History and Continuing Evolution of FreeM: A Concept Whose Time Has Come (Again)

by Dick Walters and the members of Generic Universal M (GUM)

Abstract

Through its history, there have been several versions of M that were available without charge. Several members of the M community are nearly ready to release of a free version that will run under most versions of UNIX, initially Linux and True64 UNIX. This paper details the history of free M and the current effort, known as "FreeM."

Note:

This article is written on behalf of the people around the world who are currently participating in an exciting venture that may affect the future of the M community. Although the author is a member of that group and helped found GUM, he serves more as an administrative figure-head to the project. He has, however, been involved with M for quite a long time, and writes this article from that perspective.

Introduction

The history of M has taken some interesting twists over the years. In the late 1960s when it originated, M, then known as "MUMPS" was clearly a software generation ahead of its competition. It had persistent, shared data, multi-user architecture running on what seems today to be ridiculously small computers designed for laboratory systems. It used resident partitions of 1K size, shared buffers for global data and routines, and proved equal to the task of providing an effective vehicle for a budding hospital information system.

As a result, it grew and multiplied, but, as often happens, there arose many versions of M, with different names but retaining some of the key design concepts of its originators. By 1972, there were at least 8 dialects of M, none of them compatible with any other.

Through the leadership of Joseph T. (Ted) O'Neill, then with the National Bureau of Standards, the federal government sponsored an unusual experiment: they brought together the implementors, marketers and users of the M community, provided travel and hotel expenses, and instructed them to see if the dialects could converge into a single standard.

Working from late 1972 to September, 1975, this group, which evolved into the MUMPS Development Committee, did indeed adopt a new standard with many of the best features of existing dialects. All but one of the implementors agreed to adopt the standard. Through Ted O'Neill's insight, members of the group were commissioned to write reference manuals, the new standard itself, and other aids including a translator that would convert existing M code (including itself) into the new standard. Also through his insight, the language was submitted in September, 1975, to the American National Standards Institute for approval as a standard using the then little-known process of the "canvass method" whereby a standard could be approved even when major computer vendors did not have operational versions, provided that a sufficiently strong case could be made for its need and acceptance. With some difficulty, MUMPS became an official ANSI standard language: X11.1, in September, 1977. All but one of the participants in the standardization effort adopted the new standard; the Veterans Administration accepted it as the language in which their hospital information system would be written, and the usage of ANSI Standard M became the norm for many organizations.

Emergence of the First Free MUMPS Program

M began on minicomputers. It migrated slowly to some mainframes, but there were those who believed that it would never run on a microcomputer. Some people at the University of California, Davis, assisted by technical support from Goethe University, Germany, developed a single user version of MUMPS that ran on 8-bit CPU processors running under CP/M, (Control Program/Monitor) that dominated the early days of

microcomputers. This version became widely available and served to demonstrate that Standard MUMPS could indeed run on computers with minimal horsepower. By 1980, it was generally acknowledged that the 16-bit CPUs starting to appear would further open the door for MUMPS implementation on small computers.

Soon, a wide group of implementors undertook to implement MUMPS on 16-bit computers. Both Motorola and Intel compatibles were targeted, and these implementations proved that these new computers could compete effectively in a very large and expanding segment of the marketplace. These companies invested major dollars in development, resulting in the production of extremely powerful MUMPS systems. Other vendors joined the microcomputer world, and the place of microcomputers as M platforms was assured. Because M was an ANSI standard, these vendors were able to compete head-on with vendors of minicomputer-based systems and earn a solid place in various application domains.

By the same token, the need for a public domain version of M diminished. Funding for continued development of the earlier version (now running successfully under MS-DOS) dried up, and by 1986, development ceased.

New Needs for Non-Commercial M Systems Spawned a New Effort

The MUMPS Development Committee, which started late in 1972, continued to meet regularly and to add new features to the language. M was officially revised in 1984, again in 1990, and most recently in 1995. The group had always had important membership from other countries besides the United States, and with their help, M was accepted as an ISO Standard after the 1990 revision, and the 1995 ANSI Standard is currently undergoing final preparation for publication by ISO as an ISO/IEC Standard.

As the language evolved, new concepts were proposed. At first, most of the proposed revisions were new features independently developed by the various M implementors. Later, as development became more costly, implementors became reluctant to commit to new functionality until it had been adopted by the MDC. Eventually, each time new features were proposed, the argument was raised: "but no one has implemented this feature. How do you know it will prove effective?"

It was clear that a new justification for development of Open Source versions of M had emerged: a test-bed where new concepts could be tried, evaluated, and reported back to the MUMPS Development Committee which could then make informed judgments as to the merits and disadvantages of new proposals.

Although the need existed, funding did not follow to support that need. But some people don't give up easily, and beginning in about 1992, some of the developers of the public domain microcomputer project started to explore interest in a broad-based effort to develop a new version that could serve the MDC as a test bed for proposed new language features. The project appealed to some people in Minnesota, Boston, San Francisco, Davis, and Texas, and it bubbled along slowly for several years under the name Generic Universal M, with L.D. Landis (HealthPartners, Minnesota) as the technical director, and the author as the figurehead. The name of the University of California was brought in to give the project credibility, and there were signs that it might survive. They even got so far as to develop a global handler, which has been available for several years. But, without funding and all work being done on a volunteer basis, the compiler simply did not happen.

1998: A Turning Point in the Evolution of M

Meantime, ominous signs started to appear, affecting the concept of a standard supported by multiple vendors. One of the main M implementors acquired DataTree, vendor of one of the pioneer microcomputer implementations of M, in early 1993. The largest hardware manufacturer still supporting an M product (Digital Equipment Corporation) sold to a competitor in 1994. Although these systems continue to be supported by the software company that purchased them, strong pressures are being brought to bear on users to make a switch to a new, innovative M-derived product which no longer makes any attempt to adhere to or participate in the standardization process. It is their view that standardization takes too long, and they believe they can compete more effectively with large relational database vendors by developing products that absorb relational concepts offering new "post-relational" features (many of them based on M concepts).

One other highly competitive vendor remained, however, until June 19, 1998, when the final remaining M vendor, with a substantial market share, sold to the same company that had bought the other two leading implementations.

Since then, interest has grown considerably in the emer-

gence either of one or more new commercial vendor(s) who might attract an appreciable share of the market; or completion of one or more Open Source versions of M; or both. A group has been actively pursuing levels of expertise and levels of support that might bring these concepts to fruition. The GUM participants have been active, joined by interested parties in the U.S., U.K., Europe, and Japan. A number of promising leads were developed and are still being explored. However, an even more important event took place early in 1999 that gave new impetus to this effort.

FreeM: A Jump Start to an Open Source Version of M

Early this year, this author and others in the GUM project were informed that an anonymous donor was willing to make available a version of M that, while it had some shortcomings, would still be suitable for potential distribution as an Open Source, free version, provided that the major shortcomings were addressed. After a flurry of activity, L.D. Landis accepted the lead in coordinating a frenzied effort to address those shortcomings. He was joined by what has by now become a completely international group of M developer experts, with workers in England, Europe, and the U.S. contributing to the effort. The results to date (remember, we're talking weeks, not years) are impressive. Many, but not all, of the missing elements have been addressed and fixed, and others are on a prioritized list and being tackled by people working all over the world. While this effort has not yet met the conditions by which FreeM could be released as a free, Open Source product, it is getting close.

This is one of the most exciting processes to hit the M community. Imagine a team of experts, some of them displaced by corporate changes in their previous M positions, joining hands across the oceans to work harmoniously and collectively to solve implementation problems that, in all likelihood, no single individual could have addressed alone without many months or years devoted to the effort.

It is a reflection on the time being selflessly donated by this group that members of the FreeM list-serve have exchanged hundreds of messages in the past few weeks, sharing code, suggesting priorities, accepting consensus decisions and producing more code.

What is this FreeM? Although the program was contributed as an Open Source product, it is clear that the version as released to GUM has already been through a long period of evolution and testing, and that it has benefited from major expertise in M, compiler development, and performance issues relating to M systems. In today's market, a product of this maturity would fetch a price in the hundreds of thousands of dollars. It is not a fly-by-night version, nor full of holes in the standard nor lacking in sophistication. To cite one major benchmark, one contributor wrote: "I have loaded VA FileMan 21 and have had only a few problems. Most of them have to do with . . . the system setups." Anyone familiar with this package would realize at once that this version of M is already well-tested in many important respects.

Where to From Here

It is certainly impressive that an Open Source version of M can be counted on to be reliable, well tested by M experts, and, when released to the public, formally approved by an international group of M users whose authority and reputations are quite strong. This is what will happen to FreeM. But there are other points that further strengthen the credibility of the process by which this version is coming to the public. Some of these additional factors include:

- The GUM community plans to obtain the Validation Suite for the 1990 Standard and to use that suite plus enhancements to test FreeM.

- Contributors are developing utilities that will give the version an operating robustness not found in typical free offerings.

- MDC partners in the process are acting as liaisons to make certain that FreeM is recognized by the MDC as a useful test bed for trying out new language elements.

- Liaison between the original anonymous donor of FreeM and GUM has been established, so that the product, when released, will have the full approval of its donor.

(Reflect for a moment on the remarkable success of Perl, a language that almost came close to M in some areas and falls woefully behind in others. Can't we find a similar niche for M in that same community?)

Some Commonly Asked Questions About FreeM

FreeM works. It has some known bugs and some known

holes that need to be filled, but it is a solid, almost complete, version of the 1995 ANSI Standard.

FreeM currently runs under Linux and several other flavors of UNIX. Efforts are under way to make it run under Windows and Windows NT. The basic intent of the FreeM project is to leave the door open for as many platforms as possible by avoiding code that might restrict its operation on other systems.

The code for FreeM was donated by someone who wishes to remain anonymous. We respect that offer, and to further preserve that anonymity, we are not making public the name of the individual who is acting as liaison with that donor, whose name is unknown to the other members of the GUM team. Speculation would be idle, but suffice it to say that the donor knows M thoroughly!

Workers on the GUM project include many MDC members, some individuals released from former M product companies who are not constrained by legal boundaries that would prevent their contribution, and some others who have had a long-standing interest in promoting the concepts that are embodied in M.

At this point, the GUM project can use additional people with expertise in the following areas:

- C programmers interested in M
- QA/Conformance test case creators
- Documentors
- Platform support
  (e.g., for focus on HW/OS combination)
- M developers for utilities and tools

We are not yet ready for people who want to develop and deploy production applications using FreeM. On the other hand, people who are willing to test some known applications, as a means of further testing FreeM, are certainly welcome to join the developers working on the basic code.

People who want to get involved should join the GUM list-serve and prepare to receive a good many email messages every day on subjects ranging from small bug fixes to global review of the final approval process. To subscribe to the GUM Project list (which is a majordomo list), send an email to:

majordomo@ldl.HealthPartners.com
where the subject is ignored, but the message body contains (only): subscribe gump
You are then sent the greetings and purpose of the list, etc.

When FreeM has received approval by an international board and from the anonymous donor, word will go out via as many channels as it takes to publicize this landmark event. We cannot predict precisely when that will be, but we hope to have a distributable version available by summer of 1999, and we are eager to demonstrate FreeM at the MTA meeting in September.

What Comes Next?

FreeM is not the final solution to the survival and continued growth of M. Other Open Source versions should be made available (we intend, for example, to link the GUM global handler with FreeM as one option that people might want to use for experimental purposes). Commercial versions are needed. But FreeM is a great start, a great boost to folks who were losing heart about the future of M, and a tremendous opportunity to stand up and be counted.

What we can say is that the events described above have provided new impetus to those around the world who love M and want to see it survive as a standard language.

M

Richard F. Walters, Ph.D. is a professor at the University of California, Davis.
