

## THE "PROJECT WHICH FAILED" WHICH SUCCEEDED

Robert L. Glass \*

It seems especially appropriate, shortly after the death of computing giant Maury Halstead, to share this heretofore untold story of a spinoff from some of Halstead's early work.

This is a "project which failed" story, unfortunately; and yet in human terms it is also a "project which succeeded" story. I think Halstead would be proud of what happened to this piece of his technology.

I imagine there are lots of stories like this one across the computing landscape -- a group of unknowns, piggybacking on the work of a giant, carrying it into significant new technology areas, but never publicizing their work and thus remaining anonymous. It is to these unknowns, as well as to Maury Halstead, that this story is dedicated.

*Machine-Independent Computer Programming*, by Maurice Halstead, was a pioneering book of the early 1960s. It told the story of a new programming language -- Neliac, an Algol derivative language defined for Naval Electronics Laboratory (NEL) use -- which had certain almost magical properties. One such property was that it was a language defined so as to make it easy to build a compiler for it. In that, it succeeded admirably. The listing for the full compiler, a marvel of simplicity, could be held easily on one fully extended hand, so small and light was it. Another magical property was that the compiler was coded in its own language, an idea so daring for its time that the typical software mind boggled that such a thing could be done. ("But how do you start one of those?" was the most frequently asked of many incredulous questions).

Neliac, as Halstead described it in his book, was a language for systems and scientific programming. The thrust of this story is the attempt by an earnest group of programmers at the Aerojet-General Corp. in Sacramento to make the language viable for commercial programming as well.

It must be remembered that in the early 1960s this, too, was a daring concept. Traditionally there had been a dichotomy between scientific folk, emerging from the world of mechanical and electrical desk calculators (the forerunners of today's hand-held electronic marvels), and commercial folk, emerging from the world of punched-card manipulating equipment. The scientific computer of the era was a seemingly-blazing-fast binary wonder, which spoke in octal and did arithmetic faster than anyone had every dreamed possible. The commercial data processing machine of the era was a slower, decimal-operating computer, which operated on strings of alphabetic and decimal numeric character strings (called "alphameric") of varying length, where the data processor had no words as such but used software-controllable flags or "word marks" to define the bounds of a computer-manipulatable string. Obviously, the scientific/commercial twain rarely met. Their worlds were worlds apart.

Of course the revolutions of the mid-60s -- the IBM 360 computer, which manipulated both words and strings, and the language PL/I, which provided for the needs of both kinds of users -- were to change all of that. But in the early 1960s, the team at Aerojet could not have anticipated the magnitude of the solution that was coming. They only knew -- and they were ahead of their time in knowing -- that a scientific/commercial solution was needed. And they set about to provide one.

---

\*Author's address: 6925 56th Ave. S., Seattle, Washington 98118

There was Donald W. Starr, as much the complete programmer as anyone I have ever known. Don could conceive a solution, sell the need for the solution to management, aggregate and lead a team in performing the solution, play a major role in the solution himself, document the result with clarity and humor, and teach classes in the product's use. In the world of unsung heroes, Don looms large in my memories.

There was Pat Crisman, a female version of Don Starr. No, Pat wouldn't like that. Pat was her own person, her own programmer, a liberated woman in an era before women en masse strove for liberation. Tell Pat to go innovate a new or better solution to an old problem, and she would do it.

There was Hugh Barlow. Good old Hugo, the "tough but oh so gentle" coder extraordinaire, whose programs sparkled with carefully-conceived brilliance. Hugo may not've had a college degree, but his mind functioned at a coding level above the best of us.

And there were us also-rans. Two or three others of us who, like all programmers of that era, knew we were the best programmers around. If only we didn't have the Starrs and Crismans and Barlows who were so clearly better than we were!

We were a chief programmer team, I now realize, in an era 10-15 years before the notion hit the literature. Don was our shining starr (a pun which inevitably we worked to death), and we were his satellites, happy to hang onto his celestial coattails and be a part of whatever process he was pursuing. It couldn't have been easy for Don to lead a team of us prima donnas in an orchestrated direction, but he pulled it off. I believe in chief programmer teams -- if you have the right chief.

To set the stage for the technology of this story, it is necessary to describe some political preludes. For one thing, an Aerojet management decision several years ahead of its time had edicted that the scientific and commercial programming organizations in the company be merged. Physically co-located. Technologically served by the same computer. Managerially amalgamated. The dichotomy between the two was to be erased.

Additionally, some decisions had been made about that common computer. It was to be a binary, word-oriented IBM 7040, not as fast as the IBM 7090 which was the drag racing champion of its time, but a better compromise for an often I/O bound data processing workload. Studies had shown that the 7040 was a more cost-effective solution than the decimal, string-oriented machines which had historically served the data processors.

And finally, that 7040 was to be driven by a home-grown software system. An operating system called Nimble was to be designed and implemented in-house, some needed utilities like math libraries and sort packages and report generators were to be added, and -- to bring the story back to its focus -- a Neliac compiler had been obtained from the government, put into immediate use for scientific applications, and work was begun on the necessary commercial language supplements. (And all of that software effort, it is worth repeating, had evolved under the alert eye and specific planning of Don Starr. It was, for that time and for all time, a truly impressive undertaking for an industrial production-oriented computing shop. Essentially we took a bare-bones 7040 and built the whole software system for it).

Neliac was critical to the whole scheme. Nimble, the operating system, was coded in it. Entirely. (Some parts were later recoded into assembler due to efficiency problems). And in fact, all the other system components were coded in it, too. For us, it had already become both a scientific and a systems programming language when we set out to make the commercial enhancements. Again, that may not seem exciting now. But for its time, this whole project -- and especially the reliance on a high-order language -- was daring.

The commercial enhancements were easy to define. We needed a good, capable sequential I/O package, buffered and blocked, and callable from the language transparent to the user. We needed an indexed-sequential package to support pseudo-random, real-time accessible files (more innovation!). We needed data structures, including the capability to define and manipulate heterogeneous data aggregates which often contained non-word-oriented bit and character strings.

And we did it! We added what we needed to that elegant language of Halstead's, retaining its fast-compilation origins but enhancing it for a whole new domain. We implemented what we added, sometimes consistent with the compiler's simplicity, but sometimes not. And we put it all, the language and the operating system and all its trappings, proudly on the air. Our commercial colleagues were now technologically as well as managerially integrated with us.

Why, then, is this a computing project which failed? This part of the story is hard to tell. Politics reared its ugly head again.

Remember the Biblical story of the puny David and the giant Goliath? Well, for reasons that I'll never fully understand, an adversary relationship suddenly exploded between our little David-like chief programmer team, and the mighty Goliath IBM. Since I am fairly sure we did not start it, I have to believe that it emerged from an IBM marketing strategy. But wherever it came from, it came as an enormous shock.

Up until that time, IBM had been delightfully cooperative. In one of the finest gestures I can imagine a hardware vendor providing, they had put a pre-delivery 7040 in the basement of their own posh downtown Sacramento offices, and had given us Aerojet software developers office space and free rein in its use. By the time the 7040 was delivered to Aerojet, and thanks to IBM's help, Nimble and Neliac were well wrung out. The transition to production use went relatively smoothly.

And then it all changed. IBM marketing people began frequenting the offices of Aerojet computing management. The message gradually trickled down to us developers. IBM wanted Aerojet to scrap Nimble and Neliac, and switch over to the vendor-issued OS and Fortran/COBOL. We would be more like everybody else, they said. Job and program swapping would be possible with other IBM-vanilla companies, they said. And, of course, the mighty weight of IBM's own support folk would be available, they said. Why mess with a small team of nobodys, when the best was available?

I am bitter, and it shows. Gradually, as the IBM pressure on management continued, Don Starr's team took on a beleaguered look. Defending ourselves was taking a disproportionate share of our time. We had better, more innovative things to do than fight political battles.

In the midst of all of that, one of the key managers who had sustained and supported our efforts from the very beginning left the company. His replacement, a conservative man whose technical knowledge was a cut below, quickly capitulated to IBM. Nimble and Neliac were unceremoniously unplugged, and trashed, and the wave of the future -- OS, Fortran, and COBOL -- washed over us.

Washed away, too, were Don Starr, and Pat Crisman. With their dreams dashed, there was no reason to stay. What six months before had been a powerfully motivated, innovative group was now in shambles, groping for a new role in an IBM-occupied country.

With the perspective of time, I think now I begin to understand why IBM changed. Remember that IBM's own solution to the problem we were solving -- the 360 and PL/1 -- were right over the horizon. I think, for one thing, that they were simply bulldozing a clearing for that future installation. I think also that they were threatened by the concept that a local software team could control

a corporation's destiny. After all, if there is significant in-house talent, there is nothing to prevent an installation from picking another, probably cheaper, computer vendor, and not relying on IBM's own support. It was, I now believe, a matter of bondage. IBM wanted to insure that Aerojet remained a captive customer. And Don Starr's chief programmer team was standing in the way of that bulldozer.

But that is all a political digression from the technical story I really wanted to tell. The technology of the story, I believe, was a success story -- a project well ahead of its time, in so many ways, that I still get a surge of pride and adrenalin when I think of it. And the humanity of the story, I also believe, was a success story -- a team of bright people able to merge their individualities into a functioning, compatible team. It is the success element of this project which failed that I would like to focus on. And it is that success that has caused me to dust off this nearly 20-year old story, raise it from anonymity, and share it with you readers.

There are some dangling ends to the story. Some of them, I can tie up. Others I can't.

Maury Halstead, of course, died in early 1979. He will be sorely missed.

Don Starr moved on to Computer Sciences Corp., and over the years I have lost track of him.

Pat Crisman went to Project Mac at MIT, the pioneering timesharing effort, and most recently is working for Honeywell.

I remained in the Aerospace Industry, and at the end of my 25th year in aerospace and computing I find myself a ten-year-plus person at The Boeing Company.

And what of Aerojet itself? Gradually, over the years, that once-brilliant installation atrophied with the fortunes of the company, and eventually the computing facility in Sacramento disappeared entirely.

Which, I think, is the final irony to the political side of the story. The Neliac team may have lost the political battle. But in the end, nobody won.

(This article is an excerpt from the book The Second Coming: More Computing Projects Which Failed, published by Computing Trends, 6925 56th Ave. S., Seattle, WA 98118 written by Robert L. Glass and Sue DeNim).