BLOG @CACM

The *Communications* Web site, <http://cacm.acm.org>, features more than a dozen bloggers in the BLOG@CACM community. In each issue of *Communications*, we'll publish selected posts or excerpts.

Follow us on Twitter at <http://twitter.com/blogCACM>

DOI:10.1145/2967972

<http://cacm.acm.org/blogs/blog-cacm>

Adding Art to STEM

Perry R. Cook considers the career path that led him to STEAM.

Perry R. Cook
Combining Arts and Technology, Creating STEAM Curricula
<http://bit.ly/1rC47EO>
April 28, 2016

I grew up an artsy+nerdy kid, singing in choir, playing in band, as comfortable with a soldering iron, with fixing or hacking an old radio or electronic organ, as with chord progressions or improvising harmonies on the fly. In high school, I sang in every choir, played in every band, and did theater and speech. I also kept a keen and interested eye toward technology, especially music technology.

My original goal in going to conservatory in 1973 was to become a band/choir teacher, double-majoring in trombone and voice, with education and techniques courses for choir and band certification. But something fateful happened; I discovered my music school had an electronic music and recording studio. Also around that time, and at the urging of my trombone teacher, I became a voice major. What really happened is I became a de facto major in a recording and electronic music program that my music school did not have (yet). I spent every available minute in those studios, also doing location recordings, editing tapes, soldering patch cords, and reading

every book and journal I could find on audio, acoustics, recording, and electronic music.

I loved the studio work so much that in 1976, I ended up dropping out to become a sound engineer for about five years. I did lots of outdoor and indoor sound reinforcement gigs, some system designs, lots of building, installing, repair, and some studio work as well, both as an engineer and singer. All the while I was working feverishly in my own home studio, collecting (or building) a good variety of synthesizers, recording gear, and effects devices. I made lots of music, but the more I worked as a sound engineer, the more I realized there was math and science I needed to know to be as creative, and valuable, as possible.

So I went back to school in 1981, this time in electrical engineering (EE), but also finished my music degree in the process. Pretty much every course I took in my EE program, I was asking myself how it applied to sound, acoustics, and music. I finished with honors, and even though I was now dual-degreed, I knew there was much more that I did not know. I applied to graduate schools, and got into Stanford University, and found myself at the holy city (for nerds like me): the Center for Computer Research in Music and Acoustics, also called CCRMA.

There, I got to work with brilliant people like John Chowning (the inventor of FM sound synthesis, and pioneer of spatial sound and compositional uses of sound synthesis), DSP guru Julius O. Smith, Chris Chafe, Dexter Morrill, Max Mathews (the father of computer music), John Pierce (former Bell Labs Executive Director of Communications Sciences), and many others. I worked on physical modeling, new performance interfaces, created countless new software programs for all sorts of things, and researched and developed physics-based voice models for singing synthesis, which was the topic of my Ph.D. thesis.

CCRMA taught me so much about so many topics, but possibly the most important thing was that art, science, math, and engineering can (and should) be linked together. I observed students that study in this way learn differently, better, and create amazing and novel things just as part of their coursework. Pretty much all of the curricular elements of CCRMA are STEAM (science, technology, engineering, arts, math) in nature; math, music, physics, psychoacoustics, engineering(s), and other technical/design/art areas are woven together tightly and constantly.

When I moved to Princeton University in 1996, I got to take over a course Ken Steiglitz (EE/CS) and Paul Lansky (Music) had created, called "Transforming Reality Through Computer." It was really an applied DSP course, but with musical examples and projects. For quite a while I had been

teaching a CCRMA short course every summer with Xavier Serra called “Introduction to Spectral (Xavier) and Physical (Perry) Modeling.” My 10 lectures had turned into a fairly formal introduction, a set of notes, and eventually book chapters, to which I added a couple of spectrum analysis chapters, and a couple more on applications, and it became the book *Real Sound Synthesis for Interactive Applications*. That book and course was my first “scratch-built” STEAM curriculum, cross-listed in CS, EE, and Music at Princeton. The focal topic of the book is sound effects synthesis for games, VR, movies, etc. That topic also earned me a National Science Foundation (NSF) CAREER grant.

At Princeton, I also introduced a course called “Human Computer Interface Technology,” developed jointly with Ben Knapp and Dick Duda at San Jose State University (they got an NSF grant for this), Chris Chafe and Bill Verplank at CCRMA, and other faculty at the University of California, Davis, and the Naval Postgraduate School in Monterey. The emphasis at Stanford and Princeton was on creating NIMes (New Interfaces for Musical Expression), putting sensors on anything and everything to make new expressive sound and music controllers. Another STEAM course was born.

I continued to weave musical and artistic examples into all of my teaching and student advising. The next major new STEAM curriculum creation was the Princeton Laptop Orchestra (PLOrk), founded in 2005 by Dan Trueman (a former grad student who then joined the music faculty at Princeton) and myself. This course combined art, programming, live performance (some of it live coding in front of an audience!), engineering, listening, recording and studio techniques, and much more. Dan and I begged and cajoled around the Princeton campus to get money to get it off the ground, getting funds from Music, CS, the Dean of Engineering, the Freshman Seminar Fund, the Sophomore Experience Fund, and other sources to put together an ensemble of 15 “instruments” consisting of a laptop, a six-channel hemispherical speaker, amps, and controllers. Result? BIG success. As just one exam-

ple of hundreds, here is a quote from a PLOrk member, a female undergraduate music major, a cellist who had never programmed before:

“However, when everything worked the way it was supposed to, when my spontaneous arrangement of computer lingo transformed into a musical composition, it was a truly amazing experience. The ability to control duration and pitch with loops, integers, and frequency notation sent me on a serious power trip ... This is so much better than memorizing French verbs.”

Within a year or so, we had applied for and won a \$250,000 MacArthur Digital Learning Initiative grant, allowing PLOrk to build custom six-channel speakers with integrated amps; buy more laptops, controllers, and hardware, and grow to 45 total seats in the orchestra. We also toured, played Carnegie Hall, hosted and worked with world-famous guest artists, and inspired a horde of new laptop orchestras (LORks) around the world. Dan also worked on modifying the Princeton undergrad music curriculum to incorporate PLOrk courses, and I worked to see some of the PLOrk course sequence would count for Princeton CS and Engineering credit.

For his Ph.D. thesis in Computer Science at Princeton, Ge Wang created a new programming language called Chuck. It was designed from the ground up to be real-time, music/audio-centric, and super-friendly to inputs from external devices ranging from trackpads and tilt sensors to joysticks and music keyboards. Chuck was the native teaching language of PLOrk, and then SLOrk (the Stanford Laptop Orchestra, formed by Wang when he became a CCRMA faculty member), and many other LORks. It also was and is used for teaching beginning programming in a number of art schools and other contexts.

A few years ago, Ajay Kapur and I won an NSF grant for “A Computer Science Curriculum for Arts Majors” at the California Institute for the Arts. We created and crafted the curriculum, and taught it with careful assessments to make sure the art students were really learning the CS concepts. We iterated on the course, and it became a book (by Ajay, me, Spencer Salazar, and Ge). The course also be-

came a massive open online course (MOOC) whose first offering garnered over 40,000 enrolled students.

Now to Kadenze, which is a company Ajay, myself, and others co-founded and launched a year ago. Kadenze’s focus is to bring arts and creative technology education to the world, by assembling the best teachers, topics, and schools online. My *Real Sound Synthesis* topic is a Kadenze course offered by Stanford. The CalArts ChuckK course is there, as are courses on music tools, other programming languages, and even machine learning, all created for artists and nerds who want to use technology to be creative.

The genesis of Kadenze is absolutely STEAM. Artists need to know technical concepts. They need to program, build, solder, design, test, and use technology in their art-making. Engineers and scientists can benefit greatly from knowing more about art and design. Cross-fertilizing the two is good, but it is my feeling having both in one body is the best of all. Not all students need to get multiple degrees like I did, one in music, one (or more) in EE, but all of the names I mentioned in this short “STEAM teaching autobiography” actually are considered both artists and scientists by all those around them. They do concerts and/or create multimedia art works. They research and publish papers. They create both technology-based works of art and artistic works of code, design, and technology. The “Renaissance Person” can and should be. We need many more.

Specialization is necessary to garner expertise, but striving and working to become a skilled multidisciplinary generalist creates a whole person that can create, cope, build, refine, test, and use in practice. Plus, they can explain difficult concepts to novices, and carry the magic of combining art and technology to others. In other words, they are good teachers, too.

That has been my goal in life, and I think I am succeeding (so far).

ACM Fellow **Perry R. Cook** is Professor (Emeritus) of Computer Science, with a joint appointment in Music, at Princeton University. He also serves as Research Coordinator and IP Strategist for SMule, and is co-founder and executive vice president of Kadenze, an online arts/technology education startup.

© 2016 ACM 0001-0782/16/10 \$15.00