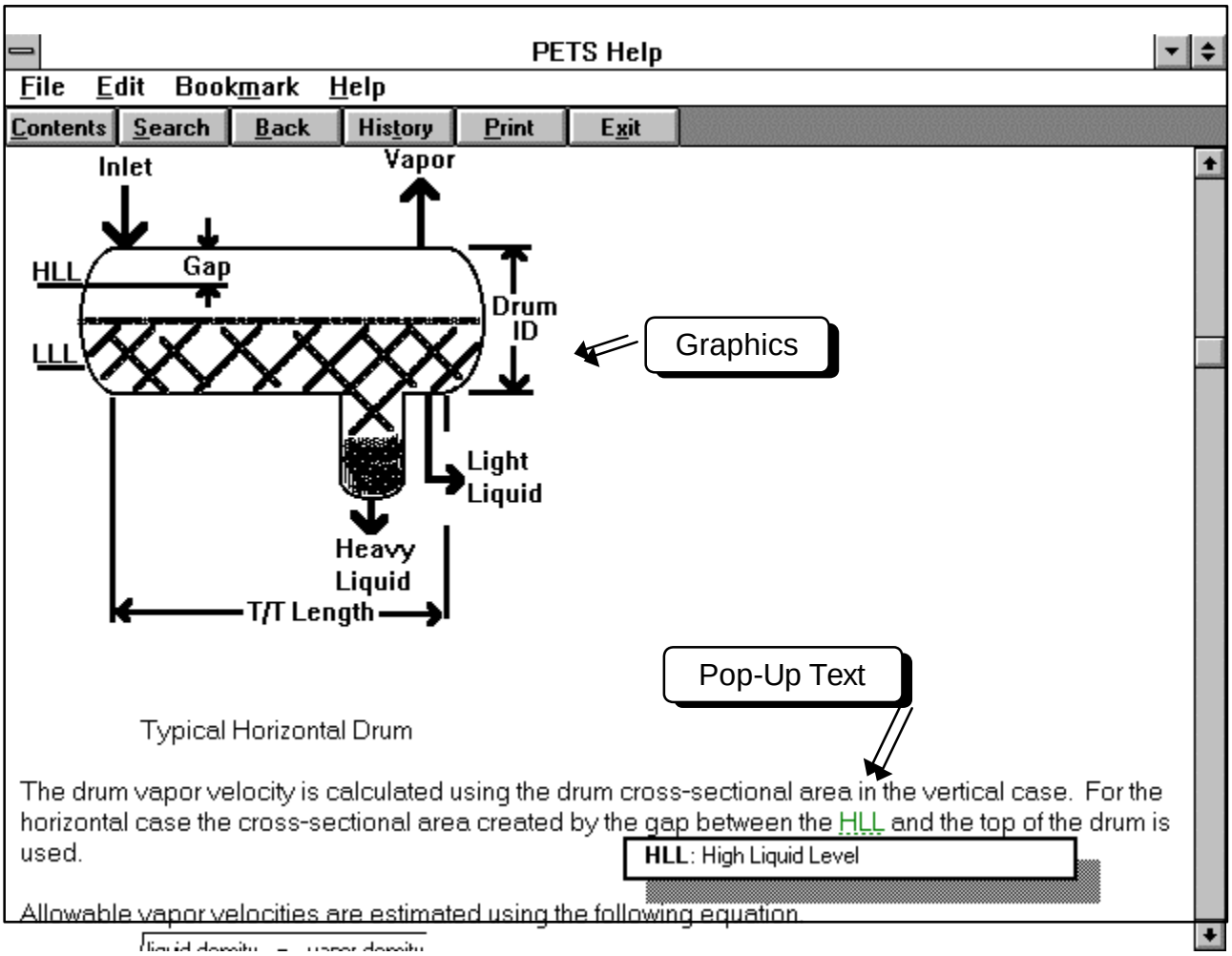

Figure 3 - Help Screen Example