the Availability Digest

www.availabilitydigest.com
@availabilitydig

The End of Custom Software

November 2017

Back in the 1970s and 1980s, my company, The Sombers Group, specialized in writing custom software packages for our customers. Today, software-implemented packages are available to meet almost any computing need. But back then, there were a paucity of such products. The need for software solutions to problems kept about a dozen of our people busy serving a half-dozen customers at any one time.

We programmed primarily using the basic assembly languages of the machines on which we worked. The only alternatives at that time were COBOL and FORTRAN – Java had not yet reared its wonderful head. Most of our work was done on Digital PDP-8s and PDP-11s and on Tandem systems (now HPE NonStop systems).

To meet the needs of one of our customers, we implemented a payroll package running on a PDP-8. The package was so successful that we decided to build a company around it to provide payroll services to small companies. Thus, MiniData Services was born. We put up billboards advertising "You Pay Us 9, We'll Pay 15." This was an indication that we would do a fifteen-person payroll for just $9.

We purchased two PDP-8 computers to run our payrolls. We only needed one from a capacity viewpoint, but realized we needed two in case a computer failed. However, in the years that we ran the dual PDP-8s, we never had a system failure. We later upgraded to PDP-11s to gain additional processing capacity.

Our first customer signed up with us within one week - a small local bank. We proudly delivered our first weekly payroll to the bank, only to be greeted with a telephone call filled with expletives. It seemed like we were paying their employees several thousand dollars an hour! It turned out that our payroll package was set up to accept time-worked in weeks, whereas the bank had given us time-worked in hours. So we were paying each employee a week's salary for each hour worked. We learned our first lesson – check the payroll sheets before we deliver them.

We printed payroll summary sheets and paychecks on a high-speed printer. One problem we faced was that it took several minutes to tear down the printer and change forms. We were doing hundreds of payrolls for companies with employees numbering in the five to twenty range. If we had to change the printer for each summary sheet and for the specific checks for each company, we would never be able to process enough payrolls to build a business. We solved the problem by using a standard check for all customers.

We would print the payroll summary sheets for about a hundred customers and then put the standard blank checks in the printer. We printed each check with the company name, the employee name, and the amount. We burst the checks and put the checks for each company with its payroll summary sheet. We then passed the checks through a magnetic ink encoder to print the companies' bank account number, the check number, and the amount in magnetic ink at the bottom of the check.

We were told that this was illegal, but we never had a problem with it. The technique let us print the payrolls of a hundred or so companies in a single pass, thus letting us process the payrolls for hundreds – and eventually thousands – of companies each week.

MiniData's servicing of small payrolls became so successful that, after a few years, it was copied by payroll giant ADP. MiniData Services grew to the point that it was ultimately acquired by the payroll division of Control Data Corporation.

Another customer of ours needed a software package that would allow it to communicate with remote computers. Initially, the need was to send messages between Digital PDP-11s. To accommodate this need, we developed our product NetWeave. NetWeave competed with IBM's MQ series. However, whereas MQ Series put messages to be exchanged in a queue, NetWeave immediately transferred the message to the recipient machine with no queuing, thus earning its nickname "the lean, mean messaging machine." NetWeave was expanded to support several other processors such as PDP-8s and Tandem systems.

NetWeave became quite popular and was licensed by many companies for interprocessor communication. It is still in service today, some forty years later, based on the payment of license fees. However, since NetWeave was a middleware product and operated just above the operating system, most companies that are still using it are not at all aware of where it is being used.

Perhaps the largest special software system we built was for the New York Racing Association (NYRA). NYRA operated three race tracks in New York state – Aqueduct, Belmont, and Saratoga. Each race track depended upon a totalizator system, known as a 'tote' system. The tote system accepted wagers from ticket issuing machines (TIMs) around the track, calculated the potential payoffs for each horse if it should win, place, or show, and posted these payoffs on a large infield display board and display terminals situated around the stands. It also calculated the payoffs for more esoteric wagers such as the daily double (picking the winning horse in two different races) and the trifecta (similar, but for three races).

When we were called in, the tote systems for the NYRA race tracks were being provided by Automatic Totalizators, Inc. (Autotote) located in Australia. The systems were implemented via relay logic. The mechanical relays would cause problems, and Autotote wanted to consider replacing them with computers.

We implemented a tote system for Autotote using a PDP-8. This included building a TIM-scanner complex that would scan the TIMs in the race track to pick up wagers. The wagers were logged in the computer tote system and signals were returned to the requesting TIMS to issue the tickets. The first PDP-8 system was put into service at the Saratoga race track which operated for just a month in the summer in Saratoga, New York. It performed superbly, and Autotote decided to use this technology at its larger racetracks at Aqueduct and Belmont.

To provide the capacity for these larger race tracks, we ported the code to PDP-11s. Three PDP-11s were provided. One was a backup system. The other two systems actively processed TIM requests and compared results. If both computers agreed to sell a ticket, a sell signal was sent to the TIM. If both computers rejected the sale, the TIM was told to cancel the ticket.

If one computer approved the sale but the other rejected the sale, the TIM was told to cancel the sale. If there were a series of cancellations by one computer but not the other computer, the computer that was doing the canceling was taken out of service and the third backup system replaced it.

We later reprogrammed the tote system to run on Tandem systems. This was in the early days of Tandem, and we were one of the first users of these highly reliable systems. Reliability was of paramount importance. Imagine a customer who wanted to place a $50 wager on a long shot but couldn't because the tote system had just failed. Then his horse wins and pays 20:1. He would have won $1,000. Incidents such as this could cause riots at the race track.

Sadly, the need for custom programming has passed into history. Existing software packages exist for just about every need. True, thousands of programmers are still kept busy. However, they are working on the software packages upon which everyone now depends rather than custom, one-off software projects.