Musicians as Programmers

By Michael Drapkin

Musicians are recognized as a special talent pool in the computer industry.

There's a story floating around the computer industry that during the early days of computers when programmers were scarce, IBM preferred to recruit and train programmers from two different groups: musicians and accountants. This preference made sense at the time; people from both groups were accustomed to performing tasks requiring sequential processing. Musicians played a piece from beginning to end, and accountants added up long columns of numbers from top to bottom (spreadsheet programs hadn't been invented yet).

When I started programming commercially, I was concerned that managers in the business world would look unfavorably upon my music background. I put the University of Rochester on my résumé instead of the Eastman School of Music. Much to my surprise, I learned that the computer industry in general regards a musical background as an asset. Today, managers frequently exclaim, "I see you got a degree in music." My standard reply is, "Yes, I'm a renegade musician." They usually nod in approval and acceptance.

NATURAL-BORN PROGRAMMERS

Musicians seem to have a natural talent for programming computers. Both skills deal with abstract concepts. In addition, musicians and programmers conceptualize their media in a hierarchical organization. When musicians think about Beethoven's Fifth Symphony, they see it as a singular work that is divided into different movements, each of which is further divided into sections, phrases, bars, and single notes.

Programmers think about their programs in a similar manner. Software certainly is hierarchical, starting at the top with a system such as "Accounting." This system is divided by function, such as payables, receivables, G/L, etc. Each function may be divided into subprograms such as data acquisition, which are further divided into loops, blocks, and lines of program code. The words in an individual line of program code correspond to the notes on a page of music. Programs even have repeats and first and second endings in the form of loops and cases.

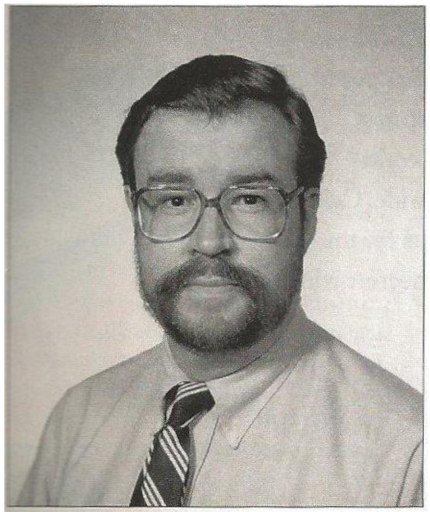
Many people relate music to right-brain activity and computers to left brain activity. According to John Podzius, Ph.D., associate psychologist at the Harlem Valley Psychiatric Center in New York, "Reality is more complex. Musicians and programmers use both left and right cerebral hemispheres when doing their respective activities. Left-brain activities include language and ascribing meaning to symbols. The right brain involves itself in geometric spatial relationships and perception of rhythm. To say that one group or the other only uses one hemisphere is overly simplistic. In reality, bilateral brain activity is involved in the daily activities of both musicians and programmers. As a group, they probably use more of their brain power than people in other fields."

SOLITARY LIVES

Musicians and programmers also spend a lot of time working alone. Musicians spend a large portion of their time practicing or "woodshedding," while programmers work largely by themselves at their terminals writing programs. Even as part of a team, they still perform a solitary role. The musician plays an individual part within an ensemble, and the programmer develops individual parts of a larger project.

Musicians and programmers have a field-independent cognitive style," says Podzius. "Cognitive styles are individual differences among people in their thinking, problem solving, and approaches to social situations. A field independent style is characterized by analytic and accurate thinking and by a preference for solitary, non-social activities in career pathways and interpersonal relations. The requirements of music and programming are quite consistent with the cognitive skills and intrinsic preferences of field-independent individuals. Analytical thinking, the capacity to perceive part/whole relationships, and comfort with solitary, self-reliant work are requirements for accomplishment in both fields."

Both musicians and programmers require a tremendous amount of self-discipline. Musicians who become programmers are frequently lauded for their abilities as self-starters who work independently and take initiative to get their jobs done.


John Podzius, Ph.D.

THE ULTIMATE PERFORMER

The act of creating computer programs holds a particular fascination for many musicians. No matter how much a musician practices a passage, and no matter how many times he or she gets it right, there is no guarantee that it will go the way it should in the heat of a live performance. Although it's a liability, the element of risk is also a source of great excitement for the performer.

In programming, the computer is the ultimate obedient performance machine. For a musician, the ability to write a program and then watch the computer perform it unerringly is a strong source of satisfaction. As musician and former programmer Rick Kunis recalls, "With programming, I experienced feelings ranging from severe frustration all the way to the sublime feeling of seeing a program work." Once a program is built, the performance is repeated over and over again as it is fine-tuned until it's absolutely perfect. Programmers call this process debugging.

MUSICIAN TO PROGRAMMER

Although musicians have always constituted a segment of the computer programming community, the flight from performer to programmer reached its zenith in the early 1980s. While some musicians discovered that programming was better than waiting tables, there wasn't a wholesale migration until the Reagan administration began to cut support for the arts in America. This caused the lives of many struggling artists to go from marginal to untenable when funding for performances began to dry up.

"I made up my mind that freelance music was too uncertain. I wanted more choices about my lifestyle, and I wanted to have a certain income," says computer analyst and former musician Katherine Askew. Computer programming was a natural solution because, like the music business, the computer industry only cared about whether you could produce and not whether you had a college degree.

Musicians generally acquire programming skills in one of two ways. Many enroll in a program of coursework offered by a college, university, or trade school. The entire course of study might span a couple of semesters to a couple of years. At the conclusion of their formal education, they begin to search for a full-time job.

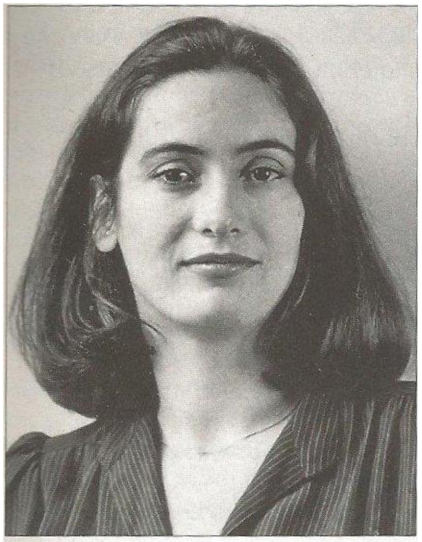
Other musicians-turned-programmers are self-taught. They become fascinated by computers and end up buying some books and software and teaching themselves to program purely for the enjoyment of it. At some point they may or may not realize that they've stumbled upon a marketable, income-earning skill.

If you take the second route, I suggest learning Microsoft *Quick C Compiler* if you're a PC user, or Symantec *Think C* you own a Mac. (A compiler is a piece of software that turns lines of programming code into computer programs.) C is the premier programming language used today, so pick up a copy of *The C Programming Language* by Kernighan and Ritchie, published by Prentice-Hall. This book is the C programmer's bible and a terrific reference. In addition, many tutorials are available on C programming. Finally, try to find a friend who knows C programming for those times when you get stuck.

GOING COLD TURKEY

Katherine Askew was a Juilliard graduate in viola when she decided to switch to computer programming. "I ran into a violinist going to the NYU computing program, and I heard that another guy already got a job, as well as a bass player from an orchestra I was in. I saw an opportunity to make something happen."

In 1983, Askew enrolled in the NYU Diploma Program in Programming and Technology for three semesters of night classes. She spent a lot of time in the computer lab "practicing." "I approached it like a musician, doing the equivalent of scales and etudes on the computer."


Katherine Askew

She also found that being a musician was helpful. "When I was nervous, I didn't show it, especially when I didn't know the piece very well. Performing experience really helped me tremendously. I know how to communicate and present to an audience." She is currently working full time for a firm that develops computer imaging systems in Manhattan.

SELF-TAUGHT SKILLS

Rick Kunis did some arrangements on a few Beatles albums in the 1960s. He was producing and recording for CBS in the late 1970s when he got interested in word processing. "I took a five-week course in word processing and finished the curriculum in two weeks." He accomplished this by going to class early to practice, and he also started learning programming from the textbook on his own.

While supervising a word-processing group at E.F. Hutton, he got involved with a group that was designing forms and learned the basics of programming on the job. Kunis eventually moved to a consulting job at Merrill-Lynch. In computer-industry parlance, consulting is the equivalent of freelance playing in music. However, the projects are measured in months instead of days, and the pay is significantly higher to woo programmers to temporary projects.

Kunis' skills as a pianist manifested themselves at school in unexpected ways. "They had a career day for the industry to meet with students, during which a jury clocked my typing at an average of 130 words per minute." Obviously, the years of scales and exercises paid off in a big way!

PROGRAMMING TO MUSIC

Ironically, computers paved the way for Kunis to return to music. "A friend of mine showed me MIDI, and I was floored. I started getting involved with music again through the electronic end of things," he explains. He got heavily involved with synths and MIDI, which ultimately led back into music on a full-time basis. He is currently working with composer Alan Menkin.

It is clear that programming and music share many common elements, including discipline as well as conceptual and cognitive skills. While it is safe to say that musicians often make great programmers, programmers don't necessarily make good musicians because music requires the innate talent of artistic and emotional expression, nevertheless, the link between these two fields is a fascinating one that many electronic musicians may wish to explore further.

Michael Drapkin is a clarinetist and a computer programmer. He has performed with major symphonies and pays his mortgage by consulting and programming for major corporations in the New York City area.