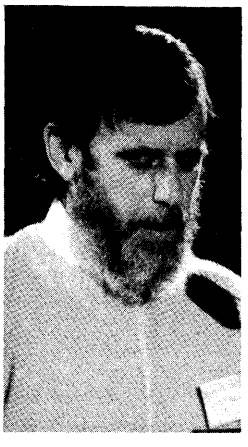
SESSION 2: 10:45–11:45

A PERSONAL VIEW OF THE PERSONAL WORK STATION—SOME FIRSTS IN THE FIFTIES

Doug Ross
Introduction by Severo Ornstein



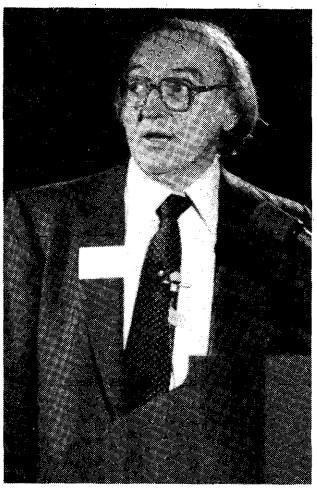
Severo Ornstein

My name is Severo Ornstein. I've fooled with computers for a little over 30 years and, in fact, I was at Lincoln Laboratory at about the same time as our next speaker. So I've been asked to be the discussant for him.

Hindsight is supposed to be easier than foresight, but it occurs to me that in my experience, hindsight is the hard thing. You get a bunch of computer people together arguing about which way things ought to go, and the arguments can be pretty fierce, but they don't ever seem to be as fierce as the arguments about how things went. I suspect there is going to be more of this as time goes on.

We tend to think of the history of a field as commencing more or less when we got into it ourselves. Everything before was somehow a given when we came along, and so we're really only interested in history from the point where we got into the field. But, of course, it's not really that way. Everything's got an antecedent.

Last night I read through Doug's paper. Although Doug and I were at Lincoln Laboratory at about the same time and must have passed one another a thousand times in the hall, we never actually met until this morning. I was, of course, aware of his work. Reading his paper took me back in a number of ways. First of all, it was lovely to run into familiar names of people and places, conjuring up all sorts of memories. One of the reasons we're here is that we like that a lot; it



Douglas T. Ross is Chairman of the Board of SofTech, Inc., in Waltham, Massachusetts. Doug has been with SofTech since its inception, serving as President from 1969 to 1975 when he was appointed Board Chairman. From 1979 to 1981 he was also Chairman of the Board of SofTech Microsystems, Inc., in San Diego, California. Previously, he held various posts at the Massachusetts Institute of Technology, first as teaching assistant (1951–1952), then head of the Computer Applications Group (1952–1969) as well as lecturer in the Electrical Engineering Department (1960–1969). He continues as a lecturer in the MIT Electrical Engineering and Computer Science departments, where he is a Member of the Life-long Continuing Education Committee.

Educated in mathematics (A.B., Oberlin College 1951) and pure math, and electrical engineering (1951–1956, with M.S., MIT 1954), Doug has published extensively on aspects of automatic programming of numerically controlled machine tools, symbolic manipulation and numerical calculation, CAD systems, software engineering, high-level programming languages, and structured analysis. He served as an editor of the *IEEE Transactions on Software Engineering*, and is currently editor of *SOFTWARE: Practice and Experience* and on the Editorial Board for *Computers in Industry*. He has won many awards, including the 1980 Society of Manufacturing Engineers (SME) Distinguished Contributions Award and the 1975 Joseph Marie Jacquard Memorial Award of the Numerical Control Society, and he authored the book, *Introduction to Software Engineering with the AED-O Language* (Waltham, Mass.: SofTech, Inc., 1969).

locates us, in some sense, in history. In the 1950s, a lot of the ideas that underlay the work that has followed since were brewing in people's heads. As I read Doug's paper last night, I realized how much of that there actually had to be. We tend to think in terms of artifacts as we look back, but what happens is some strange mixture of artifacts and ideas—ways of thinking. It appears that Doug was very influential in seeing what was available in the 1950s and recognizing possibilities growing out of that into the future. The hardware that he had available at that time looks very primitive, but the ideas were very, very far-reaching. It seems to me that the impact on what has happened is just enormous. In reading his paper I suddenly realized how revolutionary

the thinking was that was going on at that time. We take it all for granted now, but it was really revolutionary then.

Did you ever notice when you look at pictures of the old computers (ENIAC or something of that era) that people are standing up in front of them? One of the notions that Doug had was that of a partnership between a person and a computer. If you're going to enter into a partnership with somebody, you can't just stand there all day long, you eventually have to spend a lot of time together. It used to be that people sat down only when they wrote programs, off somewhere else while they were thinking through the problem. When they actually got to the computer they stood up and someone took a picture so that we could all sit here today and look at those pictures. But Doug envisioned this idea of a partnership in which you actually sit down and work at the machine. So I like to think that what Doug has done is to make it possible for the rest of us to sit down. True to form, he's going to do that for me right now.

A Personal View of the Personal Work Station
Some Firsts in the Fifties

Douglas T. Ross
SofTech/MIT

Introduction

"What the h--- kind of program do you *have* there?" the intercom blasted forth, the sound distorted by the intensity of the shout. It was the chief computer operator in Test Control (as the standard control area for the Whirlwind I Computer was called) calling me in great agitation. I couldn't reply at once, for I was some twelve feet away from the loudspeaker of the E31 console—in the far back corner of the secret and darkened room 222 of the Barta Building at MIT. I was hunched over a 16-inch oscilloscope called the area discriminator. It was mounted vertically in a box on the floor, so its face made a horizontal flat surface. Adhered to the tip of my moving finger was a bright, glowing displayed spot of blue-white light, about 1/4 inch square. I was in the process of writing my name into the computer with that spot—freehand. It was sometime in the fall of 1954.

It was one of the few programs I ever wrote that worked the first time. I remember that I dreamed it up on a long flight back from Texas, which must have been at most a few days earlier. Fewer than 200 instructions were required, and the hardest part was understanding the octal constants that calibrated the scope display coordinates.

MY SOURCES

This is my second foray into the tribulations of writing (rather than making) history. In 1977 I worked for months on my paper "Origins of the APT Language for Automatically Programmed Tools" for presentation at the History of Programming Languages Conference, June 1-3, 1978, in Los Angeles (Ross 1977). As was the case for that paper, I have taken as my primary source my extensive personal collection of archival working papers, reports, and records, which I have retained over the years, and I have attempted to include only observations I can directly support from these records. Unfortunately, I am less sure than was the case for APT whether or not my current story is complete, however, for the early portions hinge on older material that is less organized and perhaps less complete. I say "perhaps," because I cannot determine whether certain materials I remember seeing are missing, or merely are misplaced. At one point in SofTech's impecunious past,

rather than buying needed file cabinets, all of my historic files were without my knowledge summarily crammed into a large number of storage boxes whose length dimension was more than one file drawer. I still have not been able to retrace what was where, as useful chinks in the boxes were filled here and there by segments of unrelated files! In later years I may be able to fill in more details, as relevant portions come to light, but even so, what I have found already yields an engrossing trail.

As in the APT history paper, to provide concise reference to the materials in my files (which ultimately will reside in the MIT Archives) without spelling out each item I have used the following condensed notation: [C540123] means "correspondence 1954 January 23"; [R56123] means "daily resume (a form of personal professional diary) 1956 December 3"; and [WW2Q55p10] means "Whirlwind I Quarterly Report Second Quarter, 1955, page 10" (see Project Whirlwind 1952–1957).

THE THEME

I am not an historian at all, and can't bring myself to write a straightforward (dull) recounting of events. For these stories from the past I feel I must have a theme to provide some perspective. Also, by writing in the first person, I can try to give some of the feel of the times in ways that I hope add interest. But also I think such a style allows a more accurate interpretation of what actually was in the air as the documented events took place, even though surely all that is written is biased by recollections, as well.

I take the unique event of the first hand-drawn input to a computer as the starting point of my theme for this paper—that *The Personal Work Station as a working reality (rather than a speculative idea) did have a recognizable era of beginning more than 30 years ago.* Like the present era, where the physical and performance characteristics of the *work station* aspect of the PWS are so essential, the fact that Whirlwind *had* the Area Discriminator scope was an essential ingredient of my successful program. But my theme is much sharper than that, and in a sense even contradicts the natural emphasis on the *work station* characteristics which dominates most writing on the topic. My real theme is that *it is the revolution in personal work* (i.e., explicitly the PW, rather than the WS, of the PWS) *that was, and is even more so today, what is important.* (This also is why, throughout this paper I purposely spell PWS with three words rather than the accepted (correct) two, to stress my view.)

I hope to show that from the very beginning, the changes in personal working style, made possible by the idea that man and machine can *share* in the problem-solving process, is what makes the personal work station idea a unique and valuable departure from the customary

view of using computers. With the PWS idea, the computer becomes a partner not a mere tool. And to integrate and actually realize this PW and WS idea, requires a *systematized treatment* of the software that links man to machine. That is the third component of my theme.

OVERVIEW

In the time available to me, I can trace only the beginnings of this theme. As was the case with the APT story, I find that my records show that this PWS story breaks "naturally into a number of overlapping but nonetheless distinct *periods* during which the focus of attention highlighted first one and then another and then another aspect of the overall subject" (Ross 1977, p. 282).

The *first period* covers my introduction to Whirlwind and programming, in the summer of 1952, and brings out that even with all its rooms full of equipment in its own building at MIT, Whirlwind was a miniscule microcomputer by today's standards for personal work station capacity. And to start with, Whirlwind lacked some needed features, as well.

The *second period,* from August 1952 through August 1953, saw the flexibility and capacity of Whirlwind expanded, as facilities geared up for the focus on the development of the Cape Cod System—the R&D base that led to the full-scale development of the SAGE Air Defense System by Lincoln Laboratory (Redmond and Smith, pp. 201-213). Most of the environmental equipment for PWS evolution now was present, but my available records don't indicate whether or not I yet had discovered the wonders of the mysterious room 222.

In any case, well before the summer of 1954, during the *third period* when I completed my master's thesis (Ross 1954b) and our work on large-scale data reduction programs (Ross 1953b) actually got under way (which was the driving force for my own insights and endeavors), I did gain access to the classified areas of the Barta Building and knew that we simply *had* to incorporate the marvels of *manual intervention* (or MIV as it was then called) into our system capability. The problem was that I didn't know how to program for all that equipment.

The problem was solved in the *fourth period* (June 1954 through early 1955) by hiring Bill Wolf from Lincoln Lab. Bill had done some of the early MIV programming for Lincoln. His early departure forced me to learn what I needed to know as I completely rewrote his code to conform to my plans, which were more general than Lincoln's use of the same WS equipment. This also, of course, was the period when I wrote the *Scope Input Program* (Ross 1954d). Another innovation triggered by my needs was the creation of the Director Tape utility program [WW3Q54p7], the first real *operating system* command language

system (to use present-day terms) to eliminate the computer operator function for my elaborate, multi-tape runs. By the third quarter of 1954 my plans for MIV use were quite general and well developed.

The *fifth period* focuses on the preparations for and delivery of two Servo Lab symposia at MIT. The first, held on March 8 and 9, 1955 on "Design and Evaluation of Bomber Fire-Control Systems" (MIT Servo Lab 1955) covered our project's entire hardware and software system for complex system testing. The second, on June 1 and 2, 1955, covered just the MIV-controlled data reduction methods (Ward and Ross 1955). Both symposia demonstrated the new techniques to large groups from outside MIT. My PWS ideas were now a working reality, in this particular setting.

The *sixth period* concerns my first professional paper, "*Gestalt Programming: A New Concept in Automatic Programming*" (Ross 1956b), which I presented as the opening paper at the Western Joint Computer Conference in Los Angeles, February 7, 1956 (WJCC 1956). I actually had proposed the paper in November 1955 [C551117, to B.J. Bennett, Prog. Chrm.] and had presented a preliminary version at MIT on January 11, 1956 [C56015], but as I recount here, it was a long process to arrive at a reasonable formulation of the ideas in acceptable form, at the beginning. I consider that paper to be my own definitive statement of the PW theme of this current PWS paper, but written at that earlier time.

The *seventh period* actually overlaps periods two through six, in time, for it concerns the long period when, with John Ward and others on the Fire-Control System Evaluation Project, we formulated, proposed, designed, and then assisted in the installation of a full-scale *Charactron-based MIV Console for the Univac 1103* computer at Eglin Field Air Force Base in Florida for the evaluation of the B-58 "Hustler" supersonic bomber tail turret (Ward and Ross 1956). That the actual testing would be done at Eglin had been known since at least early 1954. I had done a special Charactron demo for the June 1955 symposium, and the MIV console itself finally was installed in 1957 [C571024, J. E. Ward to J. L. Moser, Stromberg Carlson]. The Whirlwind facilities were duplicated, and their actual circuit drawings were used in the design, thanks to Lincoln Lab cooperation.

Even the Whirlwind setup was not complete as yet, however, and the *eighth period* mentions briefly my proposal (March 1956) (Ross 1956d) for direct Flexowriter *keyboard input* to Whirlwind to complete our MIV facilities. Until that time, the Whirlwind Flexowriters had been hooked up and used only for printed (or punched) *output* and only punched *tape input*. Barriers to the gestalt programming concept, both conceptual and physical, were very high in those days, and even when the flexo input was available, its use was a far cry from the cen-

tral role that keyboards play in today's PWS and word processing practice. We lacked the computing capacity for full on-line keyboard use.

The flexo input was just one part of the facilities incorporated into the *SLURP System* (Servo Lab Utility Routine Programs) for large-scale, experimental program development (Ross 1958), which is what the gestalt programming concept then had evolved into. It was made possible by the luxury of the expansion of Whirlwind's magnetic core memory to 6K 16-bit words (yes, that's 12K *bytes*, in today's terminology!) with the *cf*, change fields, instruction to select any two 1K banks of words at one time [WW3Q56p44]. SLURP was the first complete interactive software development environment, with many programming, language development, debugging, and display tools all integrated into an interpretive environment with *group control* for core versus drum storage management, and automatic logging and log playback of all MIV actions. Reminiscent of today's debate regarding menu control and how many buttons to put on a mouse, a universal "MIV Box" subroutine gave complete control over these hierarchic facilities with only *major and minor exit buttons* and one *wait switch* for stepping and selecting from one control level to another. From today's vantage point, I can't imagine how we did all that in just 12K bytes with only a few (two or three?) times that much magnetic drum backup. But we did. I take SLURP to be the systematized solution, third component of my PWS theme in this paper. Development continued until the demise of Whirlwind in 1958.

The *ninth period* artificially cuts off abruptly at the start of 1960, to match my title and put a stop to burgeoning growth of what might be covered. Its beginning overlaps the other periods of focus, with the plans and actions to carry on MIV work on the IBM 704, 709, 7090, and 7094 computers at MIT, after Whirlwind's demise. Although artificial, this actually is a good stopping place, because other, broader themes spring from this period. Timesharing, computer graphics (both hardware and software), computer-aided design, software engineering (both language and practice), and software technology tools and methods all blossomed and flourished at MIT in the 1960s, and have direct ties to the story I present here. But those are tales for another time.

Preparation

Significant events always are the rearrangement into new forms of otherwise ordinary and insignificant, routine happenings. The ordinariness allows the significance to be recognized. What was the environment, and what was going on at Whirlwind that triggered a new and more intimate kind of computer use? What were the pieces from

which the PWS idea began in this story? The preparation for my theme also is the preparation stage of my story.

PROGRAMMING BACKGROUND

I came to MIT in the Fall of 1951 as a teaching assistant in the mathematics department, having received an A.B. Cum Laude in math from Oberlin College in June. Pat and I had been married the previous January, and she was the first of a group of "computers" hired by Lincoln Laboratory (Badge No. 161), which was just getting under way, performing calculation and graphing assignments for the engineers, using Marchant desk calculators. One of her assignments involved using a mechanical correlation computer, designed by Norbert Weiner, which was in use in the Servomechanisms Laboratory at MIT (Ward 1954). When I needed a summer job for 1952, I applied to the Servo Lab, and was hired also as a "computer." One of my first jobs involved checking some anomalous points that had been obtained on that mechanical correlator, and someone suggested that I see if Whirlwind could be used, for they knew that Jack Arnow had written such a program. Jack later founded Interactive Data Corporation (now part of Chase Manhattan Bank) as one of the early timesharing service companies.

I, of course, had never heard of Whirlwind or of *real* computers, so I was fascinated. I still have Jack's little program, plus the Digital Computer Lab application form for Problem #87 (July 2, 1952) to get computer time, some scribbled yellow papers where John Frankovich (still working at Lincoln Lab) gave me my first programming pointers, and the "Tape 1414 Mod O" July 11 printout of my autocorrelation program. My earliest run request (July 18) shows that I was already up to Mod 3 one week later (Ross 1952).

Before I could complete the debugging of my rather elaborate program (which later was used in many projects and theses by others) Whirlwind was completely shutdown from August 11 through August 30, 1952! [WW3Q52p6] I used the time to write a matching Fourier transform program so that power density spectra could be computed (which also was widely used, and formed the basis for an improved program for my own master's thesis in 1954). The reason for the shutdown is the real starting point for my PWS theme—for in that period the entire input/output system for Whirlwind was drastically modified to provide the elaborate hardware facilities needed for Whirlwind's new and primary Lincoln Lab mission: high priority development of an air defense system for the United States and North America.

The Whirlwind order code (of 32 instruction types, maximum) was revised so that *si, so, rd, rc, bi,* and *bo* for "select input", "select out-

put'', ''read'', ''record'', ''block in'', and ''block out'' could be used for any one of a large number of input/output lines to which analog scopes, mechanical switches, ''activate'' (one-shot) pushbuttons, indicator lights, and magnetic drums could be attached. The mechanisms for general *manual intervention* in the running of programs were at hand.

Only the barest essentials trickled out to the programmers of the user problems. Most of the subsequent construction took place behind the locked, green, double doors of Room 222, which was strictly off limits. With the end of the summer at hand, and with my important programs in limbo and my enthusiasm for programming running at white heat, my supervisor, John Ward (still at MIT) asked me whether I really wanted to go back to teaching freshman calculus, or would I rather make a career for myself in Servo Lab, full time, taking courses as a special student. I felt obligated to fulfill my 1952 fall teaching assignment, so we arranged for me to do that in addition to working full time programming and taking graduate math courses for that semester. The math department wasn't happy to lose me. I had been so serious and successful about my teaching assistant role that I had been appointed as the only graduate student member of the Dean's group of freshman advisors. But I remained with the lab until August 1969, when my colleagues and I left to found SofTech.

PROJECT BACKGROUND

The Air Force-sponsored project headed by John Ward was concerned with the test and evaluation of the accuracy and performance of airborne fire-control systems—specifically servo-controlled tail turrets in bombers. The autocorrelation and Fourier transform programs were part of the study of radar noise analysis. I was the only mathematician in the laboratory, and one reason for hiring me was that a new form of air-mass ballistic tables had just been developed (Nielsen and Heyda 1951, p. 9), more accurate than the old tables that had been developed for ground artillery and required extensive compensation to account for rapid movement through rarified air. My first major assignment (November 1952) was to see how these new tables could be used for the evaluation problem.

By January 12, 1953, I had a complete data flow diagram (Ross 1953a) for hit probability density calculation, based on fourteen measured inputs, and using vector calculations well suited to the three-dimensional problem and digital computing. With further elaboration and refinement over the next five years, the complexity of actually programming and operating this solution for production data reduction

(Ross 1953b) was the stimulus for all my MIV developments until the advent of the Computer-Aided Design Project in 1959.

The main part of the project was directed toward the development of airborne flight test instrumentation that could measure the many variables during mock attacks by a fighter aircraft pursuing a bomber. With the feasibility of the digital computer data reduction established, attention focused on digital instrumentation (primarily shaft encoders to convert selsyn-repeated analog quantities into Gray-coded binary) for recording on a flyable magnetic tape that then could automatically be read into the data reduction computer on the ground. (Telemetry was too unreliable in those days to risk on such an expensive program.) By 1954 our efforts were targeted toward meeting the needs for evaluation of the XMD-7 Fire Control System being built by Emerson Electric for the secret supersonic B-58 "Hustler" bomber. The B-58 was not yet ready, so the program was called "Pre-B-58," and tests were run with the B-47 in the beginning (MIT Servo Lab 1958). It was the most sophisticated test program of its time, and cost over $10,000 per flight hour, just for the aircraft, I believe.

By the spring of 1953, programming was complete for the autocorrelation functions and work had begun on a polynomial fit program (Ross 1954c) to place the ballistics tables in computable form. By the summer, an elaborate *mistake diagnosis routine* (MDR) (Ross 1953c) which allowed breakpoints to be set in an arbitrary program, with intermediate results saved for printout or displayed on the camera scope, was written and disclosed that the interpretive floating point arithmetic routines of the Whirlwind comprehensive system (CS) were inaccurate, so they were repaired, to everybody's benefit [WW4Q53-p25andp11]. The MDR played a continuing role in our experimental programming environment, especially when it later was placed under MIV control. In the summer of 1953 the first programming for the evaluation equations also got underway [WW3Q53p25], but most of my time in early 1954 was focussed on my master's thesis on minimizing Gibbs' phenomenon oscillations in the computation of Fourier transforms. But by the summer of 1954 the programming stage was set for our first serious use of manual intervention techniques in our work.

HARDWARE BACKGROUND

To complete the stage-setting for the Scope Input coup, I will give the following synopsis of Whirlwind hardware developments during the corresponding period. In the summer of 1952, when I started to program on Whirlwind, it had only 1024 words of *electrostatic storage* (ES) of 16 bits each, and operated at the then-blinding speed of 20,000 operations per second (hence its name) [WW3Q52p4]. In those days we

used both audio and visual aids in the debugging of programs. Originally, a scope in Test Control duplicated the *X-Y* deflections of the ES tubes' storage matrix so that the loop structure of programs could be seen, in action. The more often a loop body was executed, the brighter its trace. The audio aid consisted of a loudspeaker attached to a digit (I believe was digit 13 of the accumulator) so that every time its binary state changed from a 1 to a 0 or back, a beep was heard. The execution rate of program instructions was such that loops of different lengths made tweedly tones up and down the scale, so that you could hear the various phases of the program taking place.

After the shutdown and changeover to the new I/O system in August 1952, things remained stable for a time, but in the second quarter of 1953 ES was doubled to 2048 words by the addition of another bank of 17 tubes (one was a parity digit) [WW2Q53p30]. Then suddenly and smoothly, in the third quarter of 1953, both ES banks were permanently replaced by the first two banks of *magnetic core memory* (on-line August 8 and September 5, respectively) [WW3Q53p32-34]. The magnetic memory cycled at 9 microseconds versus 23 microseconds for ES, so Whirlwind now could do over 40,000 instructions per second [WW4Q54p4]. But still most loop structures remained in audio range, so the loudspeaker retained its usefulness.

The ES scope trace had been abandoned in the fourth quarter of 1952 when my first-ever program for computing the autocorrelation function couldn't be run without causing Whirlwind to break down. Other programs sometimes gave trouble, too, but mine showed the problem to be that its tightest calculating loop fell in a group of registers that happened to fall in one corner of the square matrix of storage locations. The spherical pattern of the ES tube's holding gun (which uniformly, but with inverse square law fall-off, sprayed the entire storage surface with electrons to compensate for leakage) couldn't be adjusted for the corners without upsetting the center of the matrix, when adjacent bits were accessed so rapidly. The engineers solved the problem by changing the diode pattern of the *X-Y* decoders so that consecutive words were randomly scattered around the square matrix, so that even tight loops were geographically dispersed [WW4Q52p6]. My program, with a short data tape, was added to the regular maintenance test suite used for ES adjustment and checkout, and the ES memory trace scope was no longer useful for debugging, since loops didn't show clearly.

In the meantime, however, many Whirlwind programs made use of the 16-inch display scopes, and Test Control had two of them mounted high up on a rack. The lower one had a 35mm camera under computer control, for making photographic records, and the upper one could have all 16 analog display lines ganged together to serve as a monitor check on whatever program was running. Thus the normal

operational milieu for the computer operators still had the full audio-visual complex of lights, scope, and loudspeaker. When a strange sound pattern occurred, the reflex action was to glance at the monitor scope, and then start to examine the indicator lights to gauge roughly by brightness where the program was operating in memory or was the accumulator stuck in some repeating pattern, etc. That's what the operators had done, when I got that loud, panic call over the intercom.

There were more than a dozen other scopes in the cavernous black-painted, shrouded room 222, eerily lit by dark red light (Fig. 1). All except the area discriminator scope were mounted at a slant, surrounded by banks of push buttons and indicator lights in several styles of interactive consoles for the various personnel needed to perform the many functions of airplane tracking and control of the air defense mission. The most elaborate was the *E31 console* of the combat data

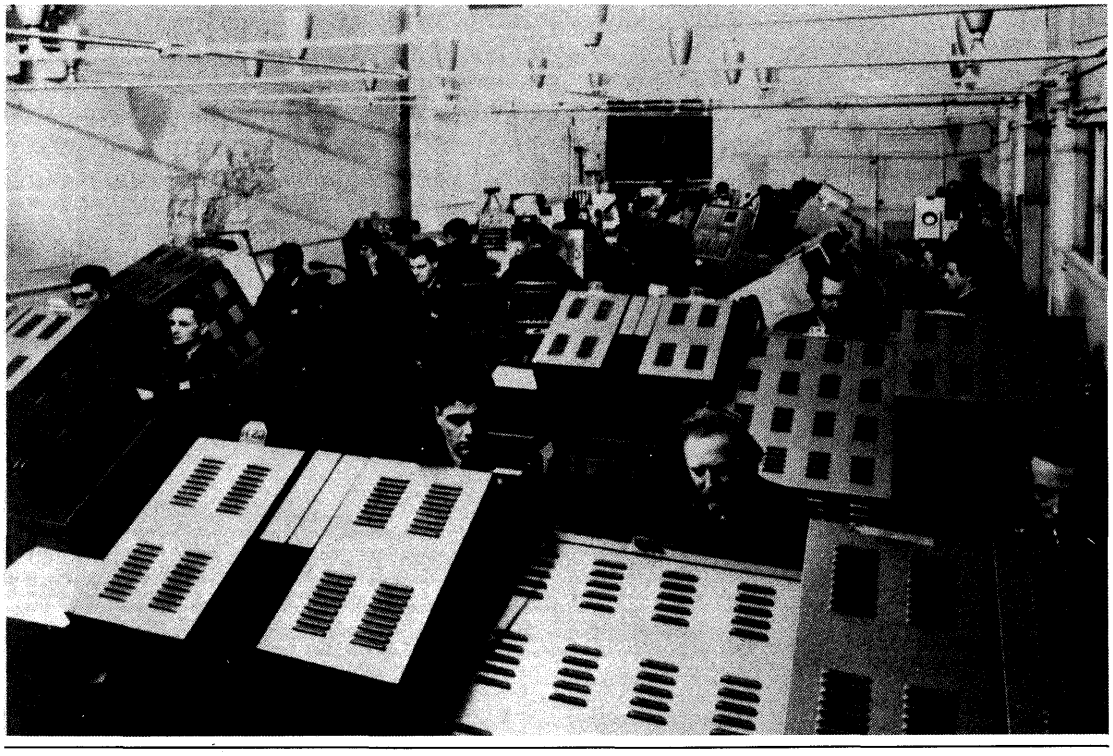


FIGURE 1
Room 222 of Cape Cod System. Closest two airmen with backs to camera are at E31 console (right) and adjacent console (left) used for gestalt programming. Directly beyond them is tripod and box of the area discriminator light cannon. Originally walls were black. Indirect ceiling lamps had red bulbs for system operation. (2/19/54)

director, which in addition to its elaborate set of buttons and lights, duplicated the essential master controls for Whirlwind, so that the entire complex could be controlled from that station, bypassing the normal operations of the master Test Control in the main computer room. But I, of course, had no knowledge of any of this, yet—it was all very hush hush, and you needed special security clearance and a need to know to even take a peek.

My records don't show when it happened, but sometime in 1953 or early 1954 arrangements finally were made with Lincoln Lab to allow John Ward and me to be introduced to the enticing room 222. I can still remember that first walk through the black-cloth-draped entryway into the red-lit, darkened room just *filled* with green-glowing scopes and orange-red neon indicator lights. Neat! Just what we could use to obtain mastery over the behemoth evaluation program, for whose complexity I already had a growing respect. The functioning of the various parts of the Cape Cod system were explained to us (although it was only later that we saw an actual test run with planes being tracked by the operators, etc.). It was clear that the E31 console (Fig. 2), with part of an adjacent console (Fig. 3), would be the best control station (PWS!) for general purposes, since the full complex of capabilities (including light gun and more buttons on an adjacent console) was available there along with the duplicated computer controls and the intercom.

The area discriminator almost got lost in the shuffle. I now will give a complete description of it. All the other scopes were mounted in consoles with switches, buttons, and light gun for manual intervention. The Whirlwind *light gun* (Fig. 4) was shaped like a backward pistol, with a sight close to your trigger finger knuckle and a wire coming out of a barrel that extended back over your hand. The barrel con-

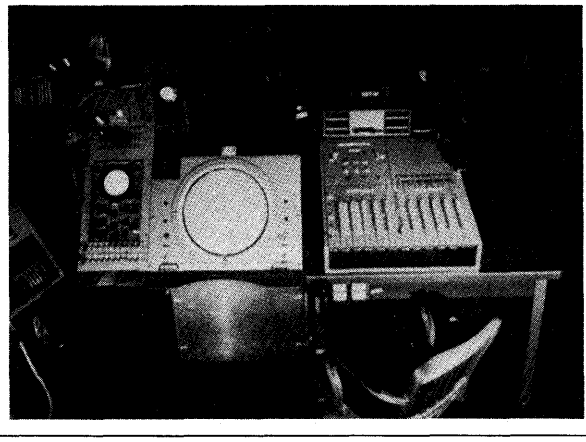


FIGURE 2
E31 console in room 222. Note intervention register octal keyboard banks and intervention switches, activate pushbuttons and lights, functionally arranged. Intercom and computer restart button on right.

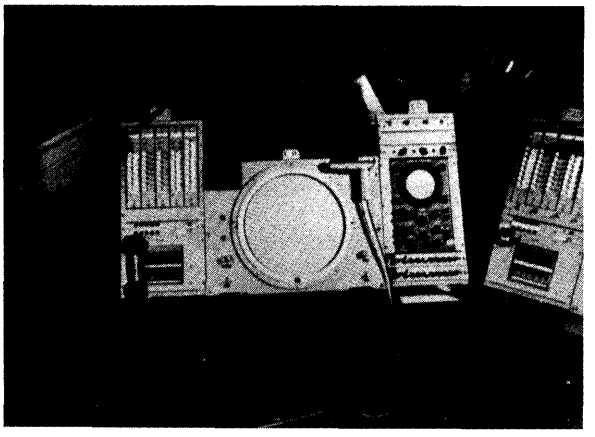


FIGURE 3
Cape Cod Console with light gun and manual intervention (MIV) equipment.

tained a photo-multipler tube, and the wire connected to an "si" line of the computer. If a displayed spot was in the sight when the trigger was pulled, that would set an *activate bit* (one-time, read-only) so that a suitable "si, rd" sequence following the "si, rc" that displayed the spot (but before any other spot was displayed) said that the operator had selected that spot from all the others in the display. This was used to assign tracking functions to radar blips which were returns from airplanes.

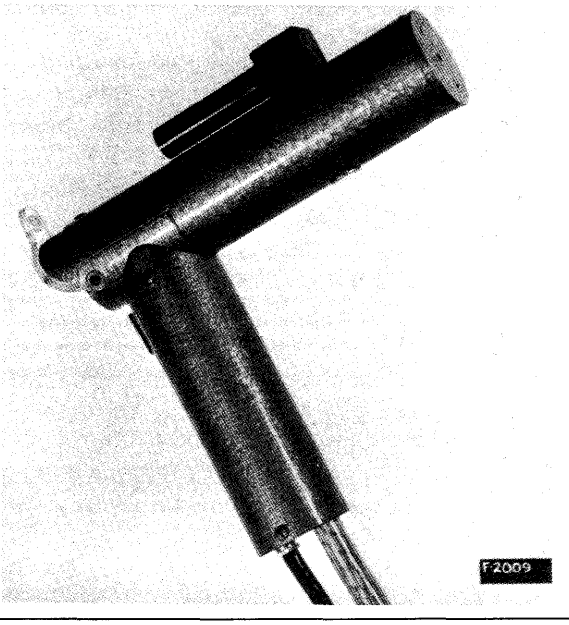


FIGURE 4
Final design of Cape Cod light gun, with sight (10/2/53).

The Area Discriminator had no buttons, lights, or anything for human intervention. It simply sat in a large box in the back corner of the room. Mounted on the box was a tripod that supported a smaller box with no bottom, so that it was suspended about 18 inches over the horizontal scope face surface. This was the light *cannon*—a photomultiplier tube mounted so that it viewed the whole scope face and could be "si, rd" read for any spot in the display. When we saw it in the Cape Cod system use, a large circular piece of dark yellow plexiglass had been laid down, centered over the scope face so that only a narrow annular ring around the edge of the scope was open to the light cannon's view. The yellow filtered out of the blue phosphor flash of any displayed points, so the cannon was blind to all central display activity, but any fresh radar tracks crossing the perimeter could be seen and could be brought to the attention of the track initiator/monitor operator for light gun tracking. This is why it was called "the area discriminator," and with the plexiglass disk in place, it actually was an analog computing element component of the Cape Cod system.

SOFTWARE BACKGROUND

My guess is that our tour of room 222 probably was sometime in the spring of 1954, for the three 16-inch scopes mentioned in the fourth quarter 1952 Whirlwind Quarterly Report (at the time of the I/O shutdown) had grown to more than twenty, and I think all the Cape Cod consoles were complete at the time. In any case, I had my thesis to finish, Dick Turyn who had been programming on the evaluation program was leaving so I needed a replacement, and we needed to learn the ins and outs of room 222 programming. Therefore we hired Bill Wolf from Lincoln Lab, solving several problems at once. He came in June 1954, but stayed only about six months, (MIT Servo Lab 1952–1958) when he left to go into consulting for himself. Later he formed Wolf Research and Development Corp., which acquired Whirlwind itself under Navy sponsorship, when Lincoln and MIT had no further use for it (Redmond and Smith 1980, p. 224). Wolf R&D also had the first facilities contract to run NASA Houston computers, I believe.

I guess I'd have to say that Bill was a better businessman than programmer. He did a fine job for me, writing a basic set of programs to use all the functions of the E31 console and a program to display large, fat letters on the scope, big enough to be directly read from the 35 mm film so that we could frame our classified results with "CONFIDENTIAL—SERVO LAB—CONFIDENTIAL" before and after our plots and displays (Wolf 1954). But the programming style was very tangled and hard to follow (and I suppose not atypical of the code of other Lincoln people who were learning to master the tricks of this new manual intervention trade, for that was a major reason I hired

him). Bill's code was far from debugged when he got the wanderlust. I hadn't been working closely enough with him to know what I was getting into, but I said I'd finish the debugging because that would give me the education I'd wanted anyhow. So off he went to make his fortune.

Almost immediately I saw that I would have to completely rewrite and replace Bill's work, and proceeded to do so. Many features of modularity, flexibility, robustness, and generality were needed in order to match my growing vision of what general MIV (i.e., the PW functionality of PWS) was to be all about. Without Bill's efforts to critique, many of these aspects (which I certainly had not articulated beforehand as specifications) would not have occurred to me until much later. Also some basic principles of MIV programming (that I have seen violated repeatedly in the 1970s and 1980s as the microcomputer age programmers have re-encountered and reinvented those 30-year old phenomena) showed up in my critique of Bill's code. For example, he had the "si, rd" to read a switch directly at the point-of-use in the program flow, throughout. I structured separate read cycles and stored the switch settings in program variables. Not only did this make the overall program better structured, but it also meant that anything that could be done manually also could be done automatically, because actual program control came from the stored variables, not from the switches themselves. This feature made possible our log and playback of manual actions as well as test simulations.

DIRECTOR TAPES

It was while I was first reading, with some understanding, about the complete set of "si" capabilities of Whirlwind (including the Lincoln Lab-only portions, previously unknown to me) that I discovered that the mechanical tape reader of the Test Control Flexowriter was still operable, even though there always had been a photoelectric tape reader (PETR) since I had been around. Although the print mechanism of the Flexowriter had been in daily use all along, for operator communication, and the direct tape punch recorded some logging information, the tape reader had no current use. The number of separate tapes and the complexity of operator instructions for our evaluation program runs was getting so bad that I suggested to John Frankovich and Frank Helwig (on the Digital Computer Lab staff) that the Whirlwind utilities should be expanded to put all these instructions on a tape that the mechanical reader could read, while the main tapes were read in by the much faster PETR. It was worth the wait for that idea to get worked into their schedule [WW3Q54p7], for the *Director Tape* program, as it was called, included not only the terse control language basics, but also conventions for labelling punched tape headers as to type (binary,

program, post mortem request, etc.). When director tapes were further upgraded in the summer of 1956, they still were not used by many programmers (the tape room didn't like to splice all the tapes together) [WW3Q56p23]. But they were essential to our room 222 runs, for they allowed us to completely eliminate the operators. I believe the director tape capability was the first true operating system in the modern sense of the word, especially when combined with our beginning versions of group control (based on the earlier MDR needs) [WW4Q54p15] for manipulating drum storage. Director tapes and group control are parts of the systematized solution third component of my PWS theme.

THE SCOPE INPUT PROGRAM

My records don't show exactly, but I do remember dreaming up the scope input program on a long flight back to Boston, and my records show that I did attend an early Pre-B-58 meeting with John Brean (the lead engineer on the project's instrumentation work) in Dallas, on August 9, 1954, so I'd guess it was about that time (Convair 1954 and [C54817 to S.C. Marcus, Emerson Electric]). Bill Wolf still was trying to really get his programming under way, but various test runs were being made by both him and me in room 222. So I knew just enough to write the program. Perhaps triggered by the Dallas discussion about all the things that could go wrong with getting flight data from the plane into the computer via a crude tape system, I envisioned patching up bad spots in a plot of the input data, free hand. We already had had enough experience with the polynomial fit and a Lagrange interpolation programs that had been written in preparation for the evaluation program, that I didn't trust any analytic methods. As the earliest written reference I have found so far (Fourth Quarter 1954) says:

> A "scope input" routine has been written which gives a type of two-dimensional analog input to the Whirlwind computer. The principle of the program is similar to the "flying spot scanners" used on analog computers. The equipment used is a 16-inch oscilloscope under the control of the computer with a photocell mounted so that its field of view is the entire scope face. Then by programming a flying spot and asking whether or not the photocell "saw" the spot, the program can be made to follow an opaque pointer as it is moved in a random fashion over the face of the scope. Since the program displays the spot by digital coordinates, the tracking of the pointer constitutes analog input to the computer. [WW4Q54p15]

The scope input program is quite simple and direct. The tracking function itself consists of a tight loop that displays a 5 × 5 square array of close-spaced dots, left to right, top to bottom, interrogating the light cannon after the display of each dot (see Fig. 5). If a dot is *not* seen, the array is recentered on that location and the cycle is restarted from

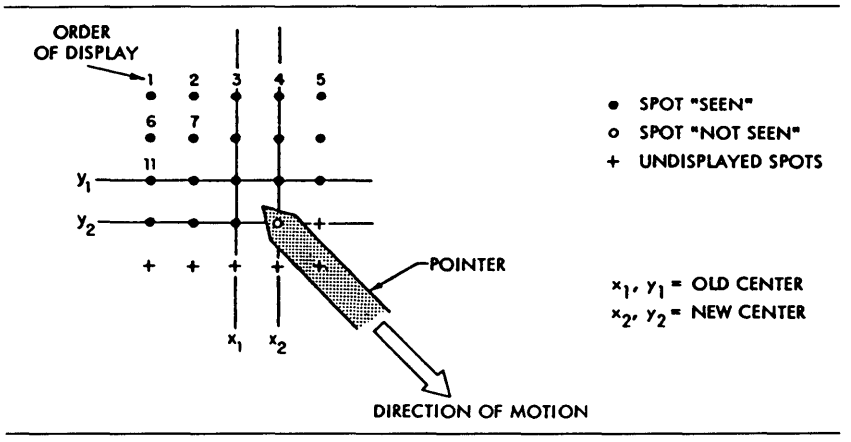


FIGURE 5
Tracking principle of first scope input program August, 1954. Shadow was tracked using area discriminator light cannon.

the top left corner again. Otherwise the next dot in sequence is displayed and tested. The cycle is so fast that the array can track even quite fast motions of an opaque pointer—catching up to a new position before it can get very far.

Modified parameters of the same logic allow the pointer position to be located initially. A wide-spaced 32 × 32 array covering the entire scope face zeros in by successive half-spacing until the tracking array can take over. Since the pointer is hand held, at least the hand will be caught by the initial spacing. This two-mode program was the full extent of the initial Scope Input program, for the intent was merely to demonstrate the tracking principle.

On that fateful afternoon, I set up the scope input tape in the PETR, bid the operators adieu without saying anything special, and went off to room 222 by myself. As I approached the E31 console in the middle of the room, about to lay down my papers on its narrow desk I gasped. The restart button was there on the console, but the area discriminator was 12 feet away in the corner! Would I have to call for operator help after all? Suddenly it dawned on me! The top-to-bottom, left-to-right logic of the scan meant that my tracking cycle actually was a maximum-seeker-from-the-left! In other words, whatever shape made the shadow, my spot would rush up its left edge as fast as possible—and then (by the logic) would get stuck at the very top, because it couldn't track downhill! I saw that that was *why* I expected it to track at all.

With that insight, my problem was solved. All of our work was done on pads of yellow paper (opaque to the Ozalid copying process). I merely tore off a blank sheet as I went over to the area discriminator to remove the plexiglass disk. In its place, I laid down my sheet of paper, roughly at a 45 degree angle. I went back to the E31 console

and pressed the restart button. The lights indicated that Whirlwind was running. Was my program in a loop? You bet it was. As I approached the area discriminator I could see something was happening. There on the top corner of that piece of paper was a bright spot. The loop was my display loop. So far so good.

Very cautiously I snuck the index finger of my right hand up the yellow page, staying in its shadow. Then very gently I used my left hand to slip the paper out from underneath—and sure enough, the spot remained—but now stuck on my finger. I started to write. *That's* what got the operator's attention in the control room. The almost-random tracking recycling made a rasping sound on the loudspeaker, and that's what first got their attention. The rest is history.

I haven't found the original scope input program in my papers, but I do have the version used in the symposia the following year (Ross 1954d). (I now realize that I could have simplified the program still further, by using the left-edge-following feature to switch into tracking immediately from the first start-up dot, not seen. Then both the half-spacing logic and the initialization for full 32 × 32 spacing could have been eliminated.) Later I had our model shop make an automatic remote button-pusher, since I wasn't allowed to tamper with any of the Whirlwind wiring. A hand-held pushbutton on a long cord activated a solenoid mounted on a bracket held in place by a loosened framing screw of the E31 console frame, so that the restart button could be pushed remotely from the area discriminator area. A wheeled dolly held a power supply for the solenoid, and plugged into a 110 volt outlet. It worked, and our delicate and unique relationship with Lincoln Lab regarding use of the equipment was undisturbed. All I had to negotiate was a place to park the dolly between runs!

We never got around to actually *using* the scope input program for anything, however. I didn't need it for patching our Whirlwind test data for the evaluation program, and for some unknown reason we didn't specify a light cannon for the Eglin Field 1103 Charactron console. We did make a light cannon for the IBM 704 and 709 computers at MIT, but again it only was used for a study of various light pen and other tracking studies in 1960, for the Computer-Aided Design Project (Ross and Ward 1961). I never was successful in interesting anyone to make a generalized shape reader, which could easily be constructed just by changing the sequencing of the 5 × 5 tracking array (to seek minimum from the right, etc.). So in its historical setting, my scope input program was merely an interesting demonstration piece that the crowds loved, but was a bit before its time.

Practice

The pieces all now were in place. I could actually practice the new MIV techniques in conjunction with the evaluation program. I also could

practice talking about what was new and exciting about this intimate coupling of man and machine.

THE FIRST SYMPOSIUM (MARCH 1955)

In any project, a point is reached when it is necessary and appropriate to demonstrate some initial results to the outside world. This point was reached for the Fire Control System Evaluation Project in the spring of 1955. Even though my part of the project, on data reduction, could have used a bit more time to digest the integration of my reworking of the MIV programming with the parts of the evaluation program, a symposium on the *Design and Evaluation of Bomber Fire-Control Systems* was scheduled for March 8 and 9, 1955 (MIT Servo Lab 1955). The first day covered various engineering matters, mostly concerned with the servo dynamics of a new hydraulic antenna drive, and ended with demonstration of the drive, and (for added interest) the numerically controlled milling machine in Servo Lab. The second day covered evaluation. John Ward presented our philosophy of evaluation, John Brean the concept of digital instrumentation, and I presented the air-mass-based evaluation analysis and MIV-based data reduction programming. In the afternoon, buses shuttled two groups of about 75 people between Servo Lab and Barta Building for digital instrumentation and room 222 demonstrations, respectively. The visitors list shows over 170 names from 48 government, industry, and MIT organizations. It was an impressive show.

Other than an agenda and the visitor's list, there seems to have been no handouts for the symposium. I have a draft copy for some glass slides, an outline of my talk, and a cardboard cutout of a fighter aircraft profile which was placed on the area discriminator scope, so the light cannon could talley hits on the target according to the calculated projectile position. (We used actual test data from Project Hornet, an earlier test program carried out by Emerson Electric for the B-52.)

The complexity of the Data Reduction show is indicated by my detailed instructions (dated 3/8/55 for the demo the next day!) for preparing the ROSS DEMO Director Tape (see Fig. 6) (Ross 1955a). Although I no longer can decipher the director tape language, there are well over 100 operations encoded—loading tape segments onto the drum, executing initialization sections, calling up individual programs and executing them—all of which would have had to be carried out flawlessly by the operator, without the director tape feature. (As a matter of fact, I had pre-loaded my paper tapes onto one of Whirlwind's magnetic tape units, for director-tape-controlled loading. The first time it worked fine, but for the second group of people the magnetic tape unit broke down! I handed the microphone to John Ward to fill the gap as best as he could, ran downstairs to the tape room files, rushed back with an armful of disjointed tapes in boxes, and with the help of

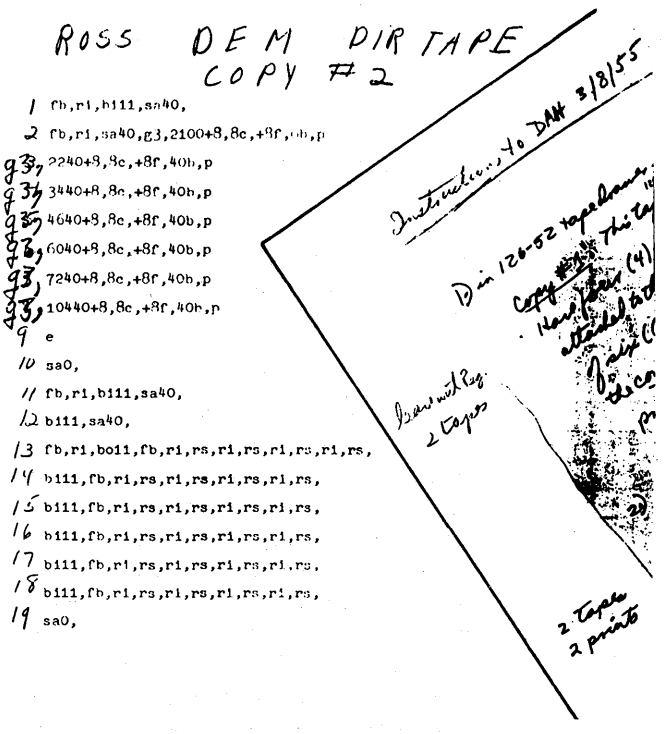


FIGURE 6
Printout of ROSS DEMO Director Tape for March 1955 Symposium. Eliminated computer operator actions and presaged operating systems of today.

the operator [probably Joe Thompson, who was the best!] managed to complete the second demo, breathlessly but in full. So we did it both ways!)

Every scope in every console in room 222 duplicated the displays I was demonstrating at the E31 console, so everybody had a good view. On request, pertinent variables could be monitored in numerical form; the geometry of the encounter could be shown (including firing pattern, measured through the airplane cutout for the light cannon), and elaborate, calibrated and labelled plots of all functions could be selected. I also showed the scope input capability, as well. The actual demo lasted about half an hour.

Another interesting tidbit about the demonstration is the following quote from the First Quarter 1955 Whirlwind Progress Report: "With the present logic the program demonstrated at the symposium had only five registers unused out of a total of about 3,500 actual program instructions. The new logic will be limited only by the drum capacity of the computer and will be more efficient as well" [WW1Q55p56]. This documents what I remember as a previous panic point in the preparations. The final assignment of group control drum

and core addresses came out to be almost 40 registers too big! Only by changing the words labelling the scope plots to abbreviations (''AZ'' for ''AZIMUTH'', etc.) was I able to achieve that tight fit in the nick of time!

THE MTC CHARACTRON DEMO

On April 14–15, 1955, I attended an *Instrumentation Symposium* at Patrick Air Force Base, Cocoa (Cape Canaveral), Florida, put on by the American Ordinance Association (AOA 1955). The most memorable thing about that meeting was the unauthorized detour the on-base shuttle bus took to drive past a large, flat, open area with some raw concrete in the ground, with reinforcing rods jutting skyward—all with no comment. It was the then-very-secret first launching pad for the Atlas missile, just being built.

The response to the MIV-controlled data reduction in our own March symposium had been so great that we were requested to put on another show on just that topic. On May 2, 1955, John Ward issued an invitation to a *Symposium on Data Reduction* to be held at MIT, June 1 and 2, 1955 (Ward 1955). Inspired by my Cape Canaveral trip, and to provide, in addition to the Room 222 demonstrations, a demonstration of the Charactron display tube capabilities, I quickly learned how to program the *memory test computer* (MTC) (Fig. 7) at Lincoln Lab (Redmond and Smith 1980, p. 206). This was a general purpose vacuum

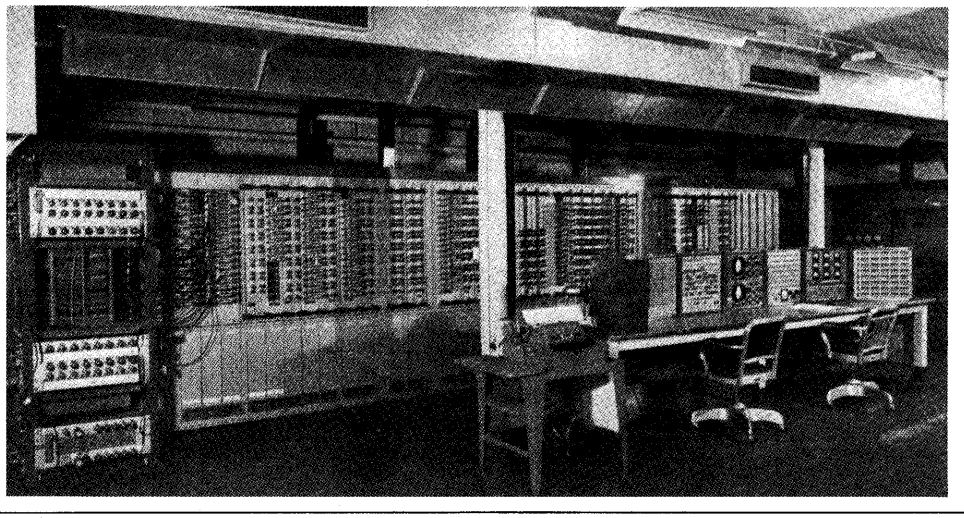


FIGURE 7
Memory Test Computer (MTC) with magnetic core stack (4/6/53). Used for Charactron demo in March and June 1955 Symposia on MIV.

tube computer that had been built specially to test the original Forrester magnetic core memory stacks. When the tests were successful and the core stacks were suddenly moved to Whirlwind, another stack was made for MTC, and it was then being used to shake down the Charactron display for use in the SAGE system.

I still have my MTC working papers for my *Missile Launch Simulator* program, along with the original CS Tape Preparation Requisition for "Tape O," submitted at "4:25 PM, May 24" (Ross 1954a). In addition to showing examples of how the MIV monitoring displays of intermediate results would look in Charactron character display format (rather than Whirlwind's elements of a 7-stick figure eight character display [WW2Q53p31]), the program used the vector plotting capabilities of the Charactron to draw a map of the Florida coast plus some islands, and then cycled through a moving display of a missile launch, showing where the pieces would fall if the test director had to abort the mission and blow up the missile in flight. The islanders would have appreciated such thoughtfulness, I reasoned.

THE SECOND SYMPOSIUM (JUNE 1955)

Three days later (and only one week before the symposium—things moved fast, in those days), John Ward invited Pat Youtz (who had led all the Whirlwind work on the ES memory tubes, and now was in charge of SAGE display work) to "describe to the people at our data reduction symposium just what can be done in the near future in the way of input-output." His letter goes on:

> As you may know we are doing data reduction studies for the Air Force and have developed techniques on Whirlwind using the scope and intervention that are a great improvement over any other techniques in use at present. We are building digital instrumentation for a forthcoming field test which will be evaluated on the ERA 1103 at Eglin Field, Florida, and the primary purpose of this meeting is to get the Eglin Field people and the ERA people together and show them what we have accomplished with Bob Weiser's attachments to Whirlwind, with the hope that they will get inspired to obtain similar equipment. We would therefore like to put on as good a show as possible and that is why we would like to show the charactron tubes in operation. We have permission to use the MTC and Doug Ross has written a special display program to put the charactron through its paces.
>
> It seems to us that if any equipment design is undertaken by ERA that the charactron should be used as the basis for design, and we would greatly appreciate any help you or your group could give us in showing our visitors what the charactron looks like and perhaps your manufacturing and test operation. We have taken the liberty of tentatively scheduling this material on Thursday, June second, as an adjunct to the MTC demonstration at 2:00 p.m. I am sending this to you

> so that you can think about the business a little and I will call you Tuesday to see if we can make some final arrangements. We would be most grateful for any assistance you could give us in convincing our visitors of the current feasibility of charactron type displays. I am enclosing a brief list of the organizations who will be present at the symposium. [C55527 JEW to Youtz]

Pat did indeed pitch in, and both the Charactron and the Memotron (a smaller storage tube version) were included in the June 2, Lincoln Lab visit.

On July 29, 1955, John Ward issued "Minutes of M.I.T. Data Reduction Symposium, June 1 and 2, 1955," Report No. 7138-S-1 (Ward and Ross 1955), which includes my 7138-M-112 memorandum "Special In-Out Equipment for Human Participation in High-Speed Computer Operation," July 25, 1955 (Ross 1955c), in which I summarized the "Program Operation" and gave a "Description of Equipment" for activate buttons, intervention switches, indicator lights, and audible alarms, each with a diagram, description of operation, and "Things to Notice" about how they would be used with the EF (external function) instruction of the ERA 1103.

For the record, John Ward's letter to Youtz shows that the following organizations were invited to the symposium, and the minutes show how many people actually attended:

Military Physics Research Laboratory, University of Texas (2)
Emerson Electric Company, St. Louis (3)
Air Force Armament Center, Eglin Field, Florida (4)
Armament Laboratory, Wright Air Development Center (4)
Engineering Research Associates, Remington Rand, Inc., St. Paul (2)
Patrick Air Force Base, Air Force Missile Test Center, FL (3)
Holloman Air Force Base, Alamogorda, New Mexico (0)
Air Force Flight Test Center, Edwards AFB, California (1)
NOTS, Inyokern (0)
Bell Telephone Labs (1)
Convair (3)
Davies (0) [manufacturer of the airborne mag tape unit]
National Security Agency (2)
Watertown Arsenal (0)
White Sands Proving Ground (1)
WCRRU, Aero. Res. Lab. WADC (2)
M.I.T., Instrumentation Lab (2)
M.I.T., Air Force Field Rep (1)
M.I.T., Digital Computer Lab (3)
M.I.T., Lincoln Lab (2)
M.I.T., Servo Lab (12)
M.I.T., Statistical Services (1)

We never had further contact with the Patrick Air Force Base people, but some years later the Sputnik era and NASA TV coverage showed

elaborate Mission Control facilities in full bloom. I like to think our earlier efforts at least had some influence on those crucial developments.

Gestalt Programming

The response to the two MIT symposia and the clarification in my own mind that what we were building was indeed a new way to use computers prompted me to make this the topic of what would be my first professional paper. Even though all of Whirlwind may seem too huge to be a PWS, my view was and is that the PW aspect dominated the WS reality then available, so the theme of that paper and this paper is valid.

EARLY VIEWS

My records don't indicate when I decided to submit a paper for the 1956 Western Joint Computer Conference (precursor to the Spring and Fall, and now the National Computer Conference series). I was chairman of the printing committee for the November 1955 Eastern Joint Computer Conference (EJCC 1955), held in Boston (we designed the conference logo, combining the logos of the IRE, AIEE, and ACM joint sponsors, which was used for several years thereafter) so I knew how the conferences ran. The earliest dated item I have (the date stamp is October 13, 1955 [Ross 1955d]) is a handwritten, yellow-page draft reading "*[The] Gestalt [System of] Programming*—A New Concept in Automatic Programming," with the bracketed words crossed out, along with an aborted opening sentence. Notice how that subtle and impulsive act makes the title say that this is a paper about a generic concept of a *type* of programming. If the words were not crossed out, the paper could be about a systematic *way to* program, a school of style, or (even more specifically) the paper could be about my particular collection of programs constituting a system with that *name.* In the first two drafts I *do* use the name "Gestalt System I" in a couple of places. But when I see the cross-outs now, I see it as evidence (matching the slant of the words of all drafts of the text) that I always had the generic concept in mind as the subject of the paper.

I remember that from the first rush of insight that so entranced me in that first exciting visit to room 222 I knew that things would be profoundly different when man and machine were intimately coupled. I always had a broader view than just our data reduction application in mind. The earliest record I find on our use of MIV is in the Third Quarter 1954 Whirlwind report, which is suitably subdued, but also clearly is targeted to experimental programming:

> Work is progressing on another phase of this problem, using auxiliary in-out equipment both as an integral part of the data-reduction programs for monitoring purposes and as an aid to experimenting with improved computational techniques. Intervention registers and scopes will constitute the working parts of this system with efforts being made to provide the flexibility normally associated with analog computers. Present work is concerned with the development of routines to decode intervention-register inputs and to utilize various specialized scope outputs. After merging these routines with the data-reduction programs, an elaborate logging scheme, using magnetic-tape output, will be written for permanent records of all operations and interventions during an experimental run. [WW3Q54p14]

But in an August 11, 1955 letter to my Oberlin classmate, Ben Scheff, urging him to come and work with me at MIT following his navy tour at the National Security Agency, more of my real feelings show:

> The work on fire-control system evaluation has been extremely interesting and has grown continually in scope and capabilities. Now we are really on the forefront of a whole new way of using large digital computers. We are in the fortunate position of being purely a research group and never have to get our hands dirty with actual data reduction. Once we get a new technique developed and proved, we drop it or expand it to new, more comprehensive work.
>
> A large part of our efforts go [sic] into developing new techniques and programs to achieve more flexibility in devision [sic] and testing new programs. The MDR report, which is enclosed, is one such program, and is a pretty good example of the kind of programming and philosophy we like to follow. The other special report on our Symposium gives some idea of the techniques we're now exploring. Both reports are now out of date but give the ideas.
>
> It is really amazing the complexity which arises in making the computer act in these ways, but will be a really nice system when it's done. We have an intermediate version running now which we demonstrated at the Symposium, but the newest version (almost complete) will run rings around it. The new system is really much more general than we now require but it will essentially allow us to program with programs while the computer is running instead of coding for weeks before a run.
>
> In a nutshell, what we're doing is fixing up Whirlwind so that we can manipulate its operation while it's running, turning on and off programs, examining and modifying so that we can use the computer as a research tool on an idea or concept level, rather than on the grungy coding level. Frankly, it's pretty exciting work . . . I can honestly say that I don't know of any other work in computers and their use which is more interesting than what's going on here. [C55811]

INITIAL DRAFTS

I have several distinct drafts of my attempts to write the gestalt paper. The November 17 version actually submitted to Byron Bennett (SRI), WJCC Program Chairman [C551117] is completely different from the October 13 draft, and that paper, in turn, was completely reworked several more times before the final submission of our Project Technical Memorandum version (Ross 1956c) on February 1, 1956, for publication in the Proceedings. I think it is historically important to dwell briefly on the evolution of my expression of my then year-old, but still elusive ideas, because such an analysis will show my present PW versus WS theme in the struggles of its original context.

The original (October 13) draft jumps right in with the dictionary definition of gestalt psychology: "The theory that . . . events do not occur through the summation of separate elements, . . . but through formed patterns of these, integrated units which function singly or in interrelation," and proceeds to say that standard programming practice is to be replaced "by simple and unambiguous expression of the desired characteristics of behavior, the expression itself in effect tying together integrated units of computer behavior . . . " (presaging today's declarative languages, functional programming, and the reusable components of software engineering). And still in the opening paragraph, "automatic coding systems are aimed at easy *communication* between the human and the computer [see Figs. 8 and 9], whereas a Gestalt system of automatic programming is aimed at easy *conversation* [see Fig. 10] . . . " (a phrasing with which I was particularly pleased, and which I retained in all further drafts) (Ross 1955d). These are indeed the pertinent ideas, but the later drafts are much more roundabout in approach. At the time, even ordinary programming was still immature, and there was a complete lack of the common, shared cul-

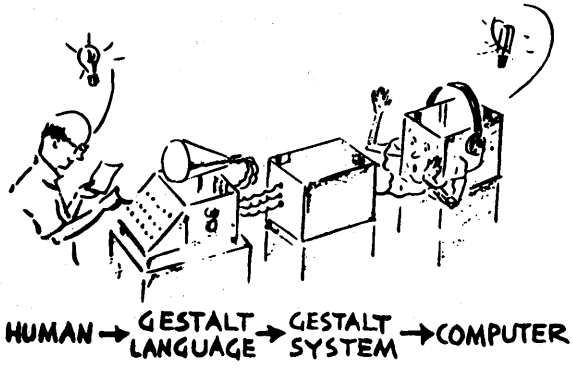


FIGURE 8
Communication from human to computer.

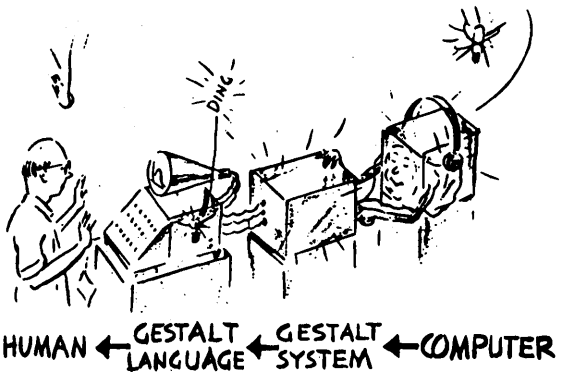


FIGURE 9
Communication from computer to human.

ture we know today. The ideas were very hard to formulate and very hard to assimilate.

The second (November 17) draft, the basis for acceptance of the paper, begins with the following introduction:

> In any human endeavor there are three major phases: conception, expression, and execution. Gestalt programming is an attempt to make these three phases as nearly identical to each other as possible with respect to computer programming. In this paper the word *Gestalt* is used to mean *a concept of a task* to be performed by a computer. In a Gestalt system of programming the Gestalt, or idea, is expressed simply and un-

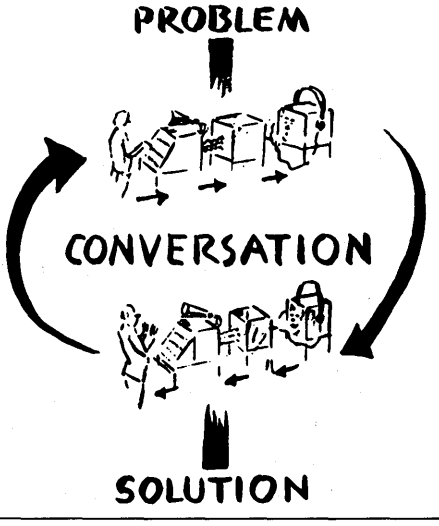


FIGURE 10
Solution by conversation.

> ambiguously in a special language, rather than through the laborious assembling of machine-codes, pseudo-codes, subroutines, etc. Using a Gestalt system, the expression itself in effect ties together integrated units of computer behavior, which function singly or in interrelation, to achieve the desired effect. The purpose of a Gestalt System is to facilitate the transmission of general ideas as in a conversation, between a human and a computer, so that the maximum use of their respective capabilities can be made.
>
> After presenting the abstract theory of Gestalt Programming this paper discusses several Gestalt Systems in use today at the Massachusetts Institute of Technology and describes briefly the types of computer hardware which are best suited to this application. (Ross 1955f)

This was retained as the *synopsis* for the final paper, but the body was reworked many times. The referenced MIT systems were the Whirlwind comprehensive system (CS) itself (including mention of director tapes), the Whirlwind marginal checking facility (which required a technician to classify oscilloscope traces, etc., through MIV, and then isolated faulty tubes and components), and our own Data Reduction and Experimental Programming System.

In addition to various attempts to describe problems in which MIV was desirable (not an easy task in the hands-off, standard mode of submitting runs to a computer operator) I needed to have an example to illustrate the actual design and programming steps. My draft pages show one attempt using the LIP and RIP (left insertion panel and right insertion panel) and various buttons for CS computer operations, but that was not used. Another complete development says "Consider the launching of the man-made satellite schedule for the coming Geophysical Year," including the suggestion that a "part of the Gestalt Language might be a kind of joy-stick, so that the human could steer the rocket remotely" (Ross 1955e). But I decided that this example would seem too flashy, and instead worked out an example of automatic factory control (Fig. 11) for the final paper (perhaps a prescient choice, since my soon-to-arrive role in APT was unknown to me then).

ORAL PRESENTATION (FEBRUARY 1956)

Then there follow endless yellow pages of my attempts to work out an oral presentation of the paper (Ross 1956a) ("The title of this paper has probably raised questions in your minds both about what I am going to say and why I have chosen to introduce a new word-usage into the already crowded computer terminology"—scrubbed, fortunately!). I presented an interdepartmental colloquium version at MIT entitled "Talking with Whirlwind—An Introduction to Gestalt Programming" on January 11, 1956 (rescheduled from December 14) [C56016 to Bennett, and (Ross 1955f Addendum)], and finally managed to do a decent

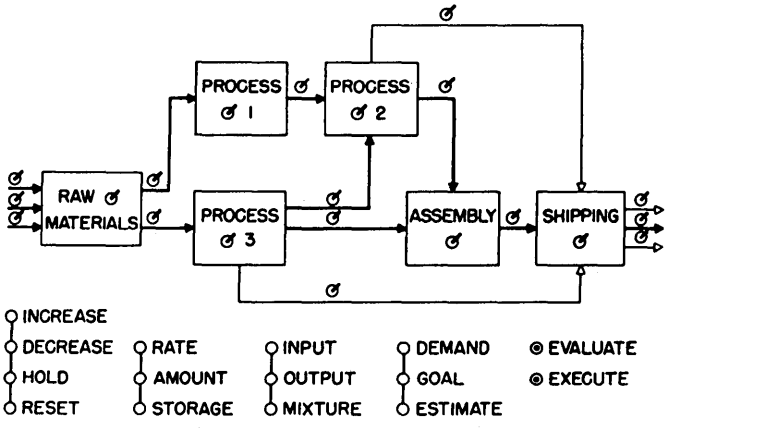


FIGURE 11
Gestalt language for automatic factory control.

job as the opening paper at the San Francisco WJCC on February 7. *Electronic Design* magazine enthused:

> Programming and coding sessions at the Western Joint Computer Conference, San Francisco, February 7, 1956, were attended by overflow crowds. Highly significant papers were read. Here is a verbatim report of the impressions on [sic] the session chairman, Francis V. Wagner, group leader, engineering computing, North American Aviation, Inc.
>
> The paper on "Gestalt Programming" by Ross, which opened the session, was a milestone of forward thinking. It refused to accept the common fallacy that human beings can contribute little when teamed directly with automatic digital computers, and took the position that only the more versatile human mind can, at present, handle some parts of some problems. In the past, many people have noted that, to exploit such a partnership, there is required not merely a satisfactory language for communication from the human to the computer but a language which allows fluent conversation between the human and the computer. Mr. Ross went beyond merely stating this, [I thought it was original with me, at the time!—D.T.R.] and presented evidence of some serious thinking towards analyzing how such a language ought to be designed. The principles which he stated and the illustrations which he gave of ways to design such a language, and of the hardware which might form the medium for the conversation should go far towards clarifying the problem for future implementation of these ideas. . . .
>
> Audience participation was distinguished more by quality than quantity. No more than three or four questions were asked of each speaker. [Wagner's overall impression of audience was that on the average they probably were not prepared to comprehend the concepts presented.—E.D. Editor] Although these were put by highly competent people, and indicated careful evaluation of the paper, they were aimed more at underlining or amplifying certain points. In no case did

> they take violent issue with any of the statements or implications of the speaker, nor did they open up any controversial issues. (*Electronic Design* 1956)

Even the *Scientific American* expressed an interest in the paper. A suitable summary of my thinking at the time is the following quote from my January 12, 1956 reply to Dennis Flannagan, editor, enclosing a copy of the paper. Actually I sent the final, TM version, but I still was concerned about its adequacy in expressing my ideas. This probably was my own dissatisfaction with my MIT talk the day before, even though it was very well received. A few Lincoln people let me know by body-english that they thought I was plagiarizing other people's ideas, however. It was clear that they thought the ideas were important—important enough to let me know their feelings—but I, in turn, saw their upset as evidence that they really didn't see the differences between the specific Cape Cod incorporation of the man in the loop as an engineering necessity and my generic gestalt programming idea. Somehow I wasn't expressing myself clearly enough, or they wouldn't feel that way, I thought. Hence I wrote to Flannagan:

> Since it is not stated too clearly in the present draft, let me try to summarize for you here, what Gestalt Programming is and where it fits into the established computer technology. As computer techniques have developed over the last few years, there has been a growing trend toward more sophisticated methods for connecting the human, who states the problem, to the computer, which is to solve the problem. Out of this trend has come, as a natural consequence of the maturing technology, a desire to use computers for solving problems which cannot be completely specified in terms which the computer can handle. These major trends are united into the general problem of using humans and computers together to solve problems. An analysis of this problem shows that more efficient and natural languages are required to bridge between human and computer, and that these languages must operate at the idea or concept level. In particular, a statement in the language must lead directly to the solution, in the same way that a Gestalt (in the psychological sense) unites at once, basic units into a single entity or pattern. The theory of Gestalt Programming is an attempt to set the initial outlines of the structure of these languages, and to indicate how to construct and use them. The physical content of Gestalt Programming is not new, but the emphasis and point of view is new, and is intended to clarify the directions in which these fascinating developments can go. [C560112]

But nothing further came from the *Scientific American* interest, which is too bad, because unbeknownst to me, at about the same time (and completely independently) George Price was in communication with the editors of *Fortune* magazine, preparing his landmark article on *The Design Machine,* which appeared in the November 1956 issue. Francis Bello, Science editor for *Fortune* wrote to me on March 27, 1957 that

> Price wrote us in November, 1955, suggesting his article and we had a rough draft in our office on January 15, 1956, which outlined the major features of his Design Machine. I mention this because it may surprise you that the article was so long in preparation before appearing in Fortune. (And certainly you would be entitled to wonder if Price had heard of your February paper.)
>
> What surprises me still more is that computer experts at I.B.M., Sperry Rand, and Lockheed, who read Price's paper for us, and checked his computations, had not heard of—or at least gave no indication that they had heard of—your work in this area.
>
> From the beginning our concern was that Price's ideas were too blue skyish for serious presentation. The world keeps moving faster than Fortune can keep up. [C57327]

To which the only comment (valid today, as it was then) is—it does indeed. Before I address Price's article, there are more items from our own work to be covered.

Console Developments

With the PW component established, next comes the WS component of my PWS theme. I can cover here only the earliest developments of what was to be a long term evolution of computer graphics and keyboard language techniques, both hardware and software, well into the 1960s, at MIT.

DIRECT KEYBOARD INPUTS

It's funny how memory plays tricks on us. For years I have known that it was at my instigation that direct keyboard input to Whirlwind was installed (beginning the chain of events that eventually led to the Compatible Time Sharing System (CTSS) and Project MAC at MIT in the 1960s). But at the same time, I thought that all of our MIV and Charactron console design and installation for the Eglin Field 1103 had come after and had been based upon what we already had going on Whirlwind. The records show that I was mistaken. The Whirlwind keyboard proposal actually *followed* my proposal for the 1103 keyboard. That came as a surprise to me, when writing this paper.

The earliest date I find in my own files regarding the ERA 1103 is a collection of general information (brochure description and installation instructions—where to install the motor generators and air conditioning, etc.) bearing date stamps of John Ward and myself of November 8, 1954 (ERA 1954a and 1954B). Two yellow pages of notes about the order code indicate that this was my first introduction to what was in store. The earliest Charactron information is a collection of Convair Charactron Project documents on the Model 70B Charactron Display Console, the Model 100 Computer Readout System, and the Rapro-

matic Camera and projection system, all with my date stamp of October 18, 1955 (Charactron Project 1955).

By December 14, 1955, I had written a memo on "Proposed Equipment Modifications for the Pre-B-58 Data Reduction Programs" (Ross 1955g). It recommends that AFAC should procure an additional 4096 registers of magnetic core memory beyond the 1024 electrostatic storage registers that were standard for the 1103, and "that the photoelectric tape reader and the Flexowriter connections be modified," in addition to installation of the "Type 70-A Charactron with photographic facilities under computer control, reported already on order," as well as "the Intervention Switch facilities, reported already on order."

Engineering Research Associates (ERA) definitely was a company of (outstanding) engineers—not programmers. I believe the AFAC 1103 was one of two "Serial One" first models of the ERA 1103 (Cohen 1985), and I encountered several features that were good engineering ideas, executed in such a fashion as to make programming and use most awkward. The 1103 had a 36 bit word length, and an "arithmetic section" with accumulator (A of 72 bits), shifting register (Q of 36 bits), exchange register (X of 36 bits), and two input-output registers (IOA of 8 bits and IOB of 36 bits). The binary form of 7-channel punched paper tape, called "bi-octal," was prepared off-line using a box full of relays with a 10 character per second mechanical tape reader on one side and a mechanical tape punch on the other. A number of empty sockets into which jumper-wired plugs could be inserted allowed the user to control the tape transformation between the two. Absolute address assembly language tapes punched on an off-line Flexowriter (Fig. 12) could be converted to bi-octal with one set of plugs.

We contracted with the MIT Digital Computer Lab to have a *WWI-1103 Input Translation* program, based on CS, built so that we could write 1103 programs with "flads" (floating addresses) etc., translate them on Whirlwind, fly them to Florida, and debug and use them there [WW1Q55p56and79-81]. The 1103 itself had no such software. I wrote a companion 1103 program, handling only absolute addresses, to support our on-site debugging efforts (Ross 1955b).

For this early machine, things were pretty awkward. The ERA photoelectric tape reader (PETR) could only be started manually, the PETR was connected directly to the Q register, thus entangling the Arithmetic Section unnecessarily, and worst of all, as my memo says:

> The bi-octal tape format now in use does not allow the use of programmed checks to insure that the information was read from the tape correctly. Since the read-in program for reading bi-octal tapes is now wired into the computer, the increased flexibility required to change tape format is not presently available.
>
> There are two major approaches to the problem of placing the photo-

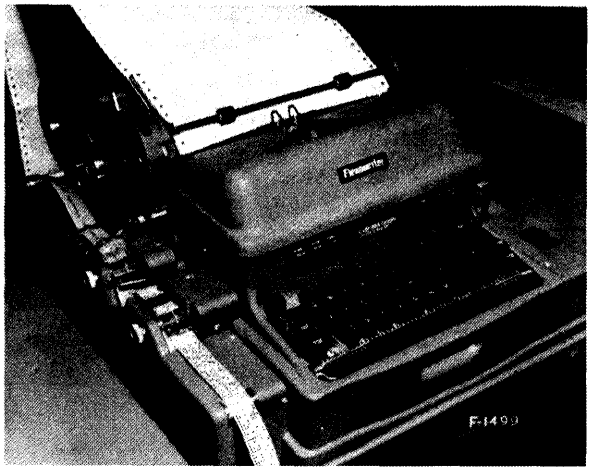


FIGURE 12
Flexowriter with 10cps reader (front) and punch (back). Keyboard was not used for Whirlwind direct input until summer of 1956, which led to later timesharing projects at MIT.

> electric reader under computer control. A Ferranti tape reader could be purchased and installed exactly as in other 1103 computers. This is the best, but most expensive solution. The other possibility is to put the drive motor of the present ERA tape reader under computer control, and to have the photocells read into IOA (or IOB) rather than into the Q register. The only disadvantage to this inexpensive method is that a few inches of extra tape feed-out must be placed in the tape wherever it is to be stopped, since the ERA reader is not a fast start-stop device like the Ferranti reader. However, this method was used successfully with a similar ERA reader for several years on the Whirlwind computer at MIT, before the Ferranti readers were obtained. [p. 2]

(The inexpensive method was chosen, and the WWI-1103 Input Translation program was forced to ensure the proper blank tape sections were present.)

The memo then continues with similar observations regarding the keyboard:

> 6. *Flexowriter Input*
>
> To augment the Manual Intervention facilities mentioned above, it is suggested that the Flexowriter presently attached to the AFAC 1103 computer be connected to act as both a keyboard input and mechanical tape reader input device. This method is considered preferable to the purchase of special keyboard input devices at this time because, again, it should be very inexpensive, and will give considerably more flexibility than any commercially available keyboard of another type.
>
> An essential part of Manual Intervention Programming is that decimal (or octal, or English) information can be communicated directly to the computer. In use, this system would use intervention switch settings to tell the program how to interpret the Flexowriter keyboard input at any given time.

> The mechanical tape reader probably could use the same circuitry as the keyboard and would be used to read control tapes (while data or program tapes are in the photoelectric tape reader) so that manual control will not be required for routine operation of the computer system, which will result in greater reliability.
>
> A comment contained in 1103 Newflash No. 15, December 7, 1955, seems to imply that the keyboard of the Flexowriter on the Ramo-Woolridge 1103 is already connected in this way, so that the required engineering may be already available. The comment does seem to imply, however, that the "directly connected typewriter" uses the Q register. It is much better to use IOA (or IOB) for this purpose so that normal operation of the arithmetic element is not impaired. It is very possible that both the keyboard and mechanical reader could share a major portion, at least, of the circuitry connecting the photoelectric tape reader to IOA (or IOB), with resulting savings in engineering and hardware. [p. 3]

It is interesting to me that even though I chide the engineers for their lack of vision, I suffered similar myopia regarding how to establish controls. I'm sure that it would not have taken significantly more programming to have an opening character or so to specify what type of information was on the tape, instead of using MIV switch settings. This had, in fact, already been done for the logging of CS tapes on Whirlwind, after all, when Director Tapes were introduced. Our "modern" views take time to evolve and develop, I guess.

In any case, this December 1955 memo shows that the AFAC 1103 keyboard suggestion formally came before the Whirlwind keyboard suggestion, and in a fashion that surprised me a second time, in my researches for this paper. I had found in my files my memo 7138-M-144 "Suggestions for Manual Intervention Facilities on the Air Force Armament Center ERA 1103 Computer," dated March 1, 1956 (Ross 1956c), and my memo 7138-M-148 "Flexowriter Keyboard Input to WWI, Preliminary Specification," dated March 14, 1956 (Ross 1956d). But digging still deeper I found that my memo 7138-M-160 "A Proposed Flexowriter Input for the Whirlwind I Computer," dated April 30, 1956 (Ross 1956f), was the formal proposal finally sent on to Lincoln Labs for formal consideration, with the concurrence of the Scientific and Engineering Computations Group and "several groups in Lincoln." So at this point in my sleuthing, I was convinced that the actual sequence for my on-line keyboard suggestion definitely was first 1103, then Whirlwind.

The second surprise came from a detailed reading of collected draft papers for a May 8, 1956 revision (to "144A") of my 7138-M-144 March 1 memo, this time done by John Ward (Ward and Ross 1956), for formal forwarding to AFAC and our WADC sponsors. In that revision, which includes John's redesigns of Whirlwind drawings for intervention, activate, and audible alarm circuits, is a large fold-out drawing

entitled "Keyboard Read In System, WWI," showing Lou Norcott, "Eng." and Larry Holmes, "Appd." both dated "2-27-56"—with the hand-scribbled notation "see him [Norcott] or Doug R. O.K." The interesting surprise to me is that this shows that I already had been active during February, and probably before, working behind the scenes with the Whirlwind engineers to confirm feasibility and costs before sticking my neck out with formal suggestions. Therefore the actual fact is that I can't sort out which came first, my thoughts regarding the 1103 or Whirlwind. My guess is that they surfaced almost together. That is, I knew general keyboard input was needed for gestalt programming, but the fact that the ERA 1103 already *had* an on-line keyboard of some sort triggered me into action. I first worked the idea out on the basis of Whirlwind, working with Larry Holmes, but the first actual writing was about the 1103.

Larry and I played the game to the hilt, for I also have a short interoffice memo from him to me, dated April 3, 1956 (Holmes 1956), which begins:

> I received a copy of your Memorandum 7138-M-148. The attached chart illustrates the five different plans that you might consider in the determination of a decision as to which installation you will request.

And then goes on to conclude:

> I sense that it would probably be the middle of June before the chosen facility would be operational.
>
> If any additional information is required for your subsequent memorandum to management, I will be pleased to procure same.

MICROPROGRAMMING FOR WHIRLWIND

It was not all that unusual that I was in close touch with the Whirlwind engineers, for another one of my on-the-side, recreational projects at that time concerned a plan I had to modify the internal logic of Whirlwind I to allow experimentation with microprogramming. This idea had come to me in the same spurt of inspiration when I studied the Lincoln-only, "si"-based input-output lines that led to the original director tape proposal. I discovered that not only was the mechanical tape reader still available (as I have already covered), but much more importantly that there was a whole additional flip-flop register, called PR—the *program register*—just sitting there, completely unused! The PR had temporarily held the memory address for electrostatic storage operating and was no longer required for magnetic core memory. It still had all the wiring that connected it to the WWI Bus, and had three outgates per digit! What a treasure trove! In those days of discrete components and tubes, flip-flops and gates were *very* expensive (several hundred dollars per bit), and here was all that gear, cut out of use by

the modifications to the main machine, but all ready to be fixed up if I could find a use for it.

Of the papers I still have, the earliest date stamp I find is that of "Engineering File #2, August 18, 1955" on copies of large drawings of "Timing Diagrams, WWI," Digital Computer Lab drawing E-51744, and "Control Pulse Output Connections, WWI," DCL drawing D-531?? (torn off), which were made for my use. My own vellum master drawing of "Proposed μ-Operation Schematic for *mi* Instruction, WWI" is dated April 5, 1956, and numerous copies show my steps of refining my design after that date (Ross 1956e).

On March 1 and 2, 1956, a seminar on microprogramming, sponsored by the MIT Digital Computer Laboratory was held with papers listed by H. R. J. Grosch (GE), D. J. Wheeler (Cambridge U.), N. R. Scott (U. of Mich.), L. C. Hubbard (IBM), B. D. Smith (ERA), J. J. Eachus (Datamatic), H. P. Donahue (NSA), G. M. Hopper (Remington Rand), J. W. Wegstein (NBS), and J. W. Carr (U. of Mich.) (MIT Dig. Comp. Lab. 1956). I was not even on the list, because the Digital Computer Lab organizers didn't know about the hobby project I'd been doing only by myself, with a few interactions with the Lincoln Lab Whirlwind engineers. I hadn't even gotten an invitation to the seminar, but when I heard about it, I was welcomed, and went.

With a June 6, 1956, date stamp, I got pages of a typed transcript of the seminar from Frank Helwig that shows that at the last minute I had been squeezed into the program before Wheeler's second talk. As is usual for my oral speaking, the transcript is almost incomprehensible, but so is the extended ensuing discussion by others.

I did manage to complete the design of the proposed *mi* microprogramming instruction and my papers do include a complete outline of the report I intended to write, but never completed. (A "to-do" list, buried in these papers starts with "Flexo keyboard memo—Fri; AFAC int sw. diagram; Pre-B-58 MIT meeting; Gestalt Paper to AFAC, etc.; mi-WWI memo" and a slew of other items, including "Numerical Control Personnel," so I'd guess the date was around May 1956 when the outline was done.)

The idea of the proposal was as follows: When you walked inside the bowels of Whirlwind, down one aisle was a mass of bare copper wires and boards of components, neatly arranged in a matrix. First came five bare vertical wires, and then 32 long, horizontal, bare wires going down the length of the aisle. These, in turn, overlaid 120 vertical bare copper wires, each of which terminated in circuitry boards arranged along the bottom, by the floor. There also were eight coaxial cables running horizontally, with one or another feeding each of the lower circuitry boards. These cables carried one of eight *time pulses,* and the original five vertical wires supplied the 5-bit instruction code of the instruction now being executed. This was the control matrix of

Whirlwind (Fig. 13). The neat thing was that soldered right there in front of you, at the various crossing points of the 5 vertical wires and the 32 horizontal wires, was a collection of germanium crystal diodes that spelled out the binary number system, so that each of the horizontal bare wires represented one of the 32 instructions of Whirlwind. To understand what each instruction did, you had merely to similarly read the pattern of diodes soldered between its horizontal wire and the 120 vertical wires. The 120 vertical wires each selected one of the *120 CPO units* (command pulse output gates) of Whirlwind, each of which could do some micro operation (such as clear the I/O register, read from memory into the A register, etc.) when its corresponding time pulse arrived. In a nutshell, my plan was to use the flip-flops and gate connections of the PR to dynamically compose instruction steps from a selection of the existing CPOs—as though a new horizontal line had variable diode connections to the CPOs.

The report outline indicates that I found a subset of 39 CPO micro

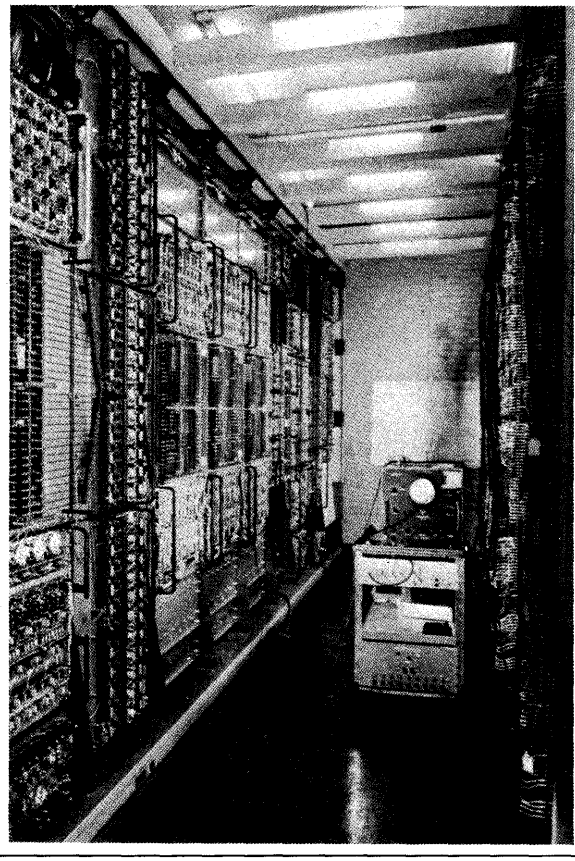


FIGURE 13
Control matrix of Whirlwind. Wires, diodes, and gate boards laid out translation of 5-bit instruction code into 120 command pulse output (CPO) micro operations.

operations that would be useful in various combinations. My design allowed any number of them to be chained together on time pulses 6, 7, 8, and 1, interspersed with memory addresses and I/O operations. Thus microcoded instructions of any length and complexity could be composed and executed, interspersed with standard Whirlwind instructions. As the outline says, "Very fertile field for problems which can use micro-program in MIV Gestalt Programming." The report was never written, however, and the change to Whirlwind was never actually proposed (I had in mind a big gang switch that would include or exclude the *mi* instruction in Whirlwind's repertoire) because when I programmed various test cases with microprogramming and then with ordinary Whirlwind programming, there was very little to be gained. The trouble was that Whirlwind was so closely designed (to save expensive gates, etc.) that the A register formed an insurmountable bottleneck. As the outline says ("md" is "multiply digits"—a masking operation):

> Fundamental difficulty with present WW construction for this type of application since only entry to AC is thru AR so that usually AR is not indep. of AC. Also this necessitates use of more steps than really required since first must load AR before can work on AC. Also would be good to have a working operation built into circuitry to give effect of md's since in logical ops this is often fundamental but is lengthy μ-op-wise or md-wise since another mi is necessary to get back to μ-mode.
>
> *Conclusion*
>
> Study of this report and independent investigation of this kind of approach to μ-prog is very instructive on problems involved. If more major changes in WW structure were acceptable might be able to obtain a useful gadget. Best bet, however, is to contrive basic idea of investigating μ-computers by similation without changing WW.

So my grand scheme for microprogrammed MIV and Gestalt programming came to naught.

KEYBOARD INPUT FOR WHIRLWIND

The direct Flexowriter input for Whirlwind was built, however, and could be operated from either the E31 console or from Test control, as I had suggested. The Second Quarter 1956 Whirlwind Progress Report says:

> Since 1954, the MIT Servomechanisms Laboratory has been using the WWI manual intervention and display equipment in the development of high-speed data reduction techniques. In order for them to expand their research into computer applications, it was essential that more

versatile manual inputs be made available on the WWI computer. Besides requiring additional on-off switches, many of the new programs will be so complex and will require so many parameters that the only reliable way to instruct them will be to use specially designed mnemonic languages and translation programs. In order to have this general language structure available on a manual intervention basis, it is necessary to have a keyboard such as a Flexowriter for direct input to the computer.

The MIT Scientific and Engineering Computations Group have contemplated the following applications for the new facility:

1. Demonstration programs would be a great deal more effective if this form of input were available for control purposes.

2. Typewriter input for Comprehensive System Flexowriter and post-mortem request tapes. Short program modifications and post-mortem requests can presently be inserted in the insertion registers. However, errors are easily made because the required vocabulary is awkward. A typewriter input facility would make available a normal mnemonic vocabulary for such purposes.

3. Experimental use of a typewriter facility for direct operator control of the computer. Here we would consider using the typewriter to replace the button-pushings required of the operator during normal operations. Vocabulary similar to that of director tapes and performance requests would be devised for these purposes. This could easily prove to be an extremely convenient and efficient method of computer operation.

The new input installation will be available for use by 4 July 1956. Much of the information to be inserted via the keyboard will be the same as is now introduced via a free running photoelectric tape reader using punched paper tapes. The keyboard input will also be treated as a free running device, i.e., selection of the facility by the computer may be followed by an arbitrary number of read instructions, each of which reads the next character which has been struck on the keyboard. The total equipment requirements amount to 15 relays and 20 tubes. [WW2Q56p63]

My original M-148 proposal specified that

since much of the information to be inserted via keyboard will be the same as is now on paper tape, it would be best to have the keyboard operate in a free running mode, i.e. an si selecting the keyboard may be followed by an arbitrary number of rd's, each of which reads the next character which has been struck on the keyboard. In this way, programs such as the entire CS system can be modified to accept keyboard rather than tape input merely by changing one or two si instructions. (Ross 1956d, p. 2)

The existing Test console Flexowriter (Fig. 14) was selected by one si, and another Flexowriter on a wheeled table plugged into the room 222 E31 console and was selected by a different si. We used it some with

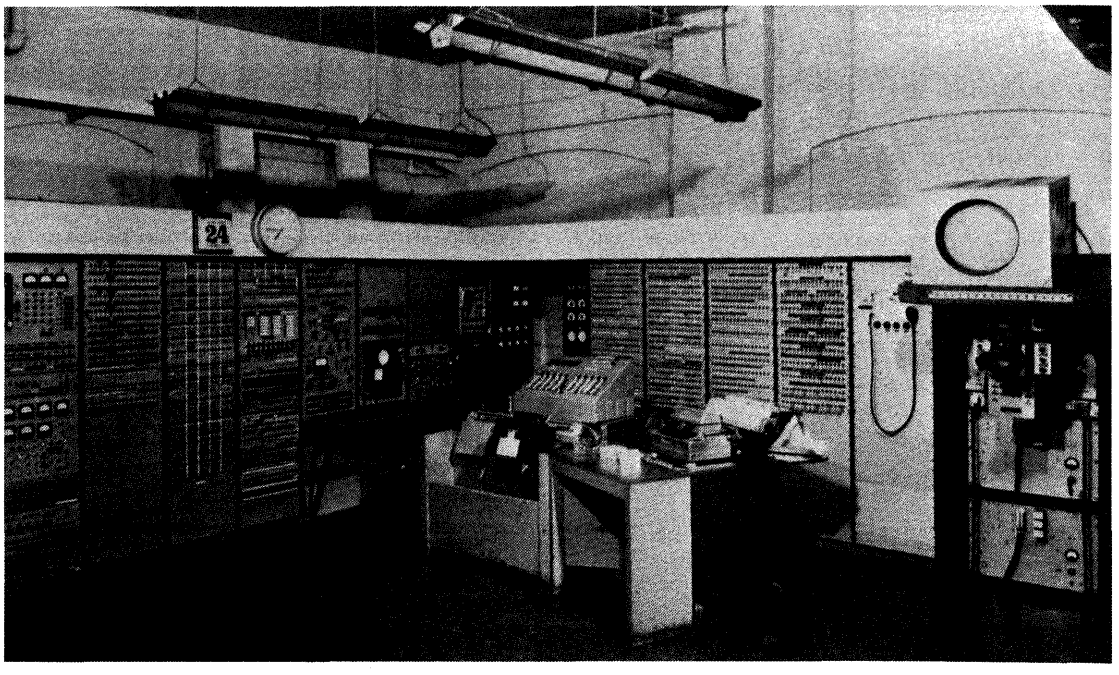


FIGURE 14
Whirlwind Test Control console (1/2/57) showing MIV panels for computer operation behind photoelectric tape reader (PETR) and Flexowriter. Monitor (above) and camera (below) output scopes are on right.

our SLURP developments, but Lincoln people didn't take advantage of it, to my knowledge.

While digging through my papers, much to my surprise I ran across a Division 2 Lincoln Lab memorandum dated February 2, 1959 (three years later) on "The Direct Flexowriter Keyboard Input System at Whirlwind I (Barta)" by C.F. Brackett in which I discovered that

> A pushbutton on the direct output Flexo table [in Test Control?—D.T.R.] allows the Flexo keyboard to be used as a direct input device. If this pushbutton is held depressed and the Flexo keyboard operated, the Flexo code for that character will be set up in the keyboard Input Relay Register where it will be available for later transfer to the In Out Register. . . . Since the keyboard Relay Register is cleared of old information during the read in of new information, the operator should normally wait until the indicators are extinguished before typing the next character.

I never knew it had such a button. Maybe it was only on the Test console Flexowriter. If the Lincoln memo indicates what actually happened when my idea was passed through the Lincoln approval cycle (I never had occasion to use it myself, as I remember), no wonder that

first installation was little used. The button would force pick and peck typing! This equipment did, however, serve as the basis for later direct keyboard inputs to the IBM 704, 709, and then 7090 computers at MIT, which led to the initiation of time sharing.

THE CHARACTRON/MIV CONSOLE FOR THE AFAC 1103

The Charactron/MIV console for the Eglin Field AFAC 1103 as finally proposed had two 36 bit intervention registers, one 8 bit activate register, one 36 bit indicator light register, a 7 bit MIV flexo register, and two alarm buzzers (Ward and Ross 1956). John Ward and I supplied reworked Whirlwind drawings for all these items plus the Whirlwind drawings for the Ferranti PETR and manual tape winder at various times (Fig. 15). All of these were forwarded to Stromberg Carlson, San

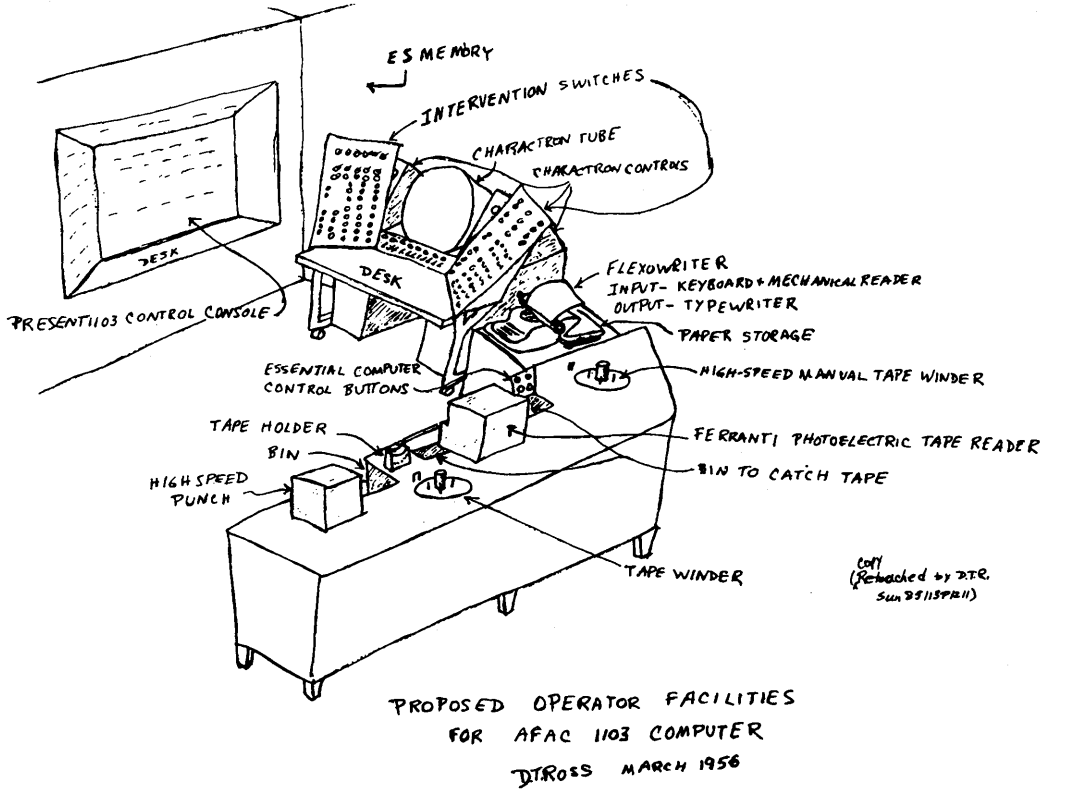


FIGURE 15
Sketch for Charactron/MIV console for ERA 1103 at AFAC, Eglin Air Force Base. Built by Stromberg Carlson based on Whirlwind experiences of Servo Lab Data Reduction Project.

Diego who reworked the designs further and incorporated them with a 19 inch Charactron for viewing and a 7 inch Charactron with camera, in the final, delivered installation.

My *daily resumes* began at about this point, and various phone calls, letters, and visits to MIT by Ed Allmon and various people from Stromberg Carlson are recorded from October 31, 1956, on. Topics included the vector generator and tape winder [R561031]; Ferranti PETR [R56116]; PETR again, plus the fact that they got the contract from AFAC [R561115]; visit regarding Flexowriter, tape reader, and 1103 MIV system [R561126-27]; connectors and pushbuttons [R561210-11, -12]; calls from their psychologist regarding switch types and layouts (considering a black background for the lights, etc.) [R561218-19]. They finally sent a design study document to us on January 1, 1956 (which I don't have), but still there were more Flexowriter questions a week later [R570118]; the Ferranti automatic turn-off to save motor wear [R57314]; a press release [R57417]; request for a San Diego training program for use of the MIV equipment [R57429]; and finally "the Charactron and MIV equipment turned up at AFAC today," on August 14, 1957 [R57814]. But I have no photos or descriptions of what finally was delivered.

A letter from John Ward to John Moser (Stromberg Carlson) on October 24, 1957, congratulates him "for a very fine job" and says we "hope to be making good use of it within a month's time" [C571024].

John Walsh (one of my programmers, and still with SofTech) labored many months on an elaborate 1103 program for extendable, labelled plots of functions for the Charactron as a major part of the AFAC system (Ross and Ward 1957). But AFAC programmers always were too short of time to do much elaborate MIV programming. The system was indeed used for the Pre-B-58 and B-58 tests, but I don't know any more about it than that. It's claim to fame really is that it was the first working work station explicitly designed for general purpose use.

MIT MIV FOR IBM COMPUTERS

In the summer of 1956 preparations were well under way for the installation of the IBM 704 computer in the MIT Computation Center. This also was the time that my Computer Applications Group was officially christened, as we acquired responsibility for the (then unnamed) APT project in addition to our other projects. My first and second memorandum for the group (Ross 1956g and 1956i) concerned an assessment of the impact of the 704 on our work. The second memo begins:

> This memorandum is a supplement to Memorandum CA-M-1, "Servo mechanisms Laboratory Requirements for Computing Facilities 1956–1957". That memorandum stated that Project DSR 6873, on the devel-

opment of automatic programming techniques for numerically controlled machine tools, would be transferred to the IBM 704 computer as soon as it became available, but that the work of Project DSR 7138, on the application of digital techniques to airborne weapon systems, would continue to need the facilities of the Whirlwind Computer System because of its dependence on manual intervention techniques. This memorandum considers in more detail the reasons why it is felt that the Whirlwind I Computer must continue to be used, by describing the system of Whirlwind programs which are now being developed for this work, and then considering the modifications to the proposed IBM 704 system which would be required to make this type of system possible using that computer. [p. 1]

Then, after a description of SLURP, it goes on:

The IBM 704 Computer

Aside from the large amount of programming which has already been completed, it is apparent that SLURP is independent of the particular computer used, provided it has the appropriate input-output devices. The manual intervention features of the system are central to the philosophy of the system, since without them it becomes merely another elaborate computer programming system. The true significance of the system is that it allows a programmer to conceive of a new method of solution and maintain his momentum and initiative on the problem, unencumbered by the restrictions of the coding system of the computer. In other words, the programmer is permitted to program with programs, and "program with ideas."

It would be extremely desirable to have the larger storage capacity and longer word length of the 704 computer for use in SLURP. It already appears that the capacities of the Whirlwind Computer are being taxed by the size and complexity of SLURP. The present incomplete version contains approximately 10,000 instructions in addition to the 20,000 or so in the comprehensive system, and to this figure must be added several thousand registers required for the storage of a satisfactory amount of data for the problem. Since a large portion of the programming for SLURP consists of generating and interpreting coded information, the short word length of the Whirlwind Computer becomes awkward on occasion, and slows down the operating speed of the system due to increased complexities. Since the Share assembly program and the algebraic coding system being developed for the 704 installation by Computation Center personnel [never carried out] will be more modern and flexible than [p. 4] the Comprehensive System in some respects, and may be expected to expand to include those features of the Comprehensive System which are unique and desirable, it is felt that SLURP efficiency could be greatly improved by the use of those systems. Except for the fact that the IBM 704 will not have manual intervention and oscilloscope-type input-output devices, it would be very desirable, and probably worth the additional programming effort, to transfer SLURP from the Whirlwind Computer to the 704 Computer.

> There appears to be some indication that there is some interest in applying this type of equipment to the 704. The Servomechanisms Laboratory should be active in support of this thinking. Systems such as SLURP seem to be the next logical step in development of improved programming systems for modern computers. The fact that work for Project DSR 6873 for the automatic programming for numerically controlled machine tools is being formulated and solved within the SLURP structure demonstrates that the philosophy of problem formulation and programming which is embodies [sic] in SLURP is by no means restricted to the type of military problems which have fostered its development and for which it is primarily at present intended. The possibility of detailed research into automatic process control and managerial business decisions should provide ample justification for the active consideration of these techniques as an integral part of the MIT Computation Center facility.
>
> *Conclusions*
>
> It is felt that the problems presently being considered in the Servomechanisms Laboratory cannot be successfully solved without the use of SLURP so that until an equal facility becomes available elsewhere this work is committed to the Whirlwind Computer. It is felt also, however, that the concept of systems such as SLURP is a significant advance in the use of computers as data processing devices, and that active consideration should be given to this type of system as an addition to the facilities of the MIT Computation Center in the near future. It would be necessary to expand the equipment planned for that facility to include manual intervention input devices and oscilloscope-type output devices in order to realize this type of system. The experience gained from the use of this type of equipment in this application in the Servo Lab SLURP system, as well as the experience of the Sage system development, should provide an adequate foundation for the planning of a Computation Center facility of this type. The potentialities of this type of a system as a research tool for experimentation in the newly unfolding area of general data processing are very great. It is felt that it would be in the interests of the Institute as a whole to pursue this line of thought. [p. 5]

My resumes show that on November 1, 1956, F.J. Corbato (still a professor at MIT) "called to inquire about Servo Lab people giving talks in Digital Computer Center [sic] seminars. He is now on their staff and is in charge of these seminars. I said that we might talk about SLURP and our general problem solving system" [R56111]. On November 6 "should possibly consider obtaining support from Mission Director Project [a new component of John Ward's Bomber Defense project] for attaching MIV equipment to the 704. . . . Will IBM let such equipment be attached to the computer and also how do we get enough computer time to make the investment useable?" [R56116] This was a problem we later solved by joining as a cosponsor of the

MIT Cooperative Computer Lab's IBM 709 in Building 10 in the early 1960s.

On November 27 I gave my Comp Center talk on "SLURP: An Experimental Human-Computer System for Problem Analysis and Solution" [R561114], and the interesting note is that Wes Clark (then doing TX-0 at Lincoln) suggested, in the discussion, "that the switches be programmed on the tube and thrown by means of a light pencil [sic]. In this way a programmer can do all of his own human engineering of switch layouts as well. [We later called this "light buttons" (Ross- and Ward 1961, p. 80).] Slurp is already set up so it could do this operation merely by writing an appropriate Slurp program. Wes Clark would like to have several copies of the Gestalt paper and I will plan to take them out when we visit TX-0 tomorrow." Also at the talk, Frank Verzuh (Director of the MIT Comp Center) "said he was interested in talking with me about getting this kind of facility for the Computation Center" [R561126-27]. And the next day "Gave a stack of Gestalt papers to Wes Clark. He seems to be quite interested in what we are doing and there may be a possibility of pooling our interests if not our efforts in making TX-2 specifically designed for Slurp type of problem solving. Should plan to discuss this with him and his group in the near future." [R561128] We did collaborate on light pen design and improvement, under John Ward, after TX-0 was moved to MIT (I heard it was scheduled to move on June 27, 1957 [R57627]), but actual collaboration never took place.

Not much happened during 1957 on these ideas. TX-0 moved from Lincoln to ESL, with Earl Pugh in charge. In September, Whirlwind "passed into Lincoln's hands" [R5794-9] and we continued our developments there and on the AFAC 1103, while doing APT work also on the 704. By year end "A summary-progress report for the mission director is scheduled . . . [writing about SLURP will serve to] explain the whole problem-solving philosophy which I have been evolving over the last several years. [This will serve] as powerful ammunition for battling to keep Whirlwind going for a long time" [R57124-58019]. I cover the gestalt programming to SLURP transition in the next section, briefly.

In January 1958, Peter Elias (then EE Department Head) suggested to John Ward that I should get together with John McCarthy, Marvin Minsky, and Richard Marcus (all then at MIT's Research Lab for Electronics, RLE) about my use of MIV Techniques [R58019-28], and John Ward wrote me a memo to that effect (Ward 1958). John's memo says Dick "is working actively on function display under Manual Intervention Control." Later Dick came to work in my group and still is at MIT. By January 30, I finally arranged to have a lunch with McCarthy, which then transferred to Minsky's home in mid-afternoon, "and we talked some more until around 2:30 in the morning. Oliver Selfridge's

group at Lincoln with whom Minsky works has $25,000 a month which must be spent on equipment, not manpower. . . . They are presently thinking of getting the [standard IBM] real time package for the Lincoln 704 and are interested in investigating whether the Convair MIV package that we worked out for AFAC would be within their budget'' [R580131]. I don't know whether Oliver contacted Convair directly or what he ended up doing, but this session did begin a long, sporadic interaction on equipment matters with John and Marvin. A week after our long session I loaned John McCarthy ''our information on the Stromberg Carlson [Convair] MIV equipment . . . informed him of our present investigations and said . . . perhaps he would be a good one to work on the automatic programming features since he has been working quite a lot on a super compiler with all sorts of logical statements, possible in it which may fit in well with the system we have been working on. We shall see'' [R5827]. LISP was in the wind, and I was in the midst of APT, of course.

Again nothing happened regarding MIV equipment for months, although a talk I gave (on APT) at IBM Kingston Labs evoked brief interest on their part in scope input and MIV for their IBM 704 [R58815-20]. In September, Dick Bennett (who then had a consultant company) was under contract to Lincoln to upgrade the Whirlwind utility system, and he thought Frank Heart's group ''also should be interested in being able to operate everything from the Room 222 console. . . . We also discussed. . .putting things under direct typewriter control'' [R58924]. But most of Dick's plans were not carried out.

Even though the preparations for the APT press conference were pressing, I was concerned about the facilities for follow-on work, so in early January 1959 I said:

> Whirlwind definitely is going to close on June 30 and since it is poor politics to fight it at this time, we are going to go along with it, so that we have to find a substitute. My thought is that by combining TX-0 and 704 we could have a really fancy facility since TX-0 with its newly expanded memory, which is not in yet, of 8,000 words should be able to do [display and light pen support]. . . . We are having a meeting . . . between [Dean] Arden, Jack Gilmore, John McCarthy, myself, Arnie Siegel, and [?] who will be in charge of the Lincoln 709 programming to consider using the real time package [IBM's terminology] on the 709 to simultaneously service about 20 Flexowriters in TX-0 fashion. . . . We are going to start off with one Flexowriter on the 704 real time package. [R581229-59019]

This was the start of MIT's time sharing developments.

In January 1959, ''Attended a meeting in Dean Arden's office concerning 704 Flexowriter input . . . I didn't feel that the group accomplished very much'' [R590112-21].

In early March, ''John McCarthy is calling a series of meetings to

consider operator and compiler programs for the 7090 computer which will be the transistorized version of the 709 which will replace the 704. We had a meeting yesterday [March 3]." I spoke about MIV and Group Control. "I am hoping that in later sessions [we can show that MIV] should be an integral part of the programmer's bag of tricks applicable to any problem" [R59219-34]. But by the end of the month "The meeting to discuss 7090 problems seems to have lost momentum, so it is unlikely we will have another one very soon" [R59323-331].

In April,

> With the imminent breakup of Whirlwind, we have started to consider putting a good manual intervention console on the 704. I have now decided to soft pedal the connection of TX-0 to 704 until we have a definite use. We got together with John McCarthy and Herb Teager who is now putting the Flexowriter on the 704. [I believe Arnie Siegel, who knew Whirlwind well, had done some early preparations to adapt our earlier work.] John Ward and Don Clements are going to assist in this and it looks like our first move will be to get a light pen hooked up to the 704 to use the present scope. . . . In the long term, however, we are thinking of developing a self maintaining display system, [Later the ESL Display Console (Ross and Ward 1961, pp. 79–92), which was Rob Stotz's Masters thesis, came from this idea.] possibly built out of Whirlwind components. There is quite a bit of engineering involved there, however. McCarthy and Teager are going over to see the Whirlwind equipment with us at 11 a.m. [R59416-511]

On April 16–18, 1959, a select conference on Symbolic Manipulation (organized by McCarthy, Perlis, and Newell [R59219-34]) was held at MIT (McCarthy 1959). This was the first public presentation of LISP by McCarthy, COMIT by Yngve, and "I talked for a short time on multimode control as applied to list searches, group control, and proposed a modified list structure which seems more appropriate to our design machine application" [R59416-511]. This was my "reversed index registers" (as I later called it) method of general pointers and "n-component elements" of Plex Programming (Ross 1961), which began what now is the field of abstract data types. Evidently Al Newell made no presentation at MIT, but "On May 4, 5 and 6 I attended a conference on self organizing systems . . . in Chicago. [Some papers] were quite good, . . . [especially] one by Allen Newell on the GPS system general problem solver that he is working on with Simon and Shaw. . . . will plan to get together with him" [R59416-511].

On May 12, I took McCarthy, Teager, and Don Clements (then project engineer for the APT Project) "to look over the Whirlwind equipment. They seemed satisfied enough but our present thoughts are to find out what is going on at Lincoln, too" [R59512 & 13]. A memo from me (Ross 1959) shows that Arden, Clements, McCarthy, Minsky, Teager, Ward, and Ross visited Lincoln (re 709, charm, TX-2,

and SAGE) on May 19, but "Our meeting out at Lincoln . . . was pretty much of a farce. People were late or missing or not very sympathetic . . . so not much was accomplished" [R59518-20].

THE DESIGN MACHINE

So as 1960 rolled around, various seminal equipment developments at MIT had been initiated, but they are better treated as components of the history of time sharing, computer graphics (hardware and software), computer-aided design (CAD), software engineering, and software technology, rather than merely as extensions of the personal work station theme of this paper. I hope to be invited sometime in the future (though not too soon, for these papers are excruciating torture to write!) to contribute to the chronicling of those developments, as well.

Any such future consideration of CAD work stations would have to begin with a mention of George Price's prescient article in the November 1956 issue of *Fortune* Magazine, (Price 1956) and since it, too, was a "first in the fifties," I must reference it here.

Going back again in time, in January 1957 I said

> Bill Webster [Air Force contract monitor for APT] also had a copy of the article on a design machine which I had been told about in New York. It's amazing the similarity between what is in that article and what was in my Gestalt paper and what we have been working on the past year or so. He even has pushbutton language with sentences of the form that we are planning for our milling machine, and the computer drawing pictures of the part being designed on the scope just like our scope plot, etc. I plan to write him a letter describing our work and enclose a copy of my paper and send a copy of this correspondence to the editor of Fortune and see what results. I think it is quite interesting that we have been actually carrying out what in his article is merely a proposal which everybody feels very strongly isn't as wild as it seems. Webster wanted to check with me that his impression that we were doing just what the article called for was correct, and I assured him that it was. [R57017-8]

I finally got a library copy to read by January 25 [R570125], and on February 6 wrote to Price, enclosing a Gestalt paper reprint, and describing our APT progress, then being issued as the first APT Interim Report. I then go on:

> These routines provide the necessary mathematical structure around which a convenient human language can be built to make a complete automatic programming system. The detailed specification of this human language has not been attacked yet, but it will be the major focus point for the efforts of the group. The language will have specially designed features for description of surfaces and their interrelation, as well

as for the instruction of the machine tool itself. The form of the language will probably be similar to that which you propose for the design machine, and as outlined in my paper, although it will be a written language and not use push buttons initially [—used in the Gestalt paper and in Price's article]. . . .

Another important part of the language problem is the computer-to-human language, which we envision in two forms. First there will be the ability of the computer to "talk back" to the human in the same type of language which the human uses, probably in a written form. The computer's language also will be of a pictorial form in which the computer will draw pictures for the human to check and work from in a way almost identical with your proposals in the FORTUNE article. A rudimentary routine of this sort has been in operation on the Whirlwind I Computer for several months now, which draws on the output oscilloscope of that computer arbitrarily positioned and scaled axonometric projections of the part being made. True perspective will be programmed later if it seems warranted. At present the picture consists of the sequence of "cut vectors" which are used to program the machine tool, but more elaborate schemes are planned. At the present time the pictorial displays are photographed and no effort has been made to increase the speed of the program so that a cycled display can be viewed easily on the display tube itself. Memotron-type tubes are not presently installed on the Whirlwind Computer although they would be very desirable for this and other applications. [The ARDS Display of Rob Stotz was the first such memory tube display (Ross and Ward 1961, pp. 100–113).]

The striking similarity between our work and the proposals of your FORTUNE article indicate that we have a number of very strong mutual interests and a similar approach to these problems. Mr. William Webster of the Air Materiel Command, who is our project monitor, brought your article to my attention and showed me a letter which you had written to him. In that letter you mentioned a study which you made concerning a design machine using the IBM 704 computer as a major component. I would very much appreciate a copy of this report if you have any available in order that we may know in more detail your ideas on this subject. We plan to transfer our work to the IBM 704 in the very near future.

I look forward with anticipation to further correspondence with you on this most interesting subject. [C5726]

A February 25 reply from Price enclosed a reprint of the *Fortune* article and his retyped "supplementary memo," but said he was changing jobs and moving, so he hadn't "been able to find the time to do more than take a quick glance at your two papers" [C57225]. On May 13 he visited my home and we "talked for an hour or so . . . but we never did get around to talking the same language or really discussing anything. . . . He has no technical training in this field" [R57513]. But he certainly wrote an interesting article, and his Design Machine

proposal surely was a first, including many features that later were used in actual CAD work stations. As a side box [p. 153] about "The Author and His Machine" said:

> Price began thinking about his Design Machine some ten years ago. Could the Machine be built today? One leading computer expert shown the proposal was skeptical. Price thereupon prepared a detailed supplementary memorandum demonstrating how an IBM 704 computer could be incorporated into a Design Machine, how a complex part could be described to the Machine, and how with the aid of certain auxiliary devices the Machine would display the part—in 3-D. Computer experts at IBM pronounced the memorandum eminently sound. The first skeptic conceded that Price had shown how the job might be done.

The main thrust of the article was "eight recipes for modernizing R. and D.," one of which was the Design Machine, and another to mechanize "reading" of technical literature for mechanized filing (with a reference to V. Bush and the Patent Office). Besides describing the pushbutton and joy stick preparation of stereoscopic pictures of parts and "invisible models—recorded in a magnetic memory," with "preparation of control tapes for guiding automatized machine tools," the article says "a single Machine might be shared among several companies scattered around the country, for a number of control stations can be in use simultaneously, with keyboard signals being briefly stored on magnetic tape, and the central computer switching from one station to another every few seconds." [p. 228] (An idea also promoted by Jay Forrester for Whirlwind, in 1948 [Redmond and Smith 1980, pp. 233, 234].) Price's article was well ahead of its time.

System Software

As I said at the beginning, a final essential ingredient for a viable, general, PWS scheme is a systematized way to link the work station capabilities to the class of problems to be solved through some form of software framework. As I have indicated, the earliest such framework, when Whirlwind had only 2K words of memory, took the form of several programming principles, such as "store switch values in variables," the language design principles of the Gestalt programming paper, and the rudiments of drum/ES storage management prompted by the mistake diagnosis routine (MDR). But with the advent of core memory, and its expansion to 6K words, and with the growing sophistication of approach engendered by early efforts at MIV in the evaluation program, a truly systematized solution was the natural evolutionary step.

THE SERVO LAB UTILITY ROUTINE PROGRAM

The first coherent expression of these ideas appeared in (Ross 1956h) (note—when APT work was just getting under way, as well), which said:

> After some experience with the integration of the manual intervention facilities into the previously fully automatic evaluation programs, by the Servomechanisms Laboratory, it was realized that the manual intervention features could be used as a direct aid to programming by incorporating a number of general-purpose utility routines into the system. Shortly thereafter, the new task concerning the investigation of the techniques and problems for an airborne mission director was undertaken. Since detailed specifications on the mission director problem were not then available, and also since the committment of the evaluation studies to the B-58 program made progress in that direction imperative, it was decided that a worthwhile merging of these two interests could be achieved. This was done by commencing the elaboration of the evaluation program on the Whirlwind computer into a general purpose problem-solving system combining the talents of the human programmer and the electronic computer for the solution of general computer problems. The result is a human-computer system which may be viewed either as an elaborate evaluation program, or as a prototype problem-solving program with the problem of evaluation of airborne fire-control systems as the motivating core. This memorandum stresses the latter viewpoint. [p. 1] . . .
>
> The combined evaluation and problem-solving system which is the subject of this memorandum has been given the name, the Servo Lab Manual Intervention System (MIV System). The main feature which distinguishes this system from other computer programming systems is that all of the major facilities of the system are instantaneously available under manual intervention control. [p. 3] . . .
>
> The facilities of the MIV System may be grouped into three categories according to their primary function and motivation. Some of the features are concerned with the problem itself (in the present case the evaluation program). Other features are strictly of a utility nature having to do with input and output of data in various tabulated and graphical forms. The third category concerns features which arise almost entirely from the fact that a manual intervention system is being used and have primarily to do with reliability and ease of operation of the overall system. The main features of each of these categories will be described in turn. [p. 3]

Five more pages then detail the plans for what came to be called "SLURP," just nine days later, in the second Computer Applications Group memo, mentioned earlier (Ross 1956i). The one-page summary of features, some of which were implemented only in later years, was:

> The Servo Lab utility routine program (SLURP) is a system of Whirlwind Computer programs which combine the manual intervention

features of that computer, with modern programming techniques, into a unique problem-solving system. SLURP includes all of the facilities of the Comprehensive System developed for the Whirlwind Computer by the Digital Computer Laboratory staff, plus a large number of general purpose routines for extracting and analyzing information about arbitrary computer programs. The governing philosophy of the system is to allow the programmer to work on one small part of a very large and complicated problem, virtually independent of the computer which is being used and those aspects of the overall problem which are not of immediate concern. The major features of the system which make this possible are as follows:

1. A group control program which allows the automatic incorporation of new sections of programming into the system.
2. A manual intervention (MIV) system of programs which allow the programmer to interrogate and instruct the computer with respect to the overall problem in terms of a specially designed and easy-to-use language represented by: switches, buttons, lights, and visual displays.
3. A logging program which records all of the MIV actions taken by the programmer.
4. An editing program which edits the logged information into an easy-to-read record of the manual actions taken, and generates a log playback tape.
5. A log playback program which automatically simulates manual actions in response to the instructions on the log playback tape.
6. An elaborate set of plotting and tabulating programs for data presentation and record keeping.
7. A mistake diagnosis routine (MDR) which may be used to abstract arbitrary intermediate computed quantities from any program and present these quantities in any of the above forms [including scope].
8. The SLURP program proper which is a simulated, generalized, special-purpose computer which allows the incorporation of all of these routines into a smoothly functioning system, along with the capacity for continued expansion of these facilities. [p. 3]

THE SLURP SIMULATED COMPUTER

The SLURP computer had many features that independently were discovered and incorporated into the Burroughs B5000 computer (I was astounded to see the similarities when I first read their manuals in the early 1960s). SLURP included built-in group control (memory paging), the memory table (virtual addresses for re-entrant program segments), alarm conditions (exception conditions with optional MIV and MDR facilities), and the ZIP Interpretive Program (which "serves the impor-

tant function of establishing the MIV language which is appropriate for a particular program'' and integrated all of the other features, including I/O in a powerful, very condensed macro-command-like language).

A unique feature of SLURP was its ''multimode control element'' (Ross 1958). As the definitive report on SLURP says:

> The instruction repertoire of the SLURP computer contains no mathematical instructions but instead concentrates on a wide variety of a) jump instructions for transferring control from one sequence of instructions to another, and b) instructions which control selection and adjustment of input-output equipment. Individual flexowriter typewriter characters are included in the instruction repertoire and the execution of this type of instruction involves the reproduction of these characters on a selected output device. A SLURP instruction may occupy an arbitrary number of Whirlwind registers. There are two binary digits called the A and C bits which are set aside in each instruction and are used to label individual instructions for execution in the various modes of the control element. . . .
>
> If the A bit is ONE and the C bit ZERO, the instruction will be executed when the control element is set to the A mode but will not be executed if the control element is set to the C mode [etc.]. All SLURP instructions (with the exception of a few jump instructions) are executed when the control element is set to the B mode, independent of the settings of the A and C bits. Since the mode of the control element may be set by jump instructions, the setting of the control element may be changed as often as necessary to accomplish a desired result. . . .
>
> In addition to the return jump instruction for termination of a remember jump sequence, it is also possible to remember jump to a flexo phrase (any sequence of characters or words), and the occurrence of a stop character in that flexo phrase will act as a return jump under certain circumstances. However, if that same flexo phrase is executed by encountering it in the normal sequence of operations (instead of by an entry via a remember jump), the stop character is ignored. In this way a lengthy flexo phrase which contains useful subphrases or words may be extracted from the entire sequence by means of remember jump instructions. (pp. 3–82 to 3–84)

This feature was used to have abbreviated labels for display and long labels for printouts, and the same technique gave complex behavior if the ''words'' were other instructions. Our favorite test case

THE SKUNK SAT ON THE STUMP
THE SKUNK THUNK THE STUMP STUNK
THE STUMP THUNK THE SKUNK STUNK

took only 13 SLURP instructions including output device selection [p. 3–85]. SLURP packed a tremendous punch in very little space.

THE MIV BOX

One of the most powerful features of MIV language design with SLURP was the *MIV Box* (Figs. 16 and 17), a re-entrant subroutine that could be used in many places [pp. 3–127 to 3–137]. The basic language idea was that

> if the human does not like what is presently going on, he merely needs to remember to depress an appropriate exit button. The appropriateness of the two exit buttons can be given a universal meaning by referring to one as a *major* exit, and the other as a *minor* exit. Then no matter what program is operating, if the human is unsure which of the two exit buttons to press, he may first try the minor exit and if that does not give the desired action, then pressing the major exit is guaranteed to work. . . . It is not necessary to know beforehand all the possible actions which may be controlled by exit buttons if it can be established that those programs will operate on a hierarchy of actions. The major and minor exits can then be associated merely with a change in level within whatever hierarchy exists for a particular program. This uniformity of meaning of language is achieved by combining the MIV Dispatcher, the wait switch, the major and minor exit buttons, and one new switch, the "cycle" switch.

The example shows how only two more switches allowed many choices of control for selecting and plotting successive frames of a sequence of functions, as in the evaluation program. One more switch

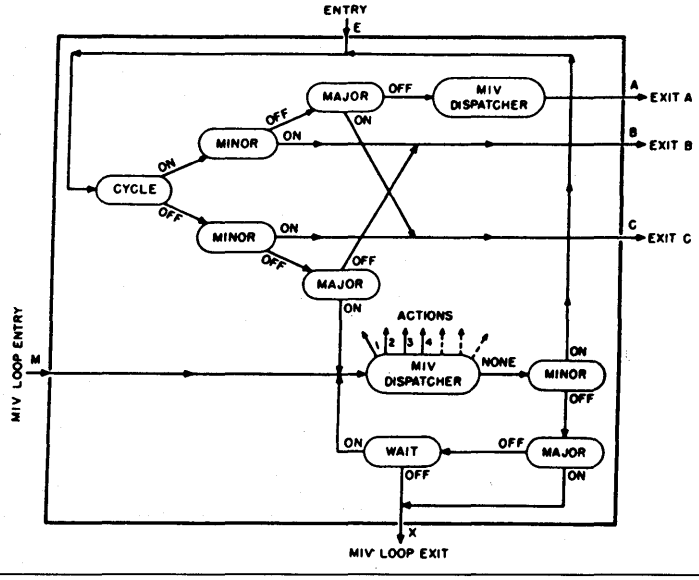


FIGURE 16
Definition of MIV Box re-entrant subroutine of Servo Lab Utility Routine Program (SLURP) simulated computer—basis for systematized MIV languages.

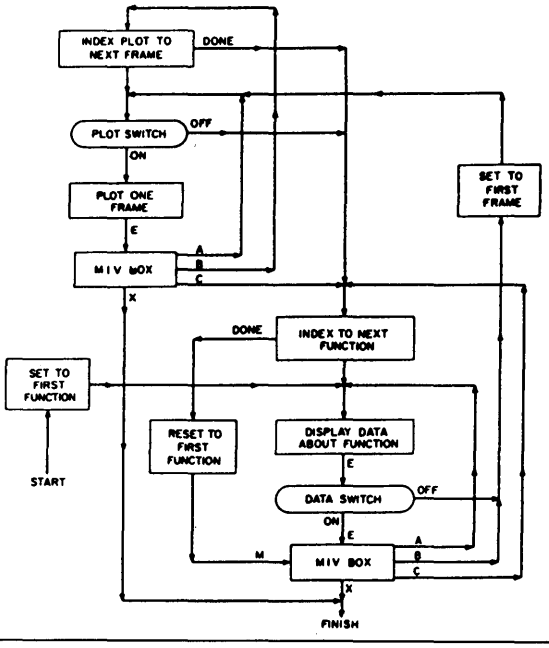


FIGURE 17
Hierarchic control of complex program action with generic MIV Box controls. Precursor to current mouse/menu control scheme.

could similarly specify console display only or off-line camera, as well, and the SLURP system would efficiently see to the details.

Conclusion

I hope that I have succeeded in defending my thesis that there were indeed some significant PWS "firsts in the fifties," even though only a tiny fraction of the computer resources that today are thought to be essential were then available. The revolutionary evolution since those early days has indeed brought much greater sophistication and broader capabilities, including the all-important hardware, software, and systematized approach improvements that allow ubiquitous spread to all types of users. But don't sell the old days short. Big ideas can come in small packages. Maybe that's an idea we've lost track of in today's technology where we *shrink* things mechanically to make them smaller. Maybe we need a sharper return to those earlier days when the only recourse was to *think* big things into their distilled essence—and make them work.

ACKNOWLEDGMENTS

Thanks to the MITRE Corporation Archives and the MIT Museum for the Whirlwind photos. Thanks also to MIT and my many colleagues for making it all possible.

REFERENCES

American Ordnance Association (1955) April 14–15. *Symposium on Proving Ground Instrumentation Problems,* Air Force Missile Test Center, Patrick Air Force Base, Cocoa, Fla. (program and papers).

Brackett, C. F. (1959) February 2. *The Direct Flexowriter Keyboard Input System at Whirlwind I (Barta).* Lexington, Mass.: Division 2—Lincoln Lab. Interoffice Memo (no number), 3 pp.

Charactron Project (1955). *Charactron Display Console Model 70B* (January 24, 1955, 14 pp.); *Charactron Computer Readout System, Model 100* (June 3, 1954, 22 pp.); *The Charactron Rapromatic* (no date, 11 pp.). San Diego, Calif. Convair division of General Dynamics Corp.

Cohen, Arnold (1985) October 4. Oral comments to D. T. Ross at Annual Meeting of the Charles Babbage Foundation.

Convair (1954) August 9. *Agenda for Flight Test Program Meeting* see also Ross (1954).

EJCC (1955) November 7–9. *Program of the Eastern Joint Computer Conference and Exhibition,* Statler Hotel, Boston, 27 pp.

Electronic Design Magazine (1956) May 1. New York, N.Y.: Hayden Publishing Co. *Computer Developments: Design Trends from Meetings,* pp. 36 & 37.

ERA (1954a). *ERA 1103 General Purpose Computer System* (brochure). New York, N.Y.: Engineering Research Associates Division of Remington Rand, Inc., 4 pp.

ERA (1954b) January 5. *ERA 1103 Installation Requirements.* St. Paul, Minn.: Engineering Research Associates Division of Remington Rand, Inc., 16 pp.

Forrester, J. W. (1948) July 29. *Whirlwind High Speed Computing.* Cambridge, Mass.: MIT Servo Lab. Rep. No. R-142. Appendix A in Redmond and Smith (1980), pp. 225–236.

Hamilton, D. A. (1955) August 4. *Conventions for Presentation of Results from Whirlwind I Polynomial Fit Program.* Cambridge, Mass.: MIT Servo Lab. Memo 7138-M-116, 4 pp. See also Ross (1954) and Ross, Thompson, and Cundiff (1956).

Holmes, L. L. (1956) April 3. *Flexowriter Input to WWI.* Lexington, Mass.: Division 6—Lincoln Lab. interoffice memo, 2 pp.

McCarthy, J. (1959) February 27. *Invitation to Conference on Symbol Manipulation Systems, April 16–18, MIT.*

MIT Digital Computer Lab. (1956) March 1 and 2. *Seminar on Microprogramming* (Announcement and agenda). Cambridge, Mass. 2 pp. plus map. June 6, 1956: Partial transcript of D. T. Ross presentation pp. 152–161, via F. W. Helwig.

MIT Servo Lab. (1952–1958). Certain Lists of laboratory personnel.

MIT Servo Lab. (1955) March 8, 9. Symposium on *Design and Evaluation of Bomber Fire-Control Systems.*

MIT Servo Lab. (1958) December. *Data Reduction Programming for Pre-B-58 Tests of the XMD-7 Fire-Control System* (3 Volumes). Cambridge, Mass.: MIT Servo Lab. Reports: Ross, D. T. Functional Description of the Data Reduction System (7886-R-1), 44 pp; Scheff, B. H. Initial Data Processing (7886-R-2), 149 pp.; Ross, D. T., and McAvinn, D. F. Evaluation of Fire-Control System Accuracy, (7886-R-3), 177 pp.

Nielsen, K. L., and Heyda, J. F. (1951) July 20. *Mathematical Theory of Airborne Fire Control.* Washington, D.C.: U.S. Government Printing Office, NAVORD Report 1493, 208 pp.

Price, G. R. (1956) November. How to Speed Up Invention. New York, N.Y.: *Fortune* magazine, p. 150 ff.

Price, G. R. (1957) February. *Technical Details Concerning the Design Machine.* Kingston, N.Y.: personal memo, 72 pp.

Project Whirlwind (1952–1957) Quarterly Summary Reports No. 31–No. 50. Cambridge, Mass.: MIT Digital Computer Lab.

Redmond, K. C. and Smith, T. M. (1980). *Project Whirlwind: The History of a Pioneer Computer.* Bedford, Mass.: Digital Press, 280 pp. Index.

Ross, D. T. (1952) July. *Autocorrelation Program* working papers.

Ross, D. T. (1953a). First *Evaluation Program Flow Diagram.* Computation Notebook pp. 26 & 27.

Ross, D. T. (1953b) June 27. *An Introduction to the Use of Air-Mass Ballistic Tables in Airborne Fire-Control System Evaluation* (CONFIDENTIAL, Title Unclassified—declassified November 1958). Cambridge, Mass.: MIT Servo Lab. Rep. No. 6506-ER-46, 25 pp.

Ross, D. T. (1953c) November 24. *A Mistake Diagnosis Routine for Whirlwind I Programs.* Cambridge, Mass.: MIT Servo Lab. Rep. No. 7138-R-1, 26 pp.

Ross, D. T. (1954a) May 24. *Missile Launch Simulator* program for Memory Test Computer working papers.

Ross, D. T. (1954b) June 24. *Improved Computational Techniques for Fourier Transformation.* (M.S. thesis). Cambridge, Mass.: MIT Servo Lab. Rep. No. 7138-R-5, 84 pp.

Ross, D. T. (1954c) July 16. Letter to W. J. Moe of Engineering Research Associates (ERA) regarding 1953 Polynomial Fit Program, 2 pp. See also Hamilton (1955).

Ross, D. T. (1954d) August. *Scope Input Program* (no original; revised copy fc Tape 126-50-505, 2/15/55).

Ross, D. T. (1955a) March 8. Handwritten instructions to DAH for ROSS DEMO. Director Tape, plus printout of same.

Ross, D. T. (1955b) June. *A Basic Input Translation Program for the ERA 1103* (program, runs, and working papers).

Ross, D. T. (1955c) July 25. *Special In-Out Equipment for Human Participation in High-Speed Computer Operation.* Cambridge, Mass.: MIT Servo Lab. Memo 7138-M-112, 8 pp.

Ross, D. T. (1955d) October 13. *[The] Gestalt [System of] Programming—A New Concept in Automatic Programming,* handwritten draft, 15 pp.

Ross, D. T. (1955e) November–December. Handwritten draft pages.

Ross, D. T. (1955f) November 17. *Gestalt Programming: A New Concept in Automatic Programming* (second draft, typed) 12 pp. plus addendum: "Proposed Changes for Final Draft," 2 pp.

Ross, D. T. (1955g) December 14. *Proposed Equipment Modifications for the Pre-B-58 Data Reduction Programs.* Cambridge, Mass.: MIT Servo Lab. memo 7138-M-128, 7 pp.

Ross, D. T. (1956a) January. Handwritten oral presentation drafts and notes.

Ross, D. T. (1956b) February 7. Gestalt Programming: A New Concept in Automatic Programming. New York: *Proceedings of the Western Joint Computer Conference.* AIEE for the JCC (now AFIPS), pp. 5–10. Also Cambridge, Mass.: MIT Servo Lab. Rep. No. 7138-TM-7, 14 pp.

Ross, D. T. (1956c) March 1. *Suggestions for Manual Intervention Facilities on the Air Force Armament Center ERA 1103 Computer.* Cambridge, Mass.: MIT Servo Lab. Memo 7138-M-144, 6 pp.

Ross, D. T. (1956d) March 14. *Flexowriter Keyboard Input to WWI, Preliminary Specifications.* Cambridge, Mass.: MIT Servo Lab. Memo 7138-M-148, 3 pp.

Ross, D. T. (1956e) April 5–Summer 1956. *Proposed Microprogramming Instruction for Whirlwind I Computer.* Working papers, drawings, and memo outline.

Ross, D. T. (1956f) April 30. *A Proposed Flexowriter Input for the Whirlwind I Computer.* Cambridge, Mass.: MIT Servo Lab. Memo 7138-M-160, 4 pp.

Ross, D. T. (1956g) September 26. *Servomechanisms Laboratory Requirements for Computing Facilities, 1956–1957.* Cambridge, Mass.: MIT Servo Lab. memo CA-M-1, 4 pp.

Ross, D. T. (1956h) November 14. *Programming Progress for Mission Director Task.* Cambridge, Mass.: MIT Servo Lab. memo 7138-M-200, 8 pp.

Ross, D. T. (1956i) November 23. *Whirlwind Versus 704 for Servo Lab Problems.* Cambridge, Mass.: MIT Servo Lab. memo CA-M-2, 5 pp.

Ross, D. T. (1958) April. A Multi-Mode Control Element; A Philosophy of Problem Solving; The SLURP System for Experimental Programming. Cambridge, Mass.: *Research in Defense Techniques for Airborne Weapons: 1957 Annual Report, Volume 2.* MIT Servo Lab. Rep. No. 7668-R-5(2), pp. 3–81 through 3-163.

Ross, D. T. (1959) May 13. *Visit to Lincoln Lab., Tuesday, May 19.* Cambridge, Mass.: MIT Servo Lab. memo (no number).

Ross, D. T. (1961) March. A Generalized Technique for Symbol Manipulation and Numerical Calculation. *Communications of the ACM* 4 (3): 147–150.

Ross, D. T. (1977). Origins of the APT Language for Automatically Programmed Tools. *History of Programming Languages* (R. L. Wexelblat, Ed.) (1981) New York: Academic Press, pp. 279–367.

Ross, D. T., Thompson, D. A. (Hamilton), and Cundiff, T. (1956) December 17. *Polynomial Fit Program.* Cambridge, Mass.: MIT Servo Lab. Memo 7138-M-202, 18 pp. See also Hamilton (1955).

Ross, D. T., and Ward, J. E. (1956) May. *Investigations in Computer-Aided Design for Numerically Controlled Production* (Final Report for 1 December 1959 to 3 May 1967). Cambridge, Mass.: MIT Electronic Systems Lab. for Air Force Materials Lab., WPAFB, Ohio. Rep. No. AFML-TR-68-206. MIT Rep. No. ESL-FR-351, 229 pp.

Ross, D. T., and Ward, J. E. (1957) October 22. *Charactron Plot Program for Pre-B-58 Data Reduction.* Cambridge, Mass.: MIT Servo Lab. memo 7668-M-251, 4 pp.

Ross, D. T., and Ward, J. E. (1961) January. Picture and Pushbutton Languages (Chapter VIII), and Ward, J. E., A Manual Intervention Facility (Chapter IX), in *Investigations in Computer-Aided Design,* Interim Report for December, 1959 to May 30, 1960. Cambridge, Mass.: MIT Electronic Systems Lab. Rep. No. 8436-IR-1, 156 pp.

Ward, J. E. (1954) July 23. *Disposal of Mechanical Correlation Computer (MCC).* Cambridge, Mass.: MIT Servo Lab. Memo 7138-M-49, 2 pp.

Ward, J. E. (1955) May 2. *Symposium on Data Reduction* (Announcement). MIT Servolab (no number).

Ward, J. E. (1958) January 27. *Manual Intervention.* (Memo to D. T. Ross.) Cambridge, Mass.: MIT Servo Lab. (no number).

Ward, J. E. (1960) January 15. *Automatic Programming of Numerically Controlled Machine Tools* (Final Report). Cambridge, Mass.: MIT Electronic Systems Lab. Rep. No. 6873-FR-3, for June 26, 1956 to November 30, 1959. Reprints 6873-FR-1 (July 30, 1952) for July 1949 to July 1954, [sic] and reprints 6873-FR-2 (March 15, 1956) for July 1949 to March 15, 1956.

Ward, J. E. (1970) June. *Computer-Aided Design for Numerically Controlled Production* (Final Report for 1 May 1967 to 30 January 1979). Cambridge, Mass.: MIT Electronic Systems Lab. for Air Force Materials Lab., WPAFB, Ohio. Rep. No. AFML-TR-70-78. MIT Rep. No. ESL-FR-420, 121 pp.

Ward, J. E., and Ross, D. T. (1955) July 29. *Minutes of M.I.T. Data Reduction Symposium: June 1 and 2, 1955.* Cambridge, Mass.: MIT Servo Lab Rep. No. 7138-S-1, 20 pp.

Ward, J. E., and Ross, D. T. (1956) March 1. *Suggestions for Manual Intervention Facilities on the Air Force Armament Center ERA 1103 Computer.* Cambridge, Mass.: MIT Servo Lab. Memo 7138-M-144A, 30+ pp.

Ward, J. E., and Ross, D. T. (1956) May 8. Draft paper of Revision of (Ross 1956c), incorporating MIT Digital Computer Lab. memo and drawings. Cambridge, Mass.: MIT Servo Lab. memo 7138-M-144A.

WJCC (1956) February 7–9. *Program of the Western Joint Computer Conference and Exhibit.* Fairmont Hotel, San Francisco, 23 pp.

Wolf, W. M. (1954) September–November. Worksheets and photos for Large Letter program. Flow diagram and WWI code for Manual Intervention tests.

Participants Discussion

Severo Ornstein What I get out of all of this is the feeling that in order to understand fully the thinking that went on you actually had to be there, and you had to be familiar with the kind of equipment that people had at the time. One K of memory and 20 thousand operations per second were really awesome in those days. When I was working on the communications between subsectors in the SAGE system, we had 1 kilobit lines interconnecting them. It seemed like just an enormous amount of communications capacity at that point.

I have two questions. We were standing talking to Allen Newell in the break here a few minutes ago, and we were wondering what it is that makes new ideas come forth. I think we all are really curious about that. We were talking about the connection between existing hardware, technology, and new insights, and you made some comments that I think others here would find interesting. I'd like to ask you to repeat them. My second question is as follows: Of the array of novel ideas presented here, which do you think are the three most influential ones? I heard you mention a number of things that were new at the time and I wonder what you think are the three most original ones, or most important ones.

Ross I'm really not very programmable, so I'm not sure how I can reproduce what we were talking about with Allen. But the idea that I have about where creative insights come from is that they come from somebody who actually is there, in an environment that's built in terms of whatever is that day's technology. What you do is that that individual penetrates into the technology and has his ideas reflected back to see what is really implied by what's there. I don't think that there are very many things that are really dreamed up completely out of the blue. Some artists do, maybe, but I think that's more that they have a slightly demented mind and are doing just the same thing with a warped set of filters. You need to have a reality in which you are a participant. What creates the innovation is this deeper insight, literally seeing into, but reflected back to you—what is your current total environment. It's always not what's on the surface. There's inherently a built-in level of abstraction that has to go along with it. This is why you do get ideas that are portable—they carry around and can apply to many things. This comes across when you formalize it, when you talk about it: Wow, it's a brilliant idea, it's a great generalization. But it really is just this

little-bit-deeper version of the same kind of seeing and interacting and participating that we do all the time for everything else.

As for the top three novel ideas that I would pick from what I was talking about, I think I'd rather leave that to you all or later historians because, again, I've never been able to track down who actually saw or was influenced by my activities. All I know is, I was there, I did them, I remember them vaguely, I found them in my materials, and I've documented them here for you. How they then permeated and influenced other people, I don't know. I do know that all during the years of my having very creative people working with me, there was mutual stimulation—this thing of "it's the environment you're in, the problems you're given, the challenges." We always had to have problems that needed solving yesterday. We had to have real people asking for real solutions in problems that were beyond the state of the art, so we had to do that penetrating and getting further. When you get a creative bunch of people in such an environment, and get the right general way of doing it, you come out with a Whirlwind. You come out with Wes Clark's paper (I just loved to read that last night). You'll just love him; he does a much better job of showing what it was like to be there and to have the people involved.

I think that the main thing is that I was aware, because of this insightful look, from the very beginning after I first walked into room 222 and had that: "wow, whee, look at all those green scopes, and red lights." If they only had laser light shows. That was the version in those days; it really turned me on. I knew that's what I had to have, because I already had building up in me this vast set of needs for what I thought was a very, very complicated set of data reduction programs to do on that little bit of memory.

Sig Treu

I wonder whether you could characterize to what extent the history of personal workstations is necessarily overlapping with the history of what one might consider to be computing or computers in general. Or, if I may rephrase the question, Can you identify criteria by which you might separate out those historical aspects that are peculiar to personal workstations? One might argue that some of the things that have been presented this morning could just as well have been included in the history of computer hardware, computer software, and so on.

Ross

I think that probably goes along with the comment that you must have that reality in which you're participating to have any of these things happen. Of course, one of the earliest personal workstations is the cobbler's workbench—you sit there and all the things are right there to work with. Again, it's the technology and the problems, the needs of that day, that make that workstation. If you follow through what Gor-

don Bell was saying, the key thing in this historic development period is the stages of how you bring the cost down so that you can get the power up to match a broader and broader class of individuals, in terms of their problem needs and in terms of their economic needs.

Again, the point that John Brackett was making in the comment on Gordon's talk and the line that Severo was bringing out, is the main one that I'd like to stress: How do you bring those together? How do you bring the PW (personal work) and the WS (work station) together with an architecture and a systematic treatment? It is more than just how to put variables, instead of programs, directly on line; it is more than just having a gestalt language idea with switches and so forth; and it is more than having just a SLURP system that has the software framework and the building blocks. It is also more than just having software engineering tool kits and languages, more than having modular software building blocks, the OCCAM system that takes it right out to the hardware building blocks that you hook together (which is another one of Wes Clark's old pet peeves). These are just stages of development. What they do is reform this reality with which we interact at each stage. As we understand more about it, you find that the man and the machine do in fact become more and more coupled, and their view of what problems are and how to go about solving them becomes more and more sophisticated, more and more coupled. I think it's really exciting stuff to do.