John R. Russell

was born April 8, 1929, in Nashville, Tennessee and received the AB *cum laude* from Princeton University in 1954. He attended the Ludwig-Maximilians-Universität in Munich in 1954-55 and the Phillipps-Universität in Marburg in 1957. In the same year he received the M.A. in German from Princeton, completing his Ph.D. in German at Princeton in 1966. Subsequently he taught at Wabash College and then at Centenary College. He joined the faculty of the Department of Germanic and Slavic Languages at the State University of New York at Stony Brook in 1965, where he was the co-developer of the Stony Brook Project for computer assisted instruction in German. Retired since 1993, he remains active in CALL.
Ruth H. Sanders

_Twenty Years of Computer Assisted Language Instruction:_

*Introduction*

Awe, fascination, disbelief, disapproval and fear — these are the mixed feelings that attended those early years of the Computer Revolution — roughly, the 1960s. In the first part of the decade, computers were still room sized whiz kids that whirred in the background of TV quiz shows and space operas. At the 1964 New York World’s Fair in Flushing Meadows, Queens, a popular exhibit featured a “handwriting analysis” of all comers by a computer the size of a pickup truck that was alleged to have gleaned from our mere signatures on a card the facts it printed out about us at the end of the line. In reality, of course, the information (which we had provided ourselves at the entry point) had been entered onto the same (punch) card we had signed, and the computer could no more read our handwriting than the household dog (who, on the other hand, knows a good deal about every person she meets without the aid of either handwriting or punch cards). But the long lines of good-natured fairgoers, unfamiliar with punch cards, were ready to be amazed by the computer, and so they were. Most of us never dreamed computers would play a serious role, up close and personal, in our lives.
By the mid sixties, however, they seemed to have taken over all record keeping, financial and otherwise, and became the scapegoat (often unjustly) for corporate errors of all kinds, a role they continue to play at this writing, though this seems to recede as the public gains in computer sophistication. In the late sixties the warm fuzzies cooled and hardened somewhat as computers achieved status as part of the Establishment. The Flower Power generation protested against them, associating them both with research supporting the Vietnam War and with a feared mechanization of society.

On college campuses this protest took sometimes violent turns, as in the early morning August 24, 1970, bombing of an Army computer research center in Sterling Hall at the University of Wisconsin, in which physicist Robert Fassnacht was killed. At my own alma mater, the State University of New York at Stony Brook, chemistry faculty members took turns maintaining 24-hour watch over the chemical computing facilities after a burglary during which the computer’s disks—at that time metal boxes about the diameter of automobile wheels—were stolen. Data disks, system disks and backup tapes were taken, so that not only chemistry research projects, but also the computer itself suffered a serious setback. The burglars had clearly been knowledgeable parties, bent on disabling the computer and destroying research records while ignoring other objects with more intrinsic value. It was at the time an unfortunately routine act of sabotage, repeated with variations across the U.S.

Though it is probably safe to say most college computing had little or nothing to do with war research, computers had acquired an association in the minds of many not only with war, but also with Big Brother: this, however, without entirely eradicating their earlier image as gentle giants. The general public wasn’t sure whether the computer was an Orwellian threat or the hope of the future.

It was against this background that I arrived at Stony Brook, in fall of 1970, as a graduate student in German. Almost immediately I got acquainted with the computer program that was the backbone of the first year German language course. The “Stony Brook Project” (codirectors John R. Russell and Ferdinand A. Ruplin) had already been in operation for five years by this time. We teaching assistants tried out the program, laughingly taking our lumps for “incorrect” answers we entered that weren’t really incorrect, only unanticipated by the program. When the
semester got underway, our students not only did their grammar exercises, but took their exams on the computer, saving us the trouble of grading them. Either the protestors who considered the computer an agent for the dehumanization of society weren’t taking German that year, or else they were subject to the ambivalence of the society around them, because we had no protests. In any case the computer could hardly be accused of dehumanizing the German program in the presence of small classes of ten to fifteen, taught, as I recall, with a mixture of ALM-type oral pattern drills, what might today be described as multimedia materials (the made-for-TV “Guten Tag” series and associated exercises), and reading-writing-vocabulary list techniques inherited from the even then much maligned (but on occasion perversely effective) “grammar-translation” method.

Initial funding for the Stony Brook Project had come from IBM; a few years after it started, other, even larger projects began (Suppes 1981). By 1969, as Robert S. Hart reports, with PLATO III in full swing at the University of Illinois, a more ambitious PLATO IV was proposed to and subsequently funded by the National Science Foundation, which at the same time funded the development of the TICCIT project at Brigham Young University.

PLATO was not just a CALL project, but a large, almost monumental effort that produced significant material in a wide range of academic disciplines, including foreign language, that continued for years and was eventually used in institutions across the country. One of the earliest effective CAI projects, it boasted by the early seventies long-distance access, a form of e-mail, and even conferencing, graphics screens, multimedia connections, and a lab at the University of Illinois with about 100 terminals just for foreign language learning. PLATO, however, provided some disappointments as well, most notably, as described by Professor Hart, in the form of the ultimate failure of the university’s association with Control Data Corporation, which was to license and distribute PLATO. Here was the first hard evidence that the values of the academic developers and those of the for-profit distributors would clash.

While PLATO operated on a paradigm in which the computing staff provided an authoring environment giving developers as much freedom as possible in lesson design, TICCIT got underway in 1972 with a different brief, according to Randall L. Jones. TICCIT was to be “learner centered,” which meant that the student could look at an on-screen
Overview of the contents of the program, then pick and choose which parts to work through, and could proceed at will through the body of the material. This was thought particularly appropriate for Brigham Young’s language student clientele, many of whom had learned basic language under field conditions as missionaries abroad. For them, the traditional text- and grammar-based programmed instruction approach didn’t seem suitable. The learner centered philosophy was to control all material written for TICCIT; the teacher-developers operated within the framework given.

At Stony Brook, IBM had proved willing to work with two new assistant professors of German who had no technical experience in order to produce a German CALL program that became the cornerstone of that large state university’s first year curriculum. Ferdinand A. Ruplin’s memories of working with IBM recall a wonderful world in which IBM provided hardware, long-distance connections to its lab in upstate New York, and technical expertise to the German subject matter experts, who wrote a textbook around the CALL material. For several years into the project, technical and human resources seem to have been plentiful.

As John R. Russell recalls in Barbara Elling’s interview with him, the horn of computer plenty began to close up in the early seventies. To many university administrators, the use of computers to teach language seemed extraneous at best: when it was unlikely to garner further external funding, CALL was left to wither on the vine. Societal ambivalence toward computers had reached academia, taking the form of accepting their relevance to mathematics and natural science, but not to foreign language. Proposals for CALL were likely to elicit one of two reactions from those who had the power to fund them, send their students to them, or approve curriculum for them:

A) Good idea, but you could never do it;
B) You could do this, but why would you want to?

Researchers in those mainframe days usually couldn’t simply proceed on their own, develop programs and look for funding afterwards, because, at most universities, working on the mainframes required an account to which the charges for using the computer had to be billed. Those who had neither external funding nor generous department budgets to absorb the cost were out of luck. And, at least in New York, budget constraints sometimes reared their heads in the most unexpected ways: a shortage of punch cards struck Stony Brook in the spring of 1975, while
I was taking a course in Introduction to Computer Science, and the cards were rationed (I seem to recall that the allowance was ten cards per day per student, although this now sounds improbably small). The rationing raised considerably the tension level in the lab (where we grimly punched our homework programs onto the cards with teletype-like machines), as the smallest entry error made the card unusable. Of course people hoarded cards, and sent their friends in to get cards for them. As an employee of the university library I dealt with this shortage by supplementation (I am recalling this with a guilty conscience) from the library’s supply, which was plentiful due to generous ordering a year earlier.

Meanwhile, CALL researchers, usually unable to attract interest from computer science departments for projects that seemed to offer no cutting edge challenges from a programming point of view, were left to their own devices in learning to use the mainframes. The results were often less than satisfactory, as reported by Dana Paramskas.

The desktop computer, with its at first tiny memory and limitations in display (often, uppercase letters only, no foreign diacritics, not to mention non-Roman alphabets), arrived on the academic scene in a serious way in the late seventies. Some developers benefited from the fledgling computer manufacturers’ willingness to donate a couple of machines against the chance that significant teaching projects developed for them would translate into a market among teachers and students. Others had bought their own machines as hobbyists and went on to develop learning materials on them. A few low cost “authoring systems” appeared (I am thinking primarily of PILOT here) and saved developers from the drudgery of learning to translate their pedagogical intentions into machine language or BASIC (whose limited capabilities were at least partially balanced by the fact that one or another version of it was issued free with virtually every computer).

Aided by a gift of hardware from Commodore, the Stony Brook Project took on new life in 1978 when it was transferred to microcomputers, as Professor Russell reports. Meanwhile, PLATO continued to flourish in a mainframe environment, nourished by a large support staff, the diversification of having materials for a wide range of disciplines, and a nationwide fan club of satisfied users.

But the advent of the personal computer brought further challenges. A host of brand names flooded the market, accompanied by rags-to-riches stories about the entrepreneurs behind each. Trying to
Thirty Years of Computer Assisted Language Instruction

decide which of these to hitch one’s CALL wagon to was a form of gambling that CALL developers soon learned came with the territory. Donald Loritz recounts his experiences with “small-and-simple-is-beautiful” in the Apple, which he valiantly harnessed in the service of natural language processing programs, at that time (1979) considered impossible to produce on anything less than a high powered mainframe. He recounts the change in atmosphere from the original Apple ethos favoring the individual user (often a hobbyist) to the commercialized, profit centered world of the high-end PC. Skeptical about the “improvements” inherent in huge memories and color displays, he views events as representing a lost opportunity for truly inexpensive student computers accessible to all.

Randall Jones, following a path that in retrospect seems a common one, had come to CALL, and to the TICCIT project, by way of philological studies pursued with computerized databases. He and his colleagues who were comfortable with statistical studies and symbolic representation began in numerical analysis of language features and continued by stretching their computers to recognize not just numbers, but letters and, hence, language. The path to this change was recognized as a rocky one as it traversed over the plain of non-Roman characters in European languages: the umlauts, digraph “s” (ß), and accented letters. These, incredibly enough by today’s standards, were once major stumbling blocks to acceptable CALL programs: Professor Jones witnessed in 1978 perhaps the first, but in any case an early, successful attempt to display umlauted characters on a microcomputer screen. Not that this solved the problem entirely: for years afterwards, would-be CALL developers were greeted with the same “blank stare” reported by Dana Paramskas when she inquired of hardware vendors about the possibility of accented characters (known in later times as the “extended ASCII character set”). By and large, the commercial market didn’t see any profit-making opportunity in, and hence spent no time at all in aiming its products at, CALL. The diacritics situation didn’t improve until the microcomputer revolution hit Europe some years later.

So it was left to the CALL establishment to provide for itself. As recounted by Frank Borchardt, this challenge was accepted at Duke by the designers of CALIS, and soon enough the CALIS authoring system provided no only accented European characters, but non-Roman alpha-
bets, and finally even ideographs. The CALIS project was probably the first to provide a comprehensive selection of language representation on screen for students and teachers.

Professor Paramskas describes her own transition from doubter to CALL developer in Canada, which followed a path to some degree separate from that of the United States: different computers, a different, because bilingual, society; and with its own funding difficulties. On a similar note are Gerhard R. Wazel’s sometimes wistful, often ironic recounts of tilting at the German Democratic Republic’s preunification academic bureaucracy, which on the one hand wished for a modern computerized educational establishment, on the other hand feared the tainted influence from the West without which, realistically speaking, progress toward that end was frustratingly slow. A convincing argument for the need for researchers to keep in touch with each other across political boundaries is provided by Professor Wazel’s description of his own conversion to CALL (which he calls catching the “Russell virus”) while on a sponsored tour to the United States. This included a stopover at Stony Brook and John R. Russell’s lab and home, where the guest room was thoughtfully provided with its own desktop computer.

Disciplinary boundaries are often just as hard to cross as political ones. Alton Sanders and I, a computer scientist and a linguist, respectively, describe some of the breakdowns in communication that occur when developers cross lines of specialization. The good news is that increasingly user friendly computing tools are making it possible to ease these border crossings, so that CALL can take advantage of expertise from more than one area even if the practitioners sometimes find it difficult to describe their work to each other.

Sebastian Künzel raises some doubts about where all this is going. Viewing CALL programs of the past and present, he is disappointed to see how many of them embody old, discredited ways of understanding teaching and learning. He calls for informed users and developers who are able to put into practice what psychology and other academic fields have found out about how we learn, and how language itself works.

How far we have come is visible from Thomas Kerth’s chronologically organized CALL bibliography, which reminds us of how much activity was going on in CALL decades ago, and how grounded today’s CALL is in past research.
Thirty Years of Computer Assisted Language Instruction

This volume is presented not as a comprehensive history, but rather as a partial collection that only recalls the flavor of the early days of CALL. A more nearly complete list of early developers would have to include, for example, CALICO founding father Frank Otto, Patrick Suppes, James Pusack, Sue Otto, David Weible, Karen Kossuth, John Underwood, Mike Bush, Carol Chapelle, and Alan Bailin. Among Europeans: Graham Davies, John Higgins, Tony Williams (K); Bernd Michael Goethals, Werner Daelemans, Yvan Rooseeleer, Fons Tuyaerts (Belgium); Françoise Demaizièrè (France) should be mentioned as well, as should many others whose accomplishments in the early days of CALL are notable even if not discussed in this volume. Quite a few are represented by their presence in Professor Kerth’s bibliography.

What does the future hold? Since my crystal ball is as clouded as everyone else’s, I decline to make predictions, except to agree with the proposition articulated by Frank Borchardt: CALL technology will ultimately be user driven. That is, if the language learners want CALL, there will be CALL. Given the overwhelmingly positive reception from learners during the past thirty years, through the thick and thin of popular attitudes toward computers, we will have CALL for a long time.

ACKNOWLEDGMENT

My thanks for editorial suggestions, advice, consultation, and encouragement to Michael Bachem, Frank Borchardt, Robert DiDonato, Nikolaus Euba, Eleanor Johnson, Charlotte Wharton, and William Wortman. Support for academic expenses was graciously provided by Miami University’s Office for the Advancement of Scholarship and Teaching. And to my husband, Alton Sanders, thanks for informed criticism and domestic tranquility in the face of editorial deadlines.