The M Experience

by Chris Bonnici

As far as M is concerned, I still consider myself a baby. I've just turned six. While I've been involved with M for a short period of time, I am in my late teens with regard to the computing industry in general. Before M, I was earning a living through COBOL (banking) and PASCAL and C (education). I currently use M in the insurance arena.

When I moved from COBOL to PASCAL and C, I was already familiar with all three languages; PASCAL was the programming language of choice to write DOS programs, and C, although not on my I-love-it list, was necessary when studying most of the freely available source code. The move to the M environment was different. I can conscientiously say that I do not recall ever having heard about this programming language in my undergraduate or graduate studies (if it was mentioned, it could be there weren't sufficient credits tied to the topic!). Despite this, moving to M did not bother me all that much.

The Experience

As my move to a new job using M was to take place soon, I could not prepare myself as well as I would have liked. The easiest solution was to get someone already involved in M programming. I was lucky to find out about this great "M guru." So, I called him. After a hello that gave me the impression I was wasting his precious time, I introduced myself and explained as briefly as possible my situation. He lit up. No problem, a meeting was set up during his one hour lunch break the following day without my having said anything. Calling this individual an M guru was putting it mildly. Not having known each other for more than two minutes, he was telling me how fantastic the language was and how easy it was to write in it and how it had built-in multitasking/multiuser capabilities and how. . . . I thought as I "yes'd" and "sure'd" and "uh-hu'd" into the mouth piece every so often, that this person was a waste in development. He should go into marketing. I hadn't even mentioned (or hinted) that I would be transacting with his company. "I like PASCAL, but I don't get excited at the thought of it," I said as I hung up, "no wonder they call him a guru, he's brainwashed."

When I went to meet my contact, he came to the reception area and guided me to his office where he offered me a cup of coffee and sat me down. "You're new to M?" "Yes," I said. "What do you want me to show you?" he asked. "Write a program that calculates the volume of a box; no screen formatting or any extra code, just three inputs and one output," I said. "No problem," he responded. In what seemed like ten seconds later, the program was up and running. "That's efficient," I said "May I see the source?"

"My God" summarizes my first impressions of M. I could see the H=, L=, B= and V=. Everything was written on one line. I asked him if this language had to be written in paragraphs. "No, you can split up commands as with BASIC," he replied. Upon inquiring as to why he wrote everything on one line, he said "because it's quite normal in M to group statements together." Among the first things I did when introducing a new group of students to PASCAL was to provide a similar program neatly laid out and then tell them to join everything up and run the program. This allowed me to dive into the topics related to code readability and to adopting and maintaining certain programming styles such as indentation. I would have definitely failed a student if he or she had submitted code written in paragraphs.

I asked my M acquaintance to load a regular program. He called up the program into his line editor and dumped a screen full of code. I asked for a dry run of the entire piece and discovered that G meant GOTO and that it was in great demand. I thanked my acquaintance and headed for home.

I am not the worrying type. I have always considered worrying to be an unnecessary use of resources that could otherwise be put to use actually solving the problem causing the discomfort. Driving home, I started analyzing what I had just experienced. I am leaving the comfort of a great editor with integrated debugging to end up in a language for two-finger typists. I was going to earn a living writing code that lacked any notion of style, structure or beauty. Debugging six-month-old code was going to be a nightmare.

Roll Up the Sleeves

I eventually bought a book on M and started work. As I progressed, I discovered that it wasn't all that bad after all. I had opted for the wrong solution to the problem. As I progressed through various texts but especially through coding, I discovered that M was a fantastic language. The fact that both primary and secondary storage can be addressed in a similar fashion is really convenient. M looks at data as it really is; pieces of information held inside a computer. It simply defines its long term duration. No other language achieves this so efficiently.

Our implementation today has 64 users. It runs on a Pentium server with dumb terminals or terminal emulation software running on PCs. No one has ever complained of slow response time. With some of today's DBMS's you would need a Pentium on each desk to make the software work acceptably. M is so efficient! It is one of the only languages today capable of making almost every cent invested in hardware work for you rather than being converted into overhead cycles.

Once one delves into the core of the language, the content is even better than the topping. The command set is powerful. I feel that the list of commands provided is aimed at making the program productive. Certain routine tasks that would necessitate additional coding in other languages are available in M. The fact that the data looks the same regardless of where it lies, reduces the commands necessary to achieve a task without diminishing flexibility. This is a definite plus.

The ability to use procedures and functions and put them in libraries together with looping constructs, allows one to write code in a structured and readable manner. The fact that a language supports such features means that a good programmer will aim to use them. A well-written program should be documented such that another programmer can easily read, understand, and modify it. This can be arrived at through the use of inline and external documentation and a lesser use of GOTO statements, limiting each to a single, short span jump and clear programming.

While initially I criticized the act of putting multiple statements on one line, after having actively programmed in M, I've come to view this in a more positive light. A compound action can make code actually more readable. I would like to clarify that I still feel that the lets-put-it-all-on-one-line attitude is very inefficient and under no circumstances would I endorse such a style.

M is unique in that it matures with you. Take commands, for example. You can progress from the noise version of the code to the shorter version (e.g., QUIT and Q).

Webbing Around

After achieving status netizen (citizen of the Internet) and having downloaded more software than I could ever use; clicking on every visible link, I started searching for information on M. It was modest compared to other programming languages. A novice can find information about M but it is not easy. To get the "good" stuff, one had to have a lot of luck, a similar amount of patience and the willpower to resist clicking on a link that would drive one off-topic into another sanctuary. This is definitely not the type of environment that would attract a person to discovering the M language. I started my own home page on M toward the end of April, with content appearing about a week later. The statement in my page states that I would like to establish a depot of information on the subject for both beginner and experienced M programmers with all others welcome to look around. I am confident that despite my limitations, I will get there.

My home page has opened new scenarios. I normally talk to at least two or three people at a time. Regardless of how communication originally commences, eventually communication does route to M. It is from such messages that I have learned that I am still a baby with respect to M. I can frankly say that I am not only learning from this experience, but that I am also enjoying myself in the process. I attempt to be very selective

when adding a site, so each site has to be vetted before it can end up in my links page. This usually means that I have to visit a site to check for suitability. I may also correspond with the entity in question. This vetting allows me to get to know what's available on M first hand.

Setting up and maintaining a site does require a quantifiable amount of effort. In my case, I perform the all-time-consuming job from 1:00AM on Saturday to 12:59AM on Sunday. During the rest of the week I handle email and jot down what has to be done the following weekend. Web publishing has given me enjoyment similar to what I had way back when I bought my first computer in kit form (to discount my overall costs) and wrote my first program.

The Internet is becoming the media through which many activities will take place. With so many sites attracting millions of WWW users, the M community should consider the benefits of attracting newcomers to the language through it. Combined M sites may provide enough attractiveness and content to pull people toward them. I envision the hooked-on and use-it effects.

The hooked-on syndrome applies to those that enjoy writing programs, be it as a hobby or livelihood. Many of those I have spoken to believe that once you discover M, you will like it. The use-it effect applies to those that have no interest in programming per se. They are in a position (current or future) to influence a decision. They need a program for the SOHO or are part of a group considering computerizing some aspect of their company. If people don't know that M exists, they can't recommend it or use it.

Who stands to benefit from an increase in the size of our community? Everyone. As with any free-market environment, an increase in demand for M products will result in an increased demand for programmers and will encourage new and existing companies to produce competing and innovative products that attract more users.

One of the ways we can achieve this is though web publications that have a portion of their pages aimed at newcomers. The ideal scenario consists of a number of publications that jointly produce what the combination of *Byte, PC Magazine*, and *Dr. Dobb's Journal* provide the computer industry in general.

There are many out there who love M. I am one of them. I have the impression that M has always taken second place to other languages that are less capable than itself. Unless our affection stops being self-centered, I believe that M will lose ground to existing and new competitors.

Conclusion

To an outsider, M resembles the prince in Beauty and the Beast with other languages showing signs of Gaston (a character in the same Disney classic). The Beast looked raw and ugly but was really kind and gentle. Gaston was handsome but eventually was shown to be the real beast. M has power and strength; it only needs a few of us to go out there and demonstrate these facts.

M

Chris Bonnici lives on the island of Malta and is the MIS manager for the Malta branch of Royal Insurance. He is involved with analysis and design and also plays an active role in programming. He has an advanced degree in computing and holds the post of Consultant for the Association of Insurance Companies. His email address is: Chribonn@keyworld.net