Experiencing Early User Interfaces

Martha E. Crosby(✉)

Department of Information and Computer Sciences, University of Hawaii,
1680 East West Road, Manoa, HI 96822, USA
crosby@hawaii.edu

Abstract. The intention of this article is to provide a slightly different perspective for the Women in Design, User Experience, and Usability discussion. This paper not only describes examples of human communication with counting artifacts and other early computing machines but it also recounts specific personal experiences with interfaces from the author's career of over fifty five years working in information sciences.

Keywords: Personal user experiences · Interface · History · Computer interfaces · Communication with computing artifacts

1 Introduction and Background

When did computer scientists begin to consider the human user as part of a computing system? When did the concept of a computing interface even surface? What innovations contributed to development of interfaces and when did they occur? This paper addresses these questions by describing some early interfaces of numerical computing systems including some that were personally experienced by the author.

The history of computing and interfaces is the history of human-kind's creativity and ingenuity. Throughout history people have used artifacts to augment their abilities, particularly with numerical processes. Even in the earliest systems, interfaces were necessary to communicate between the artifact and the humans needing to use it. Understanding significant events in the history of computing is important if one is to understand where computing concepts fit in a time continuum [1, 2].

Often what are thought to be new computing interfaces are adaptations of previous implementations. Knowing the history is a way to gain an appreciation for established concepts, notice repetitive trends, and make theories memorable [1–3]. Stories about the initial purpose of the artifact provide the rational that often accompanies design decisions and should be preserved. If the circumstances of the innovations are not recorded, they may be lost [1–3].

1.1 An Early Counting Interface

One of the earliest known recording artifacts involving a human user as part of a counting system was a Quipus (or Khipus). This artifact was a method of tying knots in ropes using a positional decimal system (perfected by the Incas) [4]. A knot in a row farthest from the main strand represented one, next farthest ten, etc. The absence of knots on a cord implied zero [4]. Caral, the largest recorded site in the Andean region with dates older than 2000 BCE, is the location where the earliest known quipus was found [4, 5]. The artifact found was a knotted textile piece that the excavators thought to be evidence that the quipus record keeping system was older than previously thought [4, 5].

The interface to the artifact was the human and over time the quipus functionality seemed to become more complicated. Evidence has emerged that the quipus may have also recorded logographic information in the same way writing does. The combination of fiber types, dye colors, and intricate knotting could be a novel form of written language [4, 5].

A Harvard anthropologist, Gary Urton, has suggested that the quipus used a binary system that could record phonological or logographic data. He claims that the quipus contain a seven bit binary code capable of conveying more than 1,500 separate units of information [4, 5].

1.2 Early Human-Artifact Interfaces

A very old "human to artifact" interface that is still in use today is an abacus. The function of an abacus is to help humans with mathematical calculations. The oldest surviving abacus, used by the Babylonians around 300 B.C.E., had an interface that consisted of the pebbles that were used for counting at the time but a modern abacus consists of rings that slide over rods [6].

A more recent human-artifact interface is the slide rule that was first built in England in 1632 [6]. Although it is rare to see them today, I personally depended on a slide rule from 1955 to 1959 when I was mathematics major at Colorado State University. The slide rule was still used by NASA engineers during the programs of Mercury, Gemini, and Apollo in the 1960's that sent men to the moon [6].

The earliest "human to artifact" interface could be really be called a "human to human" interface since "computer" was the job title for people with the task of performing the repetitive calculations that produced various types of numerical tables. People sat at counting tables using artifacts to facilitate their calculations. One of the most difficult aspects of doing large calculations with any artifact, whether it was a slide rule or an adding machine, was keeping track of the many intermediate results and using them correctly in the remainder of the calculation. The work was boring, error-prone, and created a need to simplify the tasks [6].

1.3 Punched Card Interfaces

In 1801, Joseph Jacquard invented a power loom to automatically weave a design on fabric. The loom used punched wooden cards, held together in a long row by rope [6]. The invention of the power loom met resistance as it put many of the operators out of work [6]. However, adaptations of the technology moved to the United States (U.S.) to help meet its 10 year census requirement. The first U.S. census of 1790 took nine months but by 1880 the population had grown to the point that the census took seven and one half years [6]. The census bureau saw the dilemma and offered a prize for a method that could facilitate the 1890 census. Herman Hollerith, an employee of the census bureau, inspired by Jacquard's punched cards, found an innovative way to automate the task. Hollerith eventually formed a company called the Hollerith Tabulating Machine that in 1896 became the International Business Machines (IBM) [6]. IBM grew rapidly and searched for a market for its cards that were processed using an assortment of their tabulating machines [2, 6]. In 1947, IBM commissioned a study to determine whether it should develop computing machines as one of its products. People were used to using tabulating machines for sorting tasks so some computerized work-flow processes needed only slight modifications. Although various man-machine communication devices existed, once IBM entered the computer market punched cards became a popular interface in the U.S. [6].

1.4 Communicating with Early Computers

During World War II, the British built an electronic machine called the Colossus. The Colossus was built for the purpose of breaking cryptographic codes. Although it was able to read coded German radio transmissions, it was not a general purpose computing machine, it was not reprogrammable and it relied on pulleys. In addition, the interface required a considerable amount of human physical activity [6].

Konrad Zuse was a construction engineer for an aircraft company in Berlin, Germany at the beginning of WWII. He had little knowledge of other calculating machines or their inventors with the probable exception of Leibniz, who lived in the 1600's [6]. Zuse built a series of general purpose computers to help with his engineering calculations. Between 1936 and 1938, in the parlor of his parent's home, he built the Z1 and improved it with the Z2. He made his own interface by punching holes in discarded movie film as paper tape was not available to him during the war [6]. In 1941, he built the Z3, probably the first operational, general-purpose, software controlled digital computer and improved it with the Z4. The Z1, Z2 and Z3 were destroyed during the war but the Z4 was saved because Zuse had moved it to the mountains [6]. Zuse also invented one of the first high-level computer languages called "Plankalkul." The Z machines were only known within Germany so were not considered influential in the development of other computing efforts. Yet the Z series architecture that consisted of a control unit, a calculator for the arithmetic, a separate memory to store the calculations and input-output devices for interfaces is still a fundamental design for computing systems [6].

In the U.S., between 1937 and 1942, John Atanasoff, a professor of physics and mathematics at Iowa State University, and Clifford Berry, his graduate student, built one of the earliest electronic digital computers called the ABC. Although the ABC was not programmable, lacked a conditional branch and worked on only simultaneous equations, it was innovative in that it used vacuum tubes instead of mechanical switches, and used a binary rather than a decimal system. It was also the first machine to store data as a charge on a capacitor; the method modern computers currently use to store information in their main memory [6].

In 1944, under a partnership between Harvard and IBM, Howard Aiken designed a series of Mark computers that were in use until the 1950's [6]. The Harvard Mark I was the first programmable digital computer made in the U.S. but it was not a purely electronic computer [6]. It was constructed out of switches, relays, rotating shafts, and clutches. The machine weighed 5 tons, incorporated 500 miles of wire, was 8 feet tall and 51 feet long, and had a 50 ft rotating shaft running its length, turned by a 5 horsepower electric motor. The interface consisted of four paper tape readers [6]. The Mark I noisily ran non-stop for 15 years [6]. Even though the Mark I had three quarters of a million components, it could only store 72 numbers [6]. In 1947 Aiken estimated that six electronic digital computers would be sufficient to satisfy all of the U.S. computing needs. Since only large institutions such as the government and military could afford these expensive machines, his prediction was not challanged [6].

The British Colossus, the Atanasoff-Berry Computer, and the Harvard Mark I all made important contributions. During the next decade, many specially built computing machines followed them. A computer primarily used to calculate weapon settings was called the Electronic Numerical Integrator and Computer (ENIAC). It was developed by John Mauchley and J. Presper Eckert at University of Pennsylvania and was operational from 1944 until 1955 [6]. After the ENIAC other early digital computers were the EDVAC, that used a stored program concept, included the JOHNNIAC (named after John von Neumann), and the Illinois Automatic Computer (ILLIAC) a series of 5 super computers, built at the University of Illinois between 1951 and 1974 [6]. The Universal Automatic Computer (UNIVAC), built in 1951, was the first commercially successful computer in the U.S. [6]. The wide variety of user interfaces in these early computing machines primarily involved physical interactions on the part of the users [6].

A cathode ray tube was sometimes used as a display interface. The cathode ray tube was one of the first random access storage device for digital computers. It was invented by Fred Williams at Manchester University in 1946 and was later used in the Manchester Mark I computer [7]. Any binary word in the display could immediately be read, instead of having to be accessed sequentially. Some of these tubes were made with a phosphor coating that made the data visible. The face of the tube was covered so the presence of the coating was not important to the operators. If a visible output was needed for an interface, a second tube with a phosphor coating could be used as a display device [7].

At the National Bureau of Standards (NBS) in Washington, D.C., a first-generation electronic computer was built in 1950. It was called the Standards Eastern Automatic Computer (SEAC). The SEAC went into production in May 1950 and may have been the first fully functional stored-program electronic computer in the U.S. [8]. Many modifications were added during its operation until 1964. Sometime the computer interface of the SEAC was a remote teletype, thus, it may have been one of the first on-line computers [8].

2 Personal Experiences Interacting with Computers

At this point, I can begin describing (as best I can remember), some personal experiences interacting with this generation of interfaces. I worked as a mathematician in Boulder Colorado for the Central Radio Propagation Laboratory (CRPL) of National Bureau of Standards (NBS) from 1959 to 1962. By then the SEAC had been moved to the Smithsonian and the NBS in Washington, D.C., was using an IBM 704 mainframe for computing. The CRPL in Boulder. had an IBM 650 with the storage capacity of 2 K decimal words and a compiler called SOAP. Instead of being sequential, the next instruction address had to be specified. Since the drum memory rotated, the most efficient next instruction address had to take into account the rotational speed of the drum memory. Each month the CPRL was responsible for producing contour maps representing the diurnal variations of the ionosphere for radio astronomers. These maps were done by hand and took approximately six months to complete. The time required to complete the task was an incentive to automate the process by designing the first computer-based use of numerical generated maps [9]. My task was to develop programs to run on the IBM 650 for this purpose. The average time that the IBM 650 would run without a problem (such as vacuum tube malfunction) was around two hours. As a result, it was essential to design the program as a connected series of computations with clear restart procedures.

The human-machine interface consisted of me, the human, starting the process with 3 toggle switches on the computer that loaded what was then referred to as the operating system. The next step was to put the first segment of the program into the card reader and wait for the intermediate results to be delivered on punched cards. A manual system had to be designed and maintained to organize the intermediate output. During this process, I had exclusive use of the computer so I moved into the computer room. Every two hours I retrieved and categorized the intermediate results. After 40 h, one complete contour map was generated and all the cards that were categorized for it were fed into the plotter. After the map was plotted, it was compared to the version of the map that was drawn by hand. There were some parts of the process that needed to be modified but, within a few weeks, it was clear that the ionosphere could be represented numerically. Even though the time required for the process to make one map was reduced from six months to 40 h, it was still a tedious process. Since the CRPL in Colorado was a branch of NBS in Washington, D.C., we were able to rewrite the program so that all the phases could be run on the 32 K IBM 704 computer at NBS. From 1959 through 1962, I refined the mapping programs on the IBM 704/IBM 709/IBM 7094 series as I commuted across the country. Although bringing the time it took to make one map down to 2 h, was a huge time improvement, the interface still involved human physical effort. The numerical data from the IBM mainframes was recorded on 24 inch reels of magnetic tape. I then had to take the tapes across Washington, D.C., to another government agency to have the numerical data on the tape converted to punched cards. The process was slow as one reel of tape produced approximately 80,000 cards that had to be packaged in boxes and mailed to CRPL in Boulder. Once the cards arrived, they were plotted and transformed into the format for the monthly predictions.

In 1962, I left CRPL to join Harry Diamond Laboratories (HDL), an Army research facility that was closely affiliated with the NBS in Washington, D.C. I was a hired as a research mathematician primarily to work on problems involving risk analysis and reliability. From 1962 to 1970, I wrote many mathematical and statistical programs. The interface with the computer still consisted of punched cards but the programs no longer had to be written in assembly language. HDL had an IBM 1410 with an early FORTRAN (Formula Translating System) compiler, a general-purpose programming language that was especially suited to IBM 704/IBM 709/IBM 7094 series and adapted for the 1410.

2.1 Experiences with Interfaces in Hawaii

In 1970, my husband accepted a two year assignment in Hawaii, never expecting to still be here 45 years later. Once it was clear that our stay in Hawaii was no longer temporary, I began working at a branch of the LTD Aerospace company called Kentron Hawaii. Kentron had the responsbility of writing the software and maintaining the data for the Missle Range in Kwajalein. We used the Control Data Corporation (CDC) computer that was physically located across town from the Kentron facility. We wrote programs in FORTRAN and sent the punched cards to the CDC facility by courier. We considered ourselves fortunate if we were able to see the results of our compilation the next day. If we were not extremely careful both writing the code and planning the steps for checking the results, we could spend several weeks trying to complete a project. Some of my colleagues at Kentron told me that the department of Information and Computer Sciences (ICS) at University of Hawaii (UH) offered a Master of Science (MS) degree. Although I had worked with computers more than 12 years, I thought a formal program would be an excellent opportunity to not only learn computer science theory but also be involved in the emerging discipline of computer science.

2.2 Interfaces at the University of Hawaii (UH)

In 1972, I took a job in the department of Oceanography at UH using the IBM 360 system to map the ocean floor and work on other geophysical projects. In 1973, I enrolled in the graduate program in ICS. As a graduate student, I had an opportunity to use the Berkley Computer Corporation (BCC) 500, a state of the art multi-processor computing system that was funded by DARPA to link Hawaii to the ARPANET [10]. The computer was given its name because it could support 500 interactive users. The architecture consisted of five independent processors, three levels of storage devices, a ninety bit fast memory, a drum, and large rotating disks. Figure 1 is a picture of the author holding one of the disks from the BCC 500. This combination of features gave the BCC 500 system nano-second capability, an extremely fast time for 1968 technology.

Communication and computing research began merging in the 1960's [10, 11]. At UH, Norman Abramson from the Department of Electrical Engineering (EE) transmitted

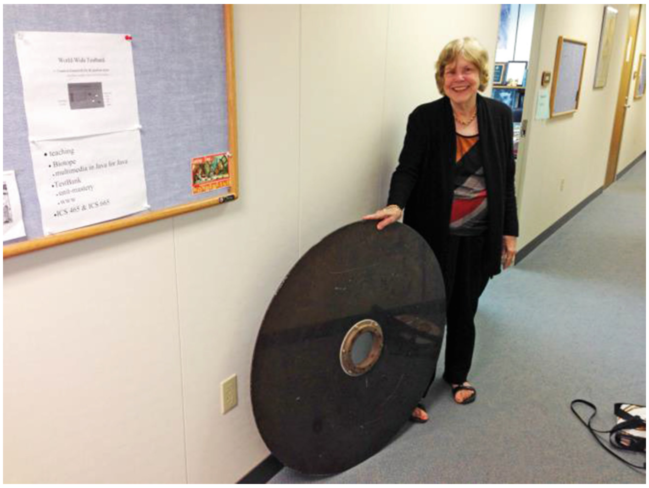

Fig. 1. A 48 inch computer disk from the BCC 500 computer

wireless data packets to a computer network across the campus in 1971 [12]. In the early 1970's, before the invention of personal computers, the ALOHA network opened the possibility for UH to deliver on-line education. High schools in Hawaii could not afford to purchase or lease computers but they could rent teletype interfaces for 32 dollars a month. I was able to teach FORTRAN classes remotely from UH to high school students in the neighbor islands and rural Oahu using teletapes to communicate with the BCC 500.

At this time, it was possible to buy components such as an Intel 8080 and, with some hardware experience, put together a computer. However, the ICS department only made a few if these. In order to teach students how to program microcomputers with our limited supply, we wrote an 8080 emulator for the BCC 500. The students were able to experience writing 8080 assembly language but had the advantage of using a teletype interface to write their programs on a large time-sharing computer that had speed and debugging capabilities. The slowest part of the process was the transferring the paper tape output from the BCC 500 to the Intel 8080. The students really enjoyed the experience of working on an early "personal computer" but they were happy to be have the advantages of the emulator on a larger time sharing computer.

2.3 User Experiences with Computing Interfaces at the University of Hawaii

After earning an MS. Degree from UH in 1975, I began teaching classes in computing. One of the courses that I taught was an Introduction to Programming Languages. This class included learning several types of computer languages that had very different interfaces such as SNOBOL (StriNg Oriented and symBOlic Language) and APL (A Programming Language).

SNOBOL was a series of computer programming languages with an emphasis on string manipulation. It was first developed between 1962 and 1967 at AT&T Bell Laboratories by David J. Farber, Ralph E. Griswold and Ivan P. Polonsky to symbolically manipulate mathematical expressions [13].

APL was originally invented in 1957 by Kenneth E. Iverson, a Harvard professor, as a matrix-oriented symbol system rather than a computer language. It emphasized array manipulation and used a graphical notation [14]. By 1966, after Iverson became an IBM employee, APL was developed into a programming language. The first computer implementation of APL notation was written in FORTRAN in 1965 as a batch-oriented language interpreter for the IBM 7090. An interactive version was written soon after using an experimental 7version of the 7090 and the TSM timesharing system. APL became more popular once IBM introduced an APL time-sharing version for the IBM/360, a completely interactive system [14, 15]. The programmer could type APL statements into a typewriter terminal connected to a time-sharing computer and receive an immediate response.

In the late 1970's and early 1980's other somewhat awkward to use interactive time-sharing systems became part of the UH computing environment. As Jodi-Ann Ito, the UH Chief Information Officer, wrote: "In 1977, when you walked into the Terminal Room, you had to know which system you wanted to use. The terminals were labeled with "HP2000", "TSO", or "APL" to indicate which system it was connected to. Each terminal had its own reservation sheet taped to the wall where you could sign up a week in advance to guarantee that you had time to work" [16].

Also in 1977, UH brought in PLATO (Programmed Logic for Automatic Teaching Operations), the first generalized Computer-Based Education (CBE) system that functioned for four decades [17]. In 1960, the original PLATO was designed and built on an ILLIAC I computer at the University of Illinois. By 1977, PLATO had grown to support graphics terminals distributed worldwide, running on a variety of networked computers [17, 18]. PLATO had a variety of functions such as forums, message boards, online testing, e-mail, chat rooms, picture languages, instant messaging, remote screen sharing, and multiplayer games and other functionalities that were innovative for the time [18]. PLATO had flat-panel gas plasma displays and was one of the first systems with touch panels built-in to the screen. Rights to market PLATO as a commercial product were licensed by Control Data Corporation (CDC), the manufacturer on whose mainframe computers the PLATO IV system was built [18]. At UH, specialized PLATO terminals were used by faculty to enhance their classes. Lessons were developed to teach many subjects one of the most innovative being teaching Kanji characters to students in Japanese language classes. PLATO supported hundreds of students in innovative classes until it was retired from UH in 1995.

2.4 Interfaces with Personal Computers

In the mid 1970's, the prevalent idea was that the future of computers would become a utility or service, run either by the government or controlled by major corporations that could afford them. Leaders in the computer industry were either designing bigger and faster mainframe computers or special purposes computers designed with specific functionality. In 1971, one of these special purpose computers built for a chemical company failed to meet specifications so the manufacturers advertised their product in Popular Electronics, a magazine read by many computer enthusiasts. The computer was very primitive but the price finally made it possible for individuals to own a computer. At that time, very few people would have predicted the personal computer movement that followed during the next decade or that anyone could actually own a personal computer. The computer industry now had an incentive to promote an easier and more transparent way for ordinary citizens to communicate with computers. By1981 Xerox had built the Star and it was quickly followed by the Apple Lisa in 1983, the Apple Macintosh in 1984, and Microsoft Windows in 1985 [6].

The ICS department at UH began building a personal computer laboratory. We bought the components such as random access memory, integrated circuits, 8 inch floppy disk drives separately. The assembled computers used the CPM operating system and had a bios that some faculty designed and wire-wrapped. The interface on the computer monitor primarily used a Wordstar line editor to write programs for a PL/1 compiler written by W. Wesley Peterson, an ICS faculty member.

In the 1970 s and early 1980 s, home computers were made useful by the programming language BASIC (Beginner's All-Purpose Symbolic Instruction Code). It was invented in 1964 by John G. Kemeny and Thomas E. Kurtz and initially ran on a General Electric computer system at Dartmouth College [19]. Because of limited memory space, BASIC became the primary language for personal computers. It was an interactive programming language with a text-based interface.

Another language adapted for personal computers was LOGO, an educational programming language, designed in 1967 at MIT by Daniel G. Bobrow, Wally Feurzeig, Seymour Papert and Cynthia Solomon. The original purpose of LOGO was to teach college students programming concepts but the turtle graphics feature of the language made it an ideal interface for introducing young children to computing [20]. In 1983, the ICS department donated an Apple IIe computer to a local grade school. The computer was housed in the school library and small groups of students were taught LOGO with great success. A volunteer teacher designed a curriculum for the students, based on graphics such as producing pictures and engaging in educational games.

Successful hypermedia systems were developed prior to the introduction of the World Wide Web. One of these systems was an application called HyperCard. It was primarily a programming tool for Apple Macintosh and Apple IIGS computers. HyperCard also had a programming language called HyperTalk that allowed the programmer to quickly prototype user interfaces. In 1988, Jan Stelovsky, a faculty member in the ICS department at UH, developed Kanji City, a simulation of a real-life environment for language instruction. He used HyperCard's ability to integrate text with digitized and synthesized sound, interactive graphics and animation to make an effective teaching platform [21]. We were able to use modified versions of Kanji City to test the extent to which Hypermedia was a facilitator for retention of Kanji characters [22].

2.5 Experiences in Human-Computer Interaction

In 1979, I enrolled in the Ph.D. program in the department of Educational Psychology (EDEP) at UH. My goal at the time was to better understand the difficulties ICS students had in understanding algorithms. My dissertation research was on human comprehension of computer programs and I compared how humans process algorithms using natural and computer languages. After completing my doctorate in 1986, I continued teaching in the ICS department at UH. With backgrounds in mathematics, computer science and educational psychology, my research gravitated toward the then emerging field of Human-Computer Interaction (HCI), at the time one of the fastest growing sub-fields within computer science. I began doing research in the areas of the human use of computing systems, individual differences of users, cognitive styles and the evaluation of innovative educational environments.

2.6 Use of Sensors

In 1984, Peter Dunn-Rankin, a faculty member in from EDEP at UH received a grant to purchase an eye-movement monitor. The state of eye- tracking technology at that time was such that it required a few years and the expertise of W. Wesley Peterson to verify the eye-tracker accuracy and write appropriate software to collect the data and visualize the results. By the late 1980's, I began using eye movements to investigate computing and interface problems. In addition to studying how people read algorithms [23], we performed several experiments on how they searched lists [24], and how they viewed data models [25]. In 2007, my colleague, Curtis Ikehara, and I received a patent on an "input devise to continuously detect biometrics" where physiological data is examined at four critical points of a task: pre-task physiological resting state, the initial physiological response upon starting the task, the physiological response to increasing task difficulty and the physiological response at task completion. We have used these measures to determine several potential indicators of cognitive load and found them more sensitive to interaction effects with task difficulty than other task performance measures. We extended this work to the use of other physiological measures such as heart rate, electro-dermal activity, temperature and the pressure applied to a computer mouse [26, 27]. It was our objective to create a set of passive physiological sensors that could provide real-time cognitive state measures.

Physiological measures can provide information on the cognitive state of the individual. In a pilot study conducted at our research laboratory, students were shown a variety of word and mathematical problems. Some of the problems were simple while other problems were ambiguous. Although performance was relatively similar, the physiological response was different on a variety of problems in unexpected ways. One participant started to excessively fidget when presented with an ambiguous word problem as detected by the large variance in readings from the blood flow sensor connected to the subject's finger. English was not the subject's first language, so these word problems caused great consternation. Another subject performed well on all questions, but the electro-dermal sensor detected a sudden increase in perspiration upon seeing a simple math question. When we debriefed the student, we found that this type of math problem brought back negative childhood memories [26]. Although task performance of this moderately difficult task was unimpaired, it could cause degraded performance in a critical task situation if it is left unchecked. In high stress or mission critical situations, these internal individual distractions can reduce the needed full attention required for accurate and timely task performance. Identifying and mitigating these responses in a low to moderate stress environment at critical points in a task is preferable to a identifying these individual differences in a high stress or mission critical environment [27].

3 Conclusion

Most of these examples of computing environments that I have described took place as the computer field was rapidly expanding. Although the implementations are always changing, the theory remains steady. Early predictions about the future of computing and interface needs have rarely been accurate. By paying more attention to the past we may become better at predicting the future. When I entered the computing field in 1959, there were very few electronic digital computers and the idea of owning your own computer was barely considered to be a possibility. The question of whether computers should be a service or utility was never asked, it just developed differently than expected. Years ago, John Von Neumann questioned whether there would ever be the need for more than 50 memory cells of storage. In a recent New York Times article about mobile devices and their interfaces [28], Nick Wingfield stated: "I can't imagine personally needing much more than a terabyte of online backup – it is more than 300,000 photos or 1,000 h of video." But perhaps being conscious of previous inaccurate predictions, he qualified this statement by continuing "But I might get there someday as the resolution in cameras increases [28]".

Through several years my work has progressively involved improving the interfaces for human needs for smaller and easier to use devices. My history communicating with computers through different generations of interfaces has experienced a variety of innovations. I began working with large machines with difficult interfaces but not as tedious as the ones prior to my entry into the field.

References

1. Crosby, M.E.: Using events from the past to inform the future. In: Tatnall, A., Blyth, T., Johnson, R. (eds.) HC 2013. IFIP AICT, vol. 416, pp. 144–148. Springer, Heidelberg (2013)
2. Crosby, M.E.: Looking back. In: Tatnall, A. (ed.) Reflections on the History of Computing. IFIP AICT, vol. 387, pp. 108–114. Springer, Heidelberg (2012)
3. Giangrandi, P., Mirolo, C.: Numerie macchine: a virtual museum to learn the history of computing. In: Proceedings of the 11th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education, pp. 78–82. ACM, Bologna, Italy (2006)
4. Urton, G.: From knots to narratives: reconstructing the art of historical record keeping in the andes from spanish transcriptions of Inka Khipus. Ethnohistory **45**(5), 409–438 (1998)
5. Beynon-Davies, P.: Informatics and the Inca. Int. J. Inf. Manage. **27**, 306–318 (2007)

6. Kopplin, J.: An Illustrated History of Computers Parts 1, 2 and 3. http://www.computersciencelab.com/ComputerHistory/History.htm, http://www.computersciencelab.com/ComputerHistory/HistoryPt2.htm, http://www.computersciencelab.com/ComputerHistory/HistoryPt3.htm (2002)
7. Lavington, S.: A History of Manchester Computers, 2nd edn. The British Computer Society, Swindon (1998)
8. Astin, A.V.: Computer Development (SEAC and DYSEAC) at the National Bureau of Standards Washington D.C., vol. 551. National Bureau of Standards, U.S. Government Printing Office, Michigan (1955). Accessed 25 January 1955
9. Hinds, M., Jones, W.: Computer program for ionosheric mapping by numerical methods. NBS Tech Note **181** (1963)
10. Anderson, D.: The future of the past. Commun. ACM **55**(5), 33–34 (2012)
11. Ryan, J.A.: History of the Internet and the Digital Future. Reaktion Books LTD., London (2010)
12. Abramson, N.: The Aloha system. In: Abramson, N., Kuo, F. (eds.) Journal of Computer Communication Networks. Prentice Hall, New York (1973)
13. Gimpel, J.: A theory of discrete patterns and their implementation in SNOBOL4. Commun. ACM **16**(2), 91–100 (1973). http://doi.acm.org/10.1145/361952.361960
14. Iverson, K.: A personal view of APL. IBM Syst. J. **30**(4), 582–593 (1991)
15. Falkoff, A., Iversion, K.: The evolution of APL. SIGPLAN Not. **13**(8), 45–57 (1978)
16. Ito, J.: A History of Interactive Timesharing at UH. http://www.hawaii.edu/infobits/s2000/interact.html (2000)
17. http://www.en.wikipedia.org/wiki/PLATO_%28computer_system%29
18. Smith, S., Sherwood, B.: Educational uses of the PLATO computer system. Science **192** (4237), 344–352 (1976)
19. http://time.com/69316/basic/
20. http://en.wikipedia.org/wiki/Logo_%28programming_language%29
21. Ashworth, D., Stelovsky, J.: Kanji city: an exploration of hypermedia applications for CALL. CALICO J. **6**(4), 27–50 (1989)
22. Crosby, M., Stelovsky, J., Ashworth, D.: Hypermedia as a facilitator for retention: a case study using Kanji city. Comput. Assist. Lang. Learn. **7**(1), 3–13 (1994)
23. Crosby, M., Stelovsky, J.: How do we read algorithms? a case study. IEEE Comput. **23**(1), 24–35 (1990)
24. Crosby, M., Peterson, W.: Using eye movements to classify search strategies. Proc. Hum. Factors Soc. **2**, 1476–1480 (1991)
25. Nordbotten, J., Crosby, M.: The effect of graphic style on data model interpretation. Inf. Syst. J. **9**, 139–155 (1999)
26. Ikehara, C., Crosby, M.: Assessing cognitive load with physiological sensors. In: Proceedings of the 38th Annual Hawaii International Conference on System Sciences (HICSS 38) (2005)
27. Crosby, M., Ikehara, C.: Using real-time physiological monitoring for assessing cognitive states. In: Ghinea, G., Chen, S. (eds.) Digital Multimedia Perception and Design. Idea Group Inc., Hershey (2006)
28. Wingfield, N.: Microsoft Has Suddenly Gotten Serious with Mobile. New York Times, Business, p. B8 (19 February 2015)