DAVE TUTTLE'S MEMOIRS

Melinda's Preface

Several years ago, when I was doing the research for an earlier paper, Romney White advised me to talk to Dave Tuttle, who had been one of the developers for VM/370 Release 1 and who had, like so many others, left VM (and IBM) when VM Development was moved from Burlington to Poughkeepsie. Like all the other advice Romney has ever given me, that was very sound. I quickly came to value Dave as an historian's dream source, for he combines an astonishing memory with warm and witty perceptions.

Over the past few months, as I've bombarded Dave with drafts of this paper, I've had the good fortune to be the recipient of several long notes containing reminiscences of his days working on VM. Having enjoyed his letters so much, I asked Dave to allow me to include them in this paper for others to enjoy.

—MWV

Dave's Preface

The time I was at IBM and involved with VM/370 was an important, and fairly intense, portion of my life and career. You may have, however, turned loose more than you counted on—by expressing an interest and being a good listener. Here are some of my recollections of "the early days", from the point of view of a young participant.[170]

(In writing up these events, I am not trying to emphasize my own part in things; everybody was a hero according to normal IBM standards. The things that I remember most vividly are the ones that I was involved in directly. I have found some curious discrepancies in what I remember. I know quite well the general sequence of events in several different "threads" of activity, but I'm not at all sure that I can relate them to actual dates or, in some cases, to each other.)

—Dave Tuttle

CP-67/CMS and CSC—1968 to 1971

Ed Hendricks and I met during 1967 while he was working in the M.I.T. Computation Center in a post-grad program and I was there as a part-time user consultant and system programmer. Through that contact, I managed to get a part-time position at the Cambridge Scientific Center in the fall of 1968. I started on October 18, 1968, one week before my 20th birthday. Not only was it an exciting technical environment, it was also the middle of the rising tide of social and political awareness in Cambridge and Boston.

My experience at the Cambridge Scientific Center started just after the first successes of CP-67/CMS. Bob Creasy had recently transferred to the Palo Alto Scientific Center, Les Comeau was off somewhere else within IBM, Dick Bayles had left IBM to start up National CSS in Connecticut—a time-sharing service bureau based partially on a modified CP-67/CMS—and a lot of the CSC people were working on using CP-67/CMS, rather than merely trying to make it work. The modified S/360-40 was no longer around; its place on the 4th floor had been taken by an IBM 1130/2250 Mod 4 system, and a 512K S/360-67, in one comer of the third floor, served as the main system for the Center.

Organization Notes

Although my memory is a little uncertain, I think there were three main groups at CSC. Craig Johnson ran the Graphics group, comprised of Ed Hendricks, myself, and several others whose names have disappeared into the mists. Dr. Tim Johnson of M.I.T. was working with us, as was Bob Seawright on a part-time basis. The CP-67/CMS system work was under Rip Parmelee, I think, and included Bob Adair, Dick Meyer, Harit Nanavati (one of the later founders [?] of Interactive Data Corp.), and the others which you have listed. The third group was known as Operations Research, run by Marty Schatzoff and including Don Hatfield, Stu Greenberg, John Ravin, and several more.

In addition to the CSC people, 545 Technology Square also housed the IBM Boston Programming Center, on the third floor, where the CPS/360 system was developed and maintained. Among their top people were Nat Rochester and Jean Sammett. The BPC reported through a slightly different chain of command. The five U.S. Scientific Centers (Palo Alto, Houston, Washington D.C., Philadelphia, and Cambridge) all reported into IBM Data Processing Division Headquarters (DPD-HQ) in White Plains. The BPC group, the Time-Life Development Center (New York), and a few others scattered around the country reported into DPD Industry Marketing and Development (DPD-IM&D). IBM Yorktown Research was an entirely separate organization, essentially a division unto itself, while all of the “mainstream” system development work was the province of the System Development Division (SDD) in Poughkeepsie, New York.

Technical Matters

The first project Ed and I worked on was the development of a BSC communications package to exchange data between OS/360-PCP, in a virtual machine of CP-67/CMS, and an IBM 1130. The OS/360 side of the connection was running an associative database for graphics data; the 1130 end had a multi-tasking monitor driving an IBM 2250 Mod 4 display, a Sylvania Data Tablet, and Sketchpad III, a 3-D drawing and display program developed primarily by Dr. Tim Johnson of M.I.T. as an extension of the original Evans & Sutherland Sketchpad concepts.

We had a few difficulties to deal with. OS/360-PCP was not a multi-tasking system, so Ed had to develop a user-level round-robin “scheduler” to support concurrent operation of the database code and the communications code. [The techniques he used were perhaps based on some earlier work he had done at M.I.T. in 1967—*i.e.,* bringing up a version of the SpaceWar game on the Computer Center’s S/360-65, which had an IBM 2250 Model 1 (channel attached) vector display.] Secondly, BSC communications were, at that time, far from being well supported or understood. The first attempts at using the OS/360 BTAM support failed because the system macros wouldn’t build the correct DCB formats; we ended up using EXCP and writing our own channel control programs. On more than one occasion we had to take out the schematics and logic manuals for the 2701, an oscilloscope, and a data-line strip recorder to find out what was

going on. We invented a protocol of our own to handle pseudo-duplex exchange of multiple unrelated data streams, with what may have been the first use of selective acknowledgements piggy-backed on data blocks.[171]

On the other end of the wire we also had a round-robin multi-tasking monitor in the 1130. The graphics display used the main memory of the 1130 for its display buffer, and there was a large collection of interrupt and command response routines which serviced light pen tracking, function keys, the display keyboard, the system keyboard and printer, a line printer, the comms line, the disks, etc. There was a group of a half dozen or so working on the 1130 code and four or five working on the OS/360 code. I ended up doing the comms code on both ends, some work on the PL/I database code, and several of the device service routines on the 1130. We eventually did get it all working, sometime towards the end of summer, 1969.[172]

The work that we did at that time seems to have had a lot of longer-term impact. The 1130 code and the OS/360 virtual machine code were made available to Brown University in a joint development agreement, and they became the substructure of the Brown HyperText system. The OS/PCP system in a virtual machine was also the basis for ONLINE/OS, a single-user interactive monitor much like CMS, except that it had the full OS/360 data management capability and online support for all of the OS compilers and utilities. It was never released outside of IBM, but we did sneak it into the Spring Joint Computer Conference in Boston in 1969 as "SJCC/DEMO", a name which conveniently had exactly the same number of characters.[173]

Similarly, the multi-tasking and multi-thread communications experience, and a version of the 1130 code which turned it into a remote spooling "workstation", led fairly directly to the original CPREMOTE, which used the BSC protocol we had developed and an Ed Hendricks' special 4K standalone monitor. The CP-67 modifications which allowed CPREMOTE to read spool file blocks *via* DIAGNOSE were also developed somewhere along the way.

---

[171] E.C. Hendricks and D.B. Tuttle, *Notes on Design Objectives and Implementation under OSI360 of a General Purpose Binary Synchronous Telecommunications Package for Multi-Programmed Applications in OS/360,* IBM Cambridge Scientific Center Report 320-2047, August, 1969.

[172] E.C. Hendricks and D.B. Tuttle, *HOTLINE: A Binary Synchronous Access Method,* IBM Technical Disclosure Bulletin WA8-70-0091, September, 1970. Tuttle, D.B., *BSCCA: An Interrupt Service Subroutine for Binary Synchronous Operation of the IBM 1130 Synchronous Communications Adapter,* IBM Cambridge Scientific Center Report ZZ20-2096, October, 1969.

[173] E.C. Hendricks, C.I. Johnson, R.D. Seawright, and D.B. Tuttle, *Introduction to ONLINE/OS* and *ONLINE/OS User's Guide,* IBM Cambridge Scientific Center Reports 320-2036, 320-2037, March, 1969.

"Environmental" Factors

The "great unbundling", in June of 1969, constituted a real threat to the Scientific Centers and the IM&D groups. The new way of doing business created a lot of confusion for the IBM organizations which were not a part of the mainstream System Development Division (SDD), because it was no longer clear how non-product software could be made available outside of IBM. The first releases of CP-67/CMS were by way of the Type III Library. The release of CP-67 Version 2 almost never happened; the submittal missed the deadline for freezing the Type III library, prior to the June 23rd announcement, and it took a lot of high-level arguing to actually get it accepted.

Following the June 23 unbundling announcement, the CP-67/CMS Development Group, at that time under Dick Meyer, split off from the Cambridge Scientific Center (almost literally; they took over some office space on the fourth floor which had been part of CSC, and put up a wall between the two areas. The door was seldom locked, but IBM required the separation.) Dick Meyer had taken over management of the group when Rip Parmelee left for an 18-month assignment at the Grenoble Scientific Center, around the time of the SJCC. CSC stayed in the DPD-HQ chain, while the CP-67/CMS group (11 people, at the time) switched over to DPD-IM&D. I stayed with the CSC Graphics group, at least for the moment

More Technical Matters

In the fall of 1969 I started working on a new editor. People generally knew that the CMS editor was somewhat limited and clumsy to use, and we wanted an editor that could also be ported to ONLINE/OS. The eventual result, in March, 1970, was called Ned ("New EDitor", of course!), and it quickly became very popular as a replacement editor for CP-67/CMS within IBM. Some of the important new features were an imbedded macro language and macro processing, external descriptor files for filetype-dependent defaults, generic "target" capability for a wide range of editing commands, a CHANGE command with support for compound, repetitive, and recursive changes, and the ability to edit files which were up to 10 times as large as available virtual memory. What it did not have was a screen interface (the 3270 was announced in 1972, I think, and 2260s were never very common) or the ability to generate UPDATE files; otherwise you might recognize it as the immediate father of Xedit[174]

The popularity of Ned within IBM was the cause of my first experience with SHARE or GUIDE meetings. I was invited to SHARE XXXV in Montreal in the fall of 1970, as the result of a behind-the-scenes nudge by Nat Rochester, of IBM's ATS/360 group. I drove up to Montreal in my orange-and-black, used-to-be-a-Public-Works truck, named "Truck", of course, only to discover that I would be staying in the brand new Hotel Bonaventure—costumed Oriental bell-boys, lobby on the tenth floor, roof gardens, and all. I survived that first meeting week, and it turned out to be only the first of my many SHARE meetings and, later, GUIDE meetings.

My first GUIDE meeting was, coincidentally, GUIDE XXXV, also in Montreal. I was somewhat unprepared for the difference in style between SHARE and GUIDE—the VM/370 group had to buy me a necktie so that I could talk at one of the DP Management sessions; I hadn't brought one along! (Many years later, after a lapse of eight years or more, I attended a GUIDE meeting in Los Angeles as a representative for GTE Telenet. Within 15 minutes of appearing at one of the

[174] D.B. Tuttle, *Context Editors, Part II: EDIT II—A Non-System-Specific Context Editor,* IBM Cambridge Scientific Center Report 320-2048, March, 1970.

VM/370 sessions, I had been recognized, given a list of all of the former IBM VM people who were attending the meeting, and reminded that the group still had the Dave Tuttle memorial necktie!)

On another occasion there was a GUIDE meeting in Boston, which gave more of the VM/370 crew a chance to listen to and talk with the user community. As one of the regular GUIDE attendees, I didn't get to go to many of the sessions, but I was expected to show up at least once or twice at SCIDS. I caused something of a sensation one night when I walked into SCIDS in a jacket and tie—and a full Scottish kilt, knee socks, sporran, and my IBM badge. I'm not very much of a Scot, but there is supposedly a sixteenth or so of Royal Stewart in my ancestry, and I didn't have a clean suit that fit.

The Early Days of VM/370 Release 1

I don't know from my own experience exactly when the planning began for VM/370. Late in 1970 I transferred from the CSC Graphics group to the Systems group, still in the Scientific Center, around the time that Rip Parmelee returned from his assignment in France. The CSC Systems group was working on the series of CP-67/CMS modifications which would eventually support a real S/370. The sequence, as I remember it, was approximately as follows:

1. CP-67 modified to provide a virtual S/360-67 (3rd quarter 1969?). This system became the production system at CSC to support further experimentation. At one point in early 1970 or 1971, IBM Yorktown Research ran some tests to verify the successful "virtualization" of the full S/360-67 function. They ran out of terminals and time when they were loading CMS in a virtual machine 17 levels deep! (CP under CP, under CP,...—1 real CP, 15 virtual CPs).

2. Modified again, to provide a limited S/360-67 Multi-Processor in a pair of virtual machines (late 1970; I was involved). This system supported development of real MP support for CP-67, which was brought up on the CSC machine pair sometime in 1971.

3. Modified separately, to provide a virtual machine with simulated S/370 privileged instructions, control registers, and one version of the S/370 dynamic address translation (DAT) architecture (4K pages, IM segments, as supported by the S/360-67 hardware). CP-67/CMS never supported 2K pages in a meaningful fashion, because the S/360-67 did not have the necessary hardware support.

4. Modified again, to support operation in a S/370 virtual machine (little if any machine check recovery, no support for S/370-specific DASD, still 4K pages, IM segments). This was ready about the middle of 1971, before we had a real S/370 to play with. Almost all of the work was done using the IBM Confidential "Red Book" architecture documents.

I joined the CP Development Group on August 1, 1971, in exchange for Lynn Wheeler. He took my slot at CSC, I took his in the CP group. The first work I did was to spend two frantic weeks writing "Alpha Test" functional and logic documentation for Ned, which was intended to be the CMS Editor for the first release. The first approval stage of the VM/370 project was to get a conditional go-ahead for the planning and design efforts; that was done in late Spring, 1971. The next hurdle was to prepare a complete set of design and function specifications, to standards set by SDD, and get them approved for implementation. When I joined the group, we were almost ready for the first submittal of the document set.

The first submittal failed, almost predictably. It took a total of, I think, four tries to secure the Alpha Test approval. One incident that was not pleasant involved my editor, Ned. It was one of the first CMS components to be approved (second submittal, I think), but a decision was made between the third and fourth submittals to pull it back, substituting a slightly updated version of the existing CP-67/CMS editor. Ned was judged to be "too sophisticated for the average user", and it was substantially more code—potentially a problem in small systems. To make matters worse, I was not told by local management that the decision had been made—I heard it from one of my friends at Yorktown Research, almost accidentally, after the final Alpha Test approval.

The Alpha Test process was a series of negotiations both of function and of scale. The original plan was to completely rewrite both CP and CMS, with a lot of new functions and improvements in both components. The plan which finally was approved involved a much reduced CMS effort, essentially a "port" of CMS to the S/370 environment, rather than a complete restructuring. The CP plans came through more or less intact, except for the planned multi-path I/O and later multi-processor support. We knew that we were working toward a deadline of the "big splash" Advanced Function announcement but nobody knew yet exactly when that would be.

After the first few weeks of documentation, working with Bob Downs, by the way, I was put to work with Paul Tardif on further modifications to the CP-67 "I" system—the version which actually ran on S/370 hardware. Some time in the late summer of 1971 a 512K (nominal) S/370-145 was installed in our machine room on the third floor. For security reasons, it was moved in the wee hours of one morning all in one piece, by crane, through one of the windows which had been removed temporarily. "Everybody" knew that the 545 Tech Square IBM groups worked only on virtual storage systems, so it would have been a pre-announcement of the S/370 capability if anyone knew we had the machine.

Some of the problems we had to deal with in the early development were interesting. Every other month or so, we would get a new version of the loadable microcode for the 145, and more often than not there would be some impact on the privileged instruction set There were three generations of slightly different DAT control instructions, and one instruction which had two different opcode assignments. One of the things that Paul and I had to deal with was support for the new DASD devices, the 3330 and the 2305, and the new Block Multiplexor channels.

The first step was to modify the CP-67 I-System to support the 3330 and 2305 as devices attached to a virtual machine. That provided an environment for us to develop an I-System version that would support them as residence and paging devices. It was in that step that Dick Newson and I almost lost everything, early one morning... The system which supported attached 3330 and 2305 devices was not particularly stable, and the 2314s which we were using as production drives were not very fast Consequently, we were in a bit of a hurry to move over to the 3330s. Also, the nominal 512K memory of the S/370-145 was actually quite a bit less; the loadable microcode support for DAT, the advanced timers, PER, etc., used about 20,000 bytes of main memory in addition to the regular control storage. The nominal 524,288 bytes of memory was reduced to about 504,000 bytes—not quite enough to load the CP nucleus from tape or cards.

I've forgotten the exact sequence which got us into trouble, but Dick and I spent about half an hour in mild panic around 7:30 in the morning. We knew that people were going to be arriving soon, but we had no running system and no way to load a new nucleus onto any of the disks. We escaped finally because I was able to IPL the system in Instruction Step mode, make some in-storage patches from the console, then bring up one of the unstable systems just long enough to rebuild a CP nucleus of the "production" I-System. Only the two of us knew how close we had come to a major setback!

The CP development team, reporting to Dick Newson, was an interesting mixture of people—one of the things which undoubtedly was a factor in the success of the implementation. We were a mixture of hot-shot "kids", former academic/research types, and long-service IBM Field Engineers. The detailed implementation approach was based on the new control block structure which Dick Newson (and possibly others—I'm not sure) had designed for CP, along with a set of register usage conventions, command scanning routines, and module linkage macros which Dick Newson and Carl Young had developed. Each of us took some pieces of the old CP-67 "I" code and either recoded it to the new control blocks, register usage, and module naming conventions or, where the S/370 architecture encouraged it, redesigned the function entirely. The people and pieces that I remember were something like the following:

- Dick Newson - DMKSCN, DMKRIO, control blocks, high-level design
- Carl Young - DMKDSP, DMKSCH, DMKPAG, DMKPTR, DMKIOS
- John Seymour - DMKFRE, DMKCCW, DMKPTR, DMKPRG, DMKPSA, DMKTRC, DMKTRA
- Ed Murray - DMKCKP, DMKVSP, all unit-record spooling
- Charlie Weagle and Ray Grein - DMKMCH, DMKCCH, model-specific support, DMKFMT, DMKLDR, DMKDIR
- Clyde Wildes - DMKQCN, DMKCNS, DMKVCN, terminal control commands
- Larry Estelle - DMKVIO, DMKVDB, DMKLNK, minidisk I/O, disk I/O recovery
- Dave Tuttle - DMKPRV, DMKVAT, DMKHVC, DMKHVD, DMKDIA, DMKDRD, DMKPER, DMKDEF, VMFxxxx utilities

Undoubtedly I have forgotten some of the details, but I'm pretty sure that there were just the nine of us in the immediate management group. There was some trading of modules back and forth as we went along. I did some work in DMKPRG to support the interfaces to DMKPRV, DMKVAT, and, later, to DMKPER. I also did the Attach, Detach, Define, and Dial commands in their entirety, after Larry Estelle had done most of the basic minidisk management commands in DMKVDB. Over a period of time each of us worked in a lot of different areas of CP, but there was generally a recognized expert for each major sub-system or set of functions.

The separation of functions into modules was a fluid thing; our goal was to keep every module within range of a single base register—4096 bytes—even within the resident portion of the nucleus. Modules in the non-resident portion had to be kept smaller than about 3700 bytes—one 4K page minus the 10% margin required by SDD standards. For example, when Dick Newson gave me the task of rewriting the privileged instruction simulation and virtual DAT functions, which had been all in one module in CP-67,1 went back to him a week or so later and told him that it would take at least three separate modules. As we incrementally added the planned functions, it grew into a cluster of five or six modules, some resident and some pageable.

For several months, from the arrival of the S/370-145 until February, 1972, everybody used the CP I-System as a production time-sharing system. CMS development, tech pubs, and the stand-alone utilities (LDR, DIR, FMT, etc.) could all proceed without much difficulty. It wasn't until sometime in February, however, that we had completed enough of the CP work to think that we might have a runnable system—-*i.e.,* able to support a CMS virtual machine and accomplish

some work in it As soon as the basic functions were in place and somewhat shaken down, we went into an interesting and intensive period of trying to use the system for everyday work, at the same time that we were working hard on filling out the CP capabilities.

From the middle of February until sometime in early June, we were on a schedule something like the following:

1. Using the production I-System, collect the latest group of fixes, new functions, and/or new modules, and build a VM-CP nucleus.

2. With 15 minutes notice to the users, shut down the I-System and bring everyone up on VM. One of the earliest functions running was the fast dump and restart logic.

3. After we had accumulated three to five CP dumps on tape, a period which ranged from as little as 45 minutes in the beginning to as much as three or four hours, take down VM and bring up the I-System again.

4. Print off the CP dumps. In the machine room (visible in the photo of Ed, Larry, and myself) was a work table which was the repository of the most current listings of each of the CP modules. Each day one or two of us had "triage" duty—scanning the dumps to roughly localize the problem, then passing it on to the person working on that area of the system. Carl Young and I frequently had the duty, because we each had a broad overview of the system.

The "Triage" Desk

Larry Estelle, Dave Tuttle & Ed Christiansen

5. When we had identified fixes for each of the dumps and reassembled the appropriate modules, usually within a few hours, we would update the listing rack and start over from step 1.

In February and March we had up to four different VM-CP systems every day, as we were shaking out some of the more obvious problems. It slowed down substantially as time went on, but the problems became not so easy to find nor quick to fix! Through it all, the CMS group and the Tech Pubs group put up with a lot a disruption, but we all ended up with a system we were very proud of.

Anecdotes

The story of Program Event Recording in VM/370 would probably interest you, because we had a chance to support it in Release 1.0 but had to pull it out at the last minute. One of my explicit roles in the CP group was to be the explorer of new devices and new features, hence my earlier involvement with the 3330 and 2305 support As part of the DMKPRV work I had to support both the control registers used by the PER feature and the interrupts which would be generated by a virtual machine system which enabled PER. Along the way I also wrote a full-function version of DMKPER and a companion command module, modelled after the old standby TRACE command, which used the PER hardware to provide non-intrusive and selective instruction tracing, fetch/store storage traps, single-step execution, etc.

Unfortunately, debugging the TRAPPER command (abbreviated TRAP, of course) uncovered some subtle problems with the PER microcode in the Model 145. There was an interaction between Extended Control Mode, Dynamic Address Translation, and Program Event Recording which resulted in a failure to suppress PER interrupts when you went from EC-mode, DAT, PER to BC-mode, non-translate—*i.e.,* whenever you re-entered the CP nucleus! On a couple of occasions I observed an "impossible" console trace of program events in the CP handling of a virtual machine privileged instruction. To this day I don't know how the system continued to operate. When we tried to communicate with the microcode group (Kingston?), I learned the real meaning of "jargon". While I could describe the problem situation very clearly, I couldn't make any sense out of the questions I got in reply, so I couldn't help the microcode people enough to get a quick fix. This was sometime in the late spring of 1972 and there was a lot of schedule pressure, both on us and on the hardware development groups. As a result, the full TRAPPER support was replaced with a "stub" DMKPER module for the first release.

Another anecdote from that amazing year arose when we did our first testing on a S/370 Model 135, again in the spring of 1972. I don't remember exactly who was with me, but two of us gathered up a set of early VM/370 tapes and a disk pack or two with a pre-built system on it and travelled to the hardware development labs in Kingston. We arrived in the truly giant system assembly building there and were led on a long walk to a tiny room somewhere in the bowels of the building. In the room was the "C Test" 135 system that we were to test on. This was a machine that had not yet been announced, with some of IBM's latest technology in it—but it looked like it had been used hard for years! The paint had worn off around many of the console controls, the system was dirty, the rotary knobs were loose, and one of the console switches was broken. The hardware "C Test", for mechanical and electrical reliability, was apparently pretty rough!

Our first attempt at bringing up the VM system on the 135 met with very little success. After an hour and a half of poking around, we discovered that there were some serious interrupt problems with the advanced timers (CPU Timer and Clock Comparator, both new features in S/370 Advanced Function). In most of the S/370 models these timers were implemented at least partially in hardware. The smaller, less expensive Model 135 had implemented them entirely in microcode. They had tested extensively with the mainstream OS/VS1 and DOS/VS systems, but VM used the timers much differently—the other systems used it for time-slicing, while we used it

primarily for usage accounting. We were lucky; on hand for the test were two of the microcode authors from the IBM labs in England. While we took a break for lunch, they rewrote the timer microcode on the fly from the system console. After lunch we tried the system again, and it worked with no problems at all!

The saga continued several weeks later, when it came time to test the "official" version of the new microcode timer support. Once more, a couple of us travelled to Kingston for the test, but we had just a set of dump tapes with us and they had given us a different set of directions. Instead of the huge system assembly building, we had been directed to the parking lot of the IBM Kingston Recreation Center. After some driving around, we arrived to find a little shack on a concrete slab in the comer of a field—the kind of pre-fab building you would put up in your back yard for storage. Inside was a complete S/370-135 system, set up for environmental (noise, heat, etc.) testing, with not another building for close to half a mile in any direction. Trying to be professional about things, we dutifully sat down at the system, formatted a disk pack or two, and set about loading the tapes onto disk. As I was sitting at the console waiting, the building started to fill up with electrical smoke—one the disk drives had caught fire! I resisted the urge to use the "Emergency Power Off' switch, but it was hard—how many people have ever had a legitimate occasion to do it? I simply hit System Reset, then Power Off, as all of us quickly cleared the building.

After a few minutes with the windows and doors open, we were able to get back in and use the big pedestal fans to clear out the smoke. The hardware people cut the affected drive out of the 2344 group—it wasn't one of the drives we had been using—and we were able to finish loading the system. This time the VM/370 testing went without a hitch. As we packed up our material and headed out, we were treated to a scene right out of the "Twilight Zone". It was at the end of dusk, in the middle of essentially nowhere; a dark sky full of purple clouds and stars looked down on a tiny building with light pouring out of the doors and windows—with a full-scale computer system and hardware engineers working away, but not another soul as far as you could see!

More Tales of VM/370 Release 1

Many of the aspects of the early VM/370 development are best appreciated in retrospect. For the entire period we were working with very little margin for error. If we had any serious hardware failure, there were no backup systems closer than Poughkeepsie or Kingston, New York; we took the normal once or twice weekly disk backups, but a few days worth of work lost to a disk failure would have been a serious setback. There was some contingency margin in the schedules, but most of that "margin" could be realized only by reduced system testing or by reduced function at First Customer Ship (FCS). In the more careful software development environment of the 1980's, nobody in their right mind would accept the same level of risk

The lack of a backup system almost caught us when it came time to try running OS/VS1 under VM, in the late spring of 1971. The VS1 group was in Endicott at the time, I think, and they had been working on the virtual memory extensions to OS/MFT longer than we had been working on VM/370. Historically we had a somewhat better relationship with the Endicott people than we did with the Poughkeepsie OS/MVT-TSO-OS/VS2 people. The OS/MFT-VS1 people were "second class citizens", viewed as being squeezed out between the success of DOS/360 at the low end and the grandiose capability of MVT-TSO-VS2 for mid-sized and larger systems. The VM/370 people, not even in SDD, were generally viewed as at best a "dark horse" possibility. There had been some cooperation between the OS/MFT group and the CP-67/CMS group, along with some experimentation to improve the operation of MFT in a virtual machine—but I'm not at all sure when it began. Over a period of a couple of years we forged an alliance with the VS1

people against the TSO-VS2 "bandwagon". VM-CP and CMS provided an interactive programming environment to complement the OS/360-compatible batch capabilities of OS/VS1, as an alternative to the OS/VS2-TSO combination.

When we first had the opportunity to test OS/VS1 under VM/370, the Endicott group sent two of their experienced system people to Cambridge. Schedules were such that we had no more than one or two days to accomplish as much as we could, prior to a development freeze of some kind for VS1. Naturally, we had been having some machine trouble—the weather had been warming up fast, the machine room was in the south-east corner of the third floor, and we had had to shut down the system once or twice because of memory overheating. The day that the VS1 people arrived, we had to shut down the system mid-morning with the memory overheated, only to discover that two cooling fans were not running in one of the memory arrays. The S/370 systems were still quite new, and the IBM repair people could only find one replacement fan in the Boston area. They assured us that we could "probably run OK" with just one fan! Rather than just sit around while the CE went to pick up the spare part, I took the old fans over to workbench in the machine room and stripped one of them down to see if I could find the problem. The overheating had apparently congealed the bearing grease, so I cleaned it out, found a can of lithium grease in the CE cabinet, and repacked the bearings. By the time the CE came back from the main IBM office outside of Harvard Square, I had a roughly-rebuilt fan which we installed along with the one new one. The system held up well during an afternoon of successful testing, and the Endicott people went home with yet another tale of "those crazy people in Cambridge".

Personnel Notes

As I mentioned previously, the VM/370 group in the summer of 1971 and onward was an interesting mix of people. Somewhere in the assembly process we picked up most of the CPS/360 (CPSS?) people from the Boston Programming Center and a significant number of "refugees" from the New York Time-Life Development Center, which was closed down sometime in 1971 or 1972, I think. While the VM/370 development was proceeding in secret, the group was still responsible for supporting CP-67/CMS, CPS/360, and one or two semi-products which had been handled by the Time-Life group. The collective responsibility for several "orphaned" products, each with its own set of dedicated users, and the fact that the group was still in the DP Division were contributing factors to the pro-user/contra-SDD attitude which generally prevailed.

One area of contention we always had to deal with was IBM product standards. We were deliberately trying to produce a mainstream System Control Program (SCP), but we didn't always agree (!) with the standards established by SDD for such things as source control, command scanning and interpretation, error message numbering and coding, documentation library structure and content, etc. Every time we wanted to do things our own way, it required another round of negotiation or justification or, in one case, a threatened programmer revolt. The administrative and marketing people in the VM group deserve a lot of credit for succeeding as often as they did in dealing with the IBM "establishment".

The "programmer revolt" resulted from an SDD standard that would have required us to reject any command which was entered with too many arguments—even for commands with a fixed or known maximum number of arguments. The "tradition" for CP-67 and CMS was to accept commands if the required arguments were present and valid, and to ignore as comments anything else which might have been entered. One afternoon Charlie Weagle and I "explained" to Dick Newson that the additional logic to scan past the last valid argument was simply too much code—it would not fit within the 4K-byte limit for the pageable CP command modules. "Trust

us, Dick. If you make us put it in, it won't fit! We will find a way, even if the current module is only 200 bytes long...", or words to that effect

Another group of unsung heroes was the documentation group, though one or two did get IBM awards for the Release 1 effort When we started the documentation effort in early 1972, the original CP-67/CMS crew of four or five writers and support people was far from adequate to the task. The eventual answer was to bring in a group of temporary office people, ten or eleven in all, from one of the regular Boston agencies. Over the nine months or so of the initial documentation cycle, those "temporary" workers learned CMS, Script, the Ned editor, and were instrumental in producing over 3000 pages of IBM product documentation! In spite of this heroic effort, we almost lost the people in an idiotic administrative scramble which followed FCS in November, 1972.

Along the road to the VM/370 inclusion in the August, 1972, S/370 Advanced Function announcement, plans emerged to incorporate the VM/370 group into the System Development Division. I don't know whether it was a concession we were forced to make or a known condition from the start, but we did transfer from DPD-IM&D to SDD on January 1, 1973. Prior to the transfer, in the late fall of 1972, some of the group had physically moved to the IBM Service Bureau Corp, building in Burlington, Mass. (24 New England Executive Park or "NEEP", several hundred yards behind the huge Burlington Mall on Route 128). We were definitely outgrowing the space in 545 Technology Square, both for systems and for people. One of the first things added in Burlington was a second S/370 system, a 155-11, while we were still sharing the building with SBC. By February, 1973, I think, all of the VM/370 group had relocated to Burlington.

Somewhere in this process the "supplemental" people working both in documentation and in the system operations group were transferred from DPD-IM&D to Service Bureau Corp.—the employment policies for SDD apparently didn't provide for supplementals! We almost lost some of the people on the spot; they wanted to work with us, but at least two or three of them had had bad experiences working for SBC in the past as temporary help. In the next few months things got worse; all of SBC was "traded" to Control Data Corp, in the settlement of one of long-standing anti-trust suits! That provoked some relatively frantic scrambling to "rescue" our people from the trade and bring them on board, finally, as full-time IBM employees with the benefits that they deserved.

Adding people to the group over the next several years was another source of occasional tales. We were one of the very few (only?) major development groups which was located in a competitive job market. IBM's unwritten policy was to set up its large facilities just far enough away from major metropolitan areas to escape heavy competition for personnel, then pay their people a little more than the going rate. There was also a concern, inherited from the late 1950's and Tom Watson, Sr., I think, to make sure that the big plants were outside the nuclear blast radius of known strategic targets—reportedly the original reason for the production move to Poughkeepsie and Kingston. As a result, our Boston area location gave us an advantage in attracting top-level people. One of the "jewels" we attracted, in late 1973 or early 1974, I think, was George Saunders. He was a refugee from IBM Kingston, and we put him to work on the remote 3270 support

From the beginning, George adapted to the system and to the environment very well—he and I shared an office, so he didn't have much choice but to succeed (: -). Some of the other notable additions that I worked closely with were Charlie Johnson and Mark Dunn. Mark was one of our early converts; he had worked as an operator at the Cambridge Scientific Center as a Co-Op student at Northeastern University.

On one other occasion (Spring 1975?), we tried to bring in a staff-level programmer from the Kingston communications group, to work with Charlie and Mark and myself on native SNA support, but we failed for an odd set of reasons. I don't remember the man's name; he came to us with a very good background in IBM systems development and communications expertise that we needed, but no prior experience with VM/370 or CP-67. He spent two weeks with us learning the system from the outside in, then went back to Kingston! As he explained to me, he could not believe that VM/370 was possible; he understood what was going on in CP, he could see it work, but he could not make himself accept it The details of intercepting program interrupts, simulating instructions, translating I/O programs, managing virtual and real storage—all the basic things which create a virtual machine—had to require more code than could possibly be executed in the time available! VM/370 obviously worked, but he couldn't understand how it was ever possible to support more than two or three virtual machines at a time. To him, it felt like magic, and he had no confidence in his ability to work on the system. (Of course, that wasn't the only factor in his decision; housing in Boston was a lot more expensive than in Kingston, New York, but that sort of ruins the story!)

Personal Glimpses

John Seymour was one of the foundations of the early VM-CP group. He was both one of the older people around and one of the long-time CP-CMS people. John was also an extremely careful programmer. You could tell immediately if the module listing you took out of the rack was one which John had worked on most recently; every line of code, including macro expansions, had a ball-point check mark next to it from his final inspection. Difficult or critical areas of the code often had two check marks on each line. As a result, John was the local champion in fewest defects per 1000 lines of code, with a measured rating of 1.1 to 1.2, in spite of the fact that most of his work was in "simple" areas such as DMKCCW, DMKFRE, DMKPTR, etc.

Not too long after VM/370 FCS, an APAR came in which required a fix to DMKFRE, the CP storage allocation module. John found the problem and fixed it, then remarked, "There's one more."—DMKFRE was a little bit more than 1000 lines of code. About six months later, another DMKFRE problem showed up. After that second problem was fixed, DMKFRE ran without failure from 1973 until well after the group broke up in late summer 1976—executing two or three hundred times a second in hundreds of different systems! There had been just two bugs, as he expected.

John Xenakis, also known as "Captain Midnight", came to us from the Time-Life Development Center, I think. He earned his nickname in the "stretch run" from Spring 1972 until the announcement in August. John was seldom around the office during normal business hours, unless you happened to catch him on his way out in the morning or on his way in about supper time. He maintained a supply of fresh fruit and soda in his office, and he had an electric popcorn popper which he used to make rice (!), all to support his marathon work sessions. It was common to arrive in the morning and find a note on your door explaining which new features had been added to CMS since the day before. Tom Rosato, then manager of the CMS group, was probably the originator of the "Captain Midnight" name—Tom was routinely exasperated about losing control of the group, but he needed the extraordinary productivity. Not long after one of Tom's tirades, John started to use the nickname as a signature on his morning notes.

Paul Tardif was another very good person to have around, though he was only with us for six or eight months. Paul brought a certain cosmopolitan flair that helped balance the prevailing fanaticism and workaholic approach. One of his chosen social roles was to arrange our lunch trips so that we went out every day, but we never went to the same restaurant twice in one month. As I remember, he succeeded month after month in completely natural fashion, seldom letting us know even a day in advance what to expect.

Paul Tardif

Eating lunch with Dick Newson was occasionally a startling thing. I have never been a particularly fast or particularly slow eater; my mother always had just about the right amount of food, so my two older brothers and I never developed a large-family competitive eating style. Newson ate like he had grown up in an orphanage right out of *Oliver Twist.* I remember vividly one time when just the two of us went out together. Our sandwiches arrived at the same time, in the middle of a technical discussion, and Dick and I kept right on talking. By the time I had added some salt and mustard and picked up my sandwich for the first bite, I looked over to see Dick cleaning his plate!

Another one of the constructive contrasts within the CP group was between the general group of crazies and the long-service IBM FEs—Charlie Weagle, Ray Grein, Larry Estelle, and Ed Murray. For several months in the beginning, we (the crazies, of course!) had fun convincing them to unbend a little, come to work without a necktie, wear a colored shirt, or be a bit late once in a while. I think it was Ray Grein who finally explained—he had been an IBM field person for so long that he didn't *own* anything except white shirts and dark suits! The only other clothes he had were ragged jeans with paint on them, sweatshirts, shorts, T-shirts, and a bathing suit.

The 545 Tech Square building had a concentration of "personality" people, going well back to my early days at the Scientific Center. Don Hatfield was the only person I ever knew who had not a single hair on his head and owned something like two dozen white silk Nehru shirts—all the same except for different embroidery patterns in the trim or subtle damask patterns in the fabric. John Ravin was one of the easy-going younger crew at CSC, and he would occasionally bring in Bumble, his 300-pound Saint Bernard "puppy". The hazard was that Bumble loved to be friendly—by leaning on you as you patted him! I spent a good fifteen minutes one weekend afternoon pinned to the door of John's office when Bumble was feeling particularly neglected. Ed Hendricks and I spent some time trying to find a really comfortable yet durable pair of leather sandals to wear around the office in the summer.[175]

[175] Eventually we found a new leather goods shop on the outskirts of Harvard Square which made custom-fitted sandals for about 25 dollars a pair. By a completely unlikely coincidence, the owner of that store, which went out of business in the early 1970's, is now my brother-in-law.

There were also several incidents that could only have happened in the environment of Cambridge, Mass., and the Viet Nam War situation. One afternoon I was sitting in the CSC reception area reading a newspaper when a stereotype walked in the door—a medium tall man in a nondescript suit wearing a trenchcoat. He looked around furtively, then stepped up to the receptionist's desk, pulled out a wallet and said something like, "Good afternoon, I'm so-and-so from the FBI and I'd like to ask a few questions," as he flashed his badge! At the time our receptionist was Lily, a tall black woman from Haiti with a mischievous sense of humor. She watched the scene with a twinkle in her eye, but she couldn't stand it very long—she broke up laughing before she could say a word in response. It happened to be a completely innocent visit; one of the women who had worked at CSC for a while had applied for a job which required a security clearance, and the FBI was merely checking her references.

On another occasion we almost had an in-house protest. Among the early users of CP-67/CMS were both the National Security Agency and the CIA; the fact that the DAT hardware isolated each user in his own address space was viewed as a powerful system security feature. One time in 1970, I think, the CIA sent two of their people to Cambridge to talk about something that Ed Hendricks had developed or was working on. In the atmosphere of the time, none of the technical people at CSC, especially Ed, wanted to talk to them at all! Ed stormed around the halls muttering "damned spooks!" for half an hour or more before Craig Johnson and Norm Rasmussen were able to coerce him into the meeting. Even more amazing is that they -*were* spooks; there was a man and a woman, both of slightly below-average height, average build, average everything! You could stand and talk directly to them or study them for five minutes or more, but if you turned around there was nothing to remember and nothing to describe; they were effectively invisible.

As time went on, there were other hazards associated with that era. The IBM Branch Office near Harvard Square was bombed once or twice, luckily causing little more than broken glass and some outside structural damage. One afternoon in the spring or summer of 1972 (or late 1971, I'm not sure), I happened to look out our third-floor window at 545 Tech Square, only to see the whole Project MAC and MIT AI Lab crew, from the eighth and ninth floors, walking out into the parking lot and turning around to look back at the building. I quickly notified Dick Newson and Tom Rosato, and all of the IBM people very shortly joined everybody else outside. Project MAC had received a bomb threat, but nobody had thought to notify the other tenants of the building!

Tales of the IBM 3704/3705

My first run-in with the IBM 3704/3705 Programmable Communication Control Units (PCCUs) was sometime towards the end of 1970, I think. The CSC work on S/360 communications was apparently recognized within the rest of IBM, since the controller hardware group in Raleigh, North Carolina, sent a couple of people up to Cambridge to talk to us about their plans for a new front-end controller product They described it as a state-of-the-art multi-line communications control unit, capable of supporting many different line types and protocols, "programmable" through user-specified parameters. We called it "the great step forward into the past!"
Some of the earliest IBM data communications work had been done next door at MIT as part of the CTSS development using a beast known as the IBM 7750—a programmable communications control unit so difficult to handle that there were reportedly fewer than a dozen people who had ever programmed it successfully! Ed Hendricks, Craig Johnson, and I had all talked, at one time or another, to the one person at MIT who ever did 7750 programming. We all considered the S/360 2701, 2702, and 2703 hard-wired controllers a real benefit; they made it possible for highly intelligent people to write communications programs. The IBM 7750 and its brethren required a genius specialist.

What goes around comes around, I guess. In the summer of 1973, after finishing up the virtual and real channel-to-channel adapter (CTCA) support, my new device support role landed me the task of supporting the 3704 and 3705 controllers in VM/370-CP. When the 370X PCCUs were first announced, VM/370 would support them in 270X Emulation Program (EP) mode only. In addition, there was no EP utility support in CP or CMS, so it was necessary to run a DOS/VS or OS/VS system under CP to build the 370X program image and load it into the controller. This was awkward, at best; if the EP was not running when VM/370 first came up, the hardware would not respond to the individual device addresses for the emulated lines, so CP would mark all of them OFFLINE. Bringing up the EP initially, or restarting it after a CP reIPL, was a major pain for the system operators.

The first problem to be solved was what to do about the Network Control Program (NCP) and Partitioned Emulation Program (PEP) modes. One of the stranger presentations I ever made was an "exploratory" discussion of possible VM support for the 370X, to one of the SHARE project groups at the Fall 1973 meeting (in Denver, I think). I still felt a bit out of place in front of a large group, at the ripe old age of not yet 25, and I was expected to somehow talk about future VM/370 support possibilities without pre-announcing anything! I had prepared a brief overview of the various 3704/5 program modes, some material on the problems unique to VM/370, and a set of four or five alternatives for the VM-CP support (all but one of the alternatives were "smoke"; we had already decided to go for full support in CP and nothing extra in CMS).

Like a trooper I got up in front of a full meeting room and worked my way through the pitch. Every leading question or plea for a response was met with dead silence. Even at the end, fifteen minutes or so ahead of schedule, I got just one question—on some basic matter which I had touched on briefly in the first five minutes! Everyone but me seemed to be satisfied with the session—I was a nervous wreck. That was one of my first experiences with what you might call "developer's time warp": you're planning things for two years out, working on things which won't be available until next year, supporting features that were shipped last month, and helping the users take advantage of features that were new last year, with user programs that were written two or three years ago.

The actual development of the 3704/5 support was also something of an adventure. At the beginning of the project I had estimated, fairly accurately, how much work would be required to complete it. Unfortunately I had very little experience in translating the amount of work into the corresponding amount of elapsed time. The support was originally aimed for VM/370 Release 2.0, but we didn't get it out until Release 2, PLC 05.

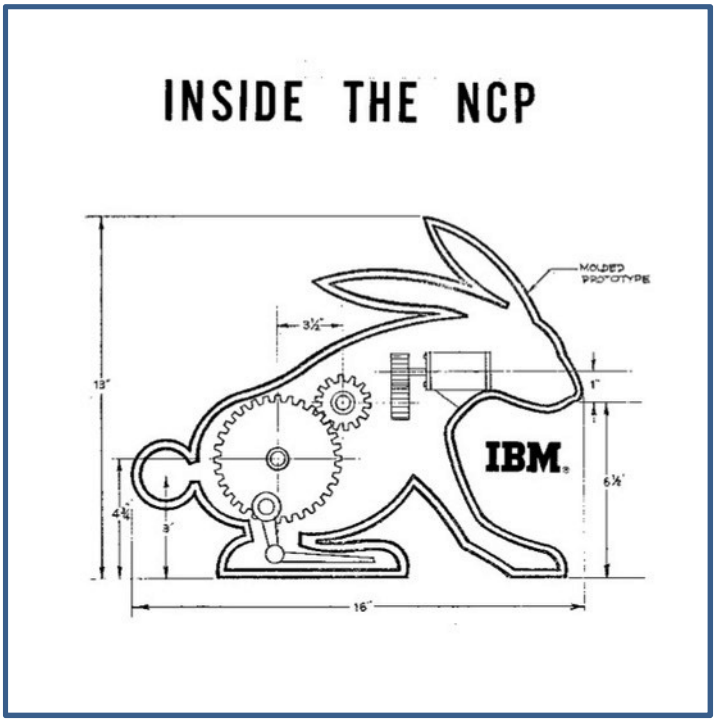
Along the way we had to develop some cooperation with the EP/NCP software and the 3704/5 hardware groups in Raleigh, since we needed to include VM/370-specific installation files on the standard distribution tape for the 370X utilities. We also enlisted two or three people from the mid-Hudson valley (or White Plains, I'm not sure) to do the bulk of the 370X generation support (EP/NCP/PEP program generation, porting the 370X macro assembler and macro library from OS/VS, etc.) in CMS. Bob Downs had to put together a limited version of the OS/VS Linkage Editor and get it running well enough in CMS to build the 370X load images, despite the fact that CMS didn't support all of the OS features which were required. I spent my time restructuring the VM-CP console and terminal support routines to provide a common device interface, trying to understand how to "speak NCP' from within CP, and doing the 370X program load and dump support

At one stage of the project I discovered that it was possible to put an IBM 2741 terminal into a state where it would gleefully ignore all characters sent to it from the host without reporting an error or causing any detectable line event. On several other occasions I was able to take down the

S/370-145 multiplexor channel, running either a pure NCP or a PEP in the 370X. That one uncovered both a bug in the model 145 channel microcode and a problem in the design of the 3705 Type 1 Channel Adapter. By that time, fortunately, we had established an unusual amount of credibility with the hardware development people. After an initial visit to verify that there was a problem, they stopped the production line in Raleigh, assembled two channel adapter cards with a probable fix, and flew them up to Burlington in the hands of one of the design engineers. An afternoon of weekend testing verified the fix, and they let us keep the new cards while they "worked the process" to release the fix as an Engineering Change (EC). For several months there was a sign taped to the inside of the 3705, warning the CEs that there were two cards that could not be replaced!

We did eventually finish the support, including the pre-SNA support for NCP and PEP mode, but I caused a lot of trouble for management by missing the schedule so badly. During the fall months of 1973, I worked a total of eight weeks of "scheduled overtime" at premium pay, a rare thing for exempt employees. IBM policy was that you should be able to do your job well working the regular hours of 8:45 AM to 5:15 PM; a healthy family life was necessary for good job performance. Any project which required too many extra hours was considered a planning failure, and the pressure was put not on the troops but on the managers!

After the 3704/5 support was released, I had the opportunity to go back to the SHARE VM/370 Project (August, 1974, in Denver?) and explain what it was that we had done. I've included a copy of the presentation material in the "CARE package" of VM/370 memorabilia.



Once again I misjudged the audience, but the presentation still felt better than the first one. That session was the scene of another coincidence; during the once-around-the-room introductions at the beginning, I discovered Ron Zeilinger, one of the small crew of MIT Computer Center people that Ed Hendricks and I had worked with in 1967!

Answers to some questions you've asked

PER and VMTRACE: There were several factors involved in the decision not to ship the support for Program Event Recording, in addition to the problem with the Model 145 microcode. The prototype that I developed was aimed toward natural support of the PER hardware features, rather than toward a well-rounded virtual machine debugging or trace facility. Many of the basic PER capabilities were functionally equivalent to the old CP-67/CMS TRACE command features, but the CP-67 TRACE capability offered some things which PER did not, and vice versa. In the best of all possible worlds we would have used the PER hardware features for many of the basic functions, used code or techniques from the CP-67 TRACE facility for some extended functions, and added new functions based on intelligent manipulation of the PER features. Unfortunately we had neither the calendar time nor the resources to do the complete job. We reluctantly made the decision to ship only the "traditional" TRACE functions. Releasing both capabilities would have created a very difficult documentation task, at best, and a lot of user confusion.

The VMTRACE capability that appeared in Release 4 is not something that I had any direct experience with, but it is vaguely familiar. Dick Jensen is almost certainly the author. The problem that it was intended to address started with the 3704/5 support in Release 2 (PLC 05?) and was aggravated with the advent of remote 3270 support—the VM/370-CP internal trace table became very difficult to interpret. There were a lot of new event codes introduced to handle both the NCP Version 2 support and the BSC remote 3270 support. At the same time, VM/370 was being used in larger system configurations to support much larger user populations, thus creating more trace entries per second. Even with the very large trace tables which were available in large-memory systems, the real clock time history recorded in the trace table was seldom more than a few minutes.

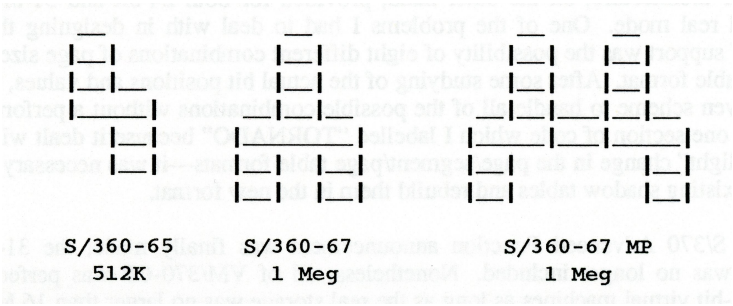
You may or may not remember that the CP internal trace table was originally released as an optional feature. We strongly recommended that users include it, but it was possible to remove the function when you were running in a small system (256K-512K nominal). I was one of the unfortunates who had to try to find a CP problem in a dump without a trace table—very close to being a lost cause, even for VM/370 Release 1. I don't remember exactly when we gave up on the optional status, but it was relatively soon after the initial release; the IBM Field Support organization let it be known that VM/370-CP without the internal trace table was not serviceable. Nonetheless, the facilities existed within CP to run without creating trace table entries. These hooks were first used internally on an experimental basis, in cases where we needed a longer history of selected events rather than a full record of CP activity. The DMKVMT module was most likely developed as an internal system test or development test tool, then extended to include the print, data reduction, and statistics support.

Systems at the CSC: When I first started working at CSC in the Fall of 1968, we had a 512K S/360-67 in a machine room which occupied the northwest comer of the third floor at 545 Tech Square. The small machine room which was in the center of the CSC (west) half of the fourth floor contained the IBM 1130/2250 Model 4 combination. Sometime in 1969, probably as a collateral event to the separation of the CP-67 support group, the CSC machine was moved down to a new machine room which occupied the west half of the second floor, along with some office space for the operations people and for Fritz Giesin, our "modelling engineer" (more on Fritz, later). We needed the extra room to expand the disk storage, add a paging drum or two, and add extra terminal lines. Sometime in 1970,1 think, we also picked up a second S/360-67 CPU and some more memory—it may have been the machine used by Lincoln Lab, but I'm not sure

The second CPU was not very useful at first; we did not have the extra channels and I/O equipment to set up two full systems. The first thing we did was to add some of the memory to

the original CPU, while Fritz tried to figure out how to make a CP-67/MP system out of two uniprocessors. If you ordered an MP system from IBM, it came with special logic in the storage bus controllers and with a "left" and "right" CPU. The original CSC system was MP-capable as far as the bus logic was concerned, but both systems were "left"-handed. According to standard IBM practice, it was not possible to put the two CPUs together in an MP configuration. As was typical of the CSC crew at the time, Fritz was not one to turn away from an "it can't be done" challenge.

A standard S/360-65 consisted of two CPU cabinets, a "crossbar" cabinet with bus cables and power control, and one to four cabinets worth of core memory. The S/360-67 uniprocessor was similar, but it had an extra CPU cabinet for the DAT hardware and a lot more internal logic to support 32-bit addressing in virtual mode. The physical layout of the systems was something like this:



The maximum memory configuration was 1 Megabyte, I think, due to cable length and timing limitations. When Fritz had finished his investigations and constructions, we had a fully operational S/360-67/MP system that was somewhat unconventional. Instead of a standard IBM cabinet as the crossbar of the "H", there was a 6-foot by 3-foot panel of 1/4" Masonite supporting a neat array of IBM-standard internal cable guides, dozens of flat cables interconnecting the two bus controllers, and the necessary array of relays and switches for power sequence control, memory unit control, and I/O interfaces. Essentially all of the interconnecting hardware was ordered as spare parts and put together according to the basic MP schematics, with some unofficial help from the Boston-area IBM CE specialists. The MP system was ready long before we had the CP-67 support for it; for a long time we ran the collection as two separate systems.

Virtual 67s and Such Things: Alain's recollections are probably more reliable than mine for the period he's describing. He was the major contributor to the virtual 67 work, but Charlie Salisbury and Bob Adair were involved at least on a regular consulting basis. At the time, Dick Newson and Lynn Wheeler were both in the CP-67/CMS Support Group and not working directly in the CSC, though the separation between the groups was more administrative than it was technical or physical.

The system version which Alain referred to as CP-67VE was probably called the "E-System" internally, leading to the lettering nomenclature which eventually generated a "G-System", "H-System", and "I-System". As I remember it, the lettering started because you had to be able to tell which CP you were talking to from the terminal. When you had a "stack" of virtual systems, you had to work your way down to the correct CP or CMS level by multiple hits on the old 2741 "ATTN" key. Each system we put together was given a different version of the "CP" prompt so that it was possible to tell where in the stack you were. Not all of the CP-67 versions were in the same sequence; the virtual MP support was based on the virtual 67 branch and never had any of the virtual S/370 support.

The first release of the VM/370-CP product was a rewrite which contained 5% or less of unmodified code from any of the CP-67 versions. My memory is a bit vague, but I think that the last system Alain produced was "CP-67H", which ran on the System/370 machines but did not support any of the newer I/O devices. Paul Tardif started the I-System work, and I joined in a little bit later. The remarks on 2K pages and 64K segments I'm not sure about; it is only marginally possible to support 2K paging on a S/360- 67, but I do have some crazy recollection about an attempt to do so. Alain may have been referring to the need to interpret the additional control register bits which were defined in the S/370 DAT architecture.

As an optional feature the S/360 Model 67 would support 32-bit addressing in virtual mode, a feature which was required for TSS/360 if I remember right. The virtual 67 support in CP-67 included the 32-bit option, but neither CP nor CMS were ever modified to take advantage of it. The S/360 physical memory address bus was never more than 24 bits. The early definition of the S/370 DAT architecture, on the other hand, provided for both 24-bit and 31-bit addressing in virtual and real mode. One of the problems I had to deal with in designing the VM/370-CP DMKVAT support was the possibility of eight different combinations of page size, segment size, and page table format. After some studying of the actual bit positions and values, I came up with a table-driven scheme to handle all of the possible combinations without a performance penalty. There was one section of code which I labelled "TORNADO" because it dealt with the situation of an "in-flight" change in the page/segment/page table formats—it was necessary to throw away all of the existing shadow tables and rebuild them in the new format.

When the S/370 Advanced Function announcement was finally made, the 31-bit addressing capability was no longer included. Nonetheless, all of VM/370-CP was perfectly capable of running 31-bit virtual machines as long as the real storage was no larger than 16 Megabytes. We had also been very careful in the CP control blocks to allow eventual extension to 31-bit real addressing; the real page frame table and the I/O CCW formats were the only areas that would have required changes. The other hooks and holes we left were for multi-processor support and full multi-path I/O support—more things that might have been done much earlier and, perhaps, much better.

Of Editors and Ned: I'm not familiar enough with the realm of UNIX[176] add-ons and add-ins to know whether the UNIX "ned" is related to mine. The CMS version of Ned was completed in the first half of 1970, and I never had enough free time to go back to it and do any major extensions. One of its early virtues was reentrancy; it would run happily in a shared segment I'm pretty sure that the VM/370-CMS version of it was included with the "July 5 System" distribution, which continued the spread of Ned within IBM. For quite a few years I was subject to periodic double-takes or attacks of paranoia when I would answer the telephone and hear something like "Hello, this is so-and-so in Corporate Legal in Armonk...", only to find out that they were avid users of Ned and needed some support. One thing that it apparently did better than any of the other CMS editors was to handle text editing, rather than program editing.

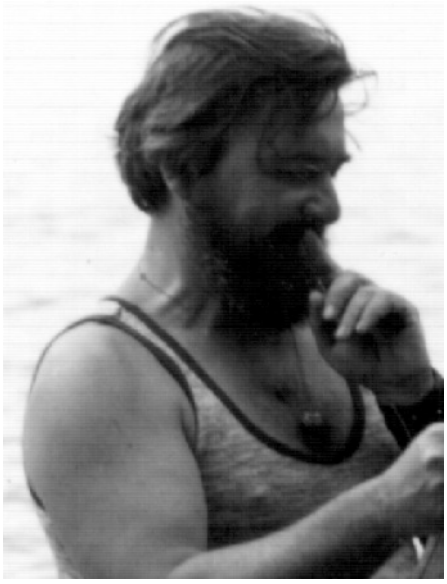
On another occasion I was flabbergasted by Stu Greenberg, sometime late in 1970. He came to me with "a little Ned macro problem", then showed me a 2400-line macro program which took an all-upper-case file of word occurrences, six or eight words per line, and reformatted it into a one-word-per-line, leading caps index in alphabetical order. This for my poor little in-storage macro processor with a lousy syntax and "buggy" conditional handling which was less than 4000 bytes of code! One lesson which I learned well from Ned was, "Don't make assumptions about usage." The users of anything you produce will almost instantaneously discover five or more

[176] UNIX is a trademark of AT&T Bell Laboratories.

things which it was never intended to do, which almost work, and/or they will immediately use it at two or more times the intended maximum load or configuration.

More Personal Glimpses

Fritz Giesin (I hope I have the spelling right) was one of the lesser-known heroes of the Cambridge Scientific Center. His title was something like "Models Engineer", and he was sometimes known as the only CE in IBM who had his own budget. Fritz was the hardware person who actually assembled the DAT box for the S/360 Model 40 which was used to develop CP-40. For Ed Hendricks and I, Fritz designed and put together two or three generations of synchronous modem eliminators to support our BSC communications work. He was also the creator of the interface between the Sylvania Data Tablet and the 2250 Model 4 scope which allowed SketchPad III to support high-resolution free-hand drawing. (The standard 2250 supported a light pen, but it was necessary to generate a "tracking cross" on the screen to find out where the light pen was; its position sensing was based on comparison with the position of the CRT display beam.)



Fritz Giesin

Over the years Fritz contributed a lot of different gadgets and simple things to support both the CSC systems and, later on, the S/370 systems used by the VM Development Group. As we expanded the amount of disk storage on the CP-67 machine, it became more and more difficult to tell somebody where a disk drive was. We had five banks of 2314s, nine drives per bank, and it was possible to switch the controllers between two different channels. The solution was color coding; IBM offered two different color schemes for S/360 equipment, but that clearly wasn't enough. One weekend Fritz and a helper took all of the exterior panels from three of the 2314 disk banks to an auto-body paint shop over in Somerville and had them painted in three different colors! From then on it was easy to explain—"Go over to the Orange bank...", or red, green, blue, grey!

When it was necessary to put together some special piece of gear, be it a coaxial cable patch panel for local 3270s or the complete arithmetic unit which he once added to a "surplus" 2250 Model 1, Fritz was more likely to go down the street to Eli Heffron's Solid State Sales (the local electronics surplus/discount house) or to stop at the local Radio Shack than he was to order from the normal distributor channels. For mechanical assemblies or high-power bus bars, Fritz had some arrangement with MIT which let him use the machine tools and power equipment in the hobby shop. He enjoyed learning new techniques and attacking new problems, and he always found some way to do what we wanted—just the right sort of magic to provide engineering "backup" for the CSC, CP-67, and VM/370 software people!

Philosophical Matters

Development, distribution, and support for VM/370, in the Cambridge and Burlington days, was handled in a variety of ways which were very unusual for IBM products. Some of the procedures and attitudes grew out of the group's organizational history (DPD-HQ and DPD-IM&D), but many of them were the result of individual people's advocacy and obstinacy. It's a little tricky discussing specific things, because I am not at all sure that I know exactly who was involved in a lot of the important decisions. In several areas I was one of the torch bearers over time, but I really don't know if I was ever responsible for "lighting the fire". The areas I'm referring to include these:

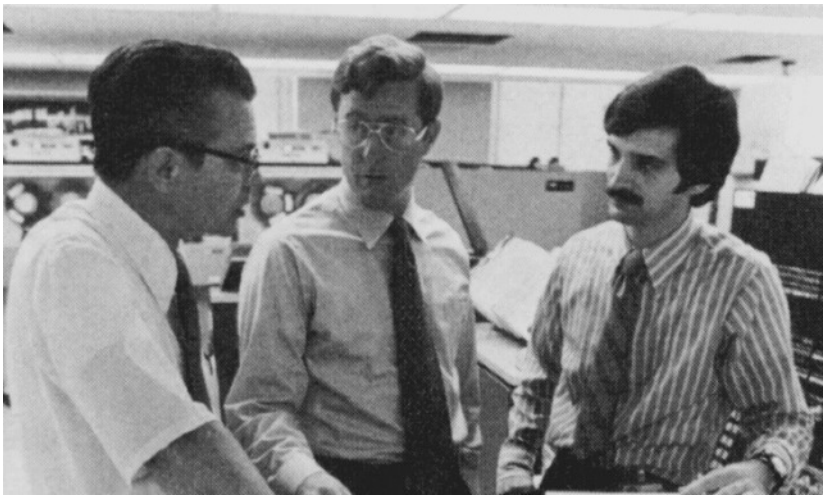
Source Level Distribution: This was probably a follow-on from the days of the Type III Library and CP-67/CMS, because we were aware of the many valuable contributions and extensions from other groups within IBM and from the user community. As you have mentioned, it was an important factor in the acceptance and eventual success of VM/370. Selection of standard S/370 Assembly Language was also a factor; Dick Newson, Tom Hennesey, and Dick Meyer had to resist some fairly strong pressure towards using PL/S as the development language.

Ed Christiansen, Larry O'Dell, Tom Hennesey & Ben Pettersson

Distribution of the Development Tools: By this I mean both the UPDATE facilities, VMFASM, VMFLOAD, VMFMAC, etc., and the load lists and control files necessary to build
the VM/370 components. The decision to ship the system with all of the tools was somewhat separate from the decision to ship the source code. The alternative which was discussed was to ship only "finished" source modules—*i.e.,* source code with all of the updates applied to the base. It would have been possible to add changes on top of the basic source, but it would have been much more difficult for users to make changes and keep track of them.

By the way, the multi-level UPDATE which I originally coded was written to the conventions of CP-67/CMS, not VM/370-CMS. John Xenakis took the "tool" version and did some fairly extensive modifications to fit it into the new CMS structure of command line handling, file system macros, error codes and message handling.

Frequent and Cumulative Update Releases: The Program Level Change (PLC) procedure was completely alien to the normal SDD way of supporting products. It was either invented by the VM/370 support people, with Dom Lacava, Ron Snell, and/or Larry O'Dell as the major advocates, or it was an adaptation of some similar procedure from their Time/Life or BPC DPD-IM&D background. It amounted to a serious commitment of system test and integration resources with the deliberate goal of quick service turn-around. We routinely had as many as four or five different PLC levels of VM/370 "active" at the same time, something which would not have been possible without both source-level distribution and the distribution of the development tool set. The original goal was one PLC per month, with a target APAR turn-around of no more than two PLC cycles (submit a problem on PLC "N", have an official fix incorporated no later than PLC "N+2"). The cycle stretched out after the first year or so only because the number of fixes went down!



Ron Snell, Larry O'Dell & Dom LaCava

Development/Release Process for "Minor Enhancements": The minor enhancement process is something that may or may not have been apparent to the user community, and I don't remember exactly when it started or how long it survived. The concept was to provide an official mechanism for adding small features or extensions to VM/370, outside of the main development planning, specification, approval, etc., cycle. The guidelines as I remember them were no more than one man-week of development effort and no more than one man-month release effort (documentation, test, etc.). For each PLC cycle there was a limit of something like five minor enhancements, varying based on the amount of normal-cycle fixes and new function included. The CP "SLEEP with Timeout" extensions that I developed were one example, and I think the CMS in-storage UPDATE support was another case. The same process was used, on a number of

occasions, to incorporate features which had been developed outside of the Burlington VM/370 group.

High Availability / High Integrity Approach: The basic architecture of a virtual machine system gave us many advantages in achieving a highly reliable system, but there were some early "traditions" that contributed as well. The separation of function between CP and CMS allowed us to develop CP entirely as a "closed" environment; there was no user code in CP and there was no possibility of user code access to the real system address space. This led to the CP principle of "fail early, fail often"—now recognized as one of the classic approaches to achieving high availability but somewhat radical at the time. The underlying principles were:

1. Nothing which happens in a virtual machine can be allowed to result in a CP failure;

2. No recoverable error in the real machine realm can be allowed to result in corrupted data;

3. Anything which looks like an internal problem (control block inconsistency, "can't happen" code path, pointer errors, etc.) should result in an immediate CP abend.

Part of the trick was to stop CP as close to the failure as possible. The internal CP trace table contained enough data for us to completely reconstruct the code paths leading up to any failure, as long as we didn't cover up the trail by trying to recover.

Another result of the CP approach was that CMS and its file system had to be built to survive a CP failure at any point without corrupting the disk structure. Jim Walsh was the primary file system wizard, if I remember right. He worked on the minidisk I/O and file system code until the only exposure was interrupted execution of a channel program. It was possible to lose data, but it was not possible to lose a minidisk unless the channel or control unit failed badly.

Fix the Problem, Not the Symptoms: This has always been one of my personal commandments, and it was one of the things which was surprisingly hard to teach. Essentially all of CP was critical path code in one sense or another, since all but a few instructions in the dispatcher ran with interrupts disabled. We couldn't afford a lot of special-case checking or exception handling—the structure and basic logic had to fit the problem. As we added support over the first few years, we had to be willing to change the existing structure when new function or fixes required it

On more than one occasion I had to be quite stubborn to avoid the "minimum fix" approach which was the more normal IBM way of doing business. One of the fairly early PLCs on Release 1 contained an "update" to DMKVAT which replaced the entire module, because early development testing of OS/VS2-MVS had uncovered a basic misinterpretation of the DAT validity checking. Later the same year I ended up rewriting almost all of DMKCNS and modifying all of the CP console I/O modules to make a place for the 3704/5 EP and NCP support and the subsequent remote 3270 support.

Another aspect of the same principle could be expressed as "Don't be afraid of the big change." We started running into a problem with Release 2 because there was a limit on the number of real I/O devices which could be configured in DMKRIO, based on the initial use of 16-bit offset fields in the control block definitions. As people started using local 3270s on very large VM/370 systems, it became more and more of a problem. I came up with a technically simple fix—using the 16-bit offset as a count of double-words instead of a count of bytes—and it only required updating every CP module which referenced the real I/O blocks!

Use the System You Ship, and Ship What You Use: With the exception of the Ned editor and a few IBM proprietary tools, the system which we used every day in the VM/370 group was exactly the same as the released (or soon-to-be-released) VM/370 product We did not use the elaborate CLEAR/CASTOR source control system which required proprietary terminal hardware. We avoided using PL/S, IBM's proprietary "structured assembly" language. We did, for a little while, use an internal tool called "FL/1" which generated flow charts based on statements embedded in the module source, but we were able eventually to talk our way out of the requirement for flowcharts in the program logic manuals. Most of the diagnostic aids and performance monitoring or measurement tools had to be developed, tested, and documented at close to product quality because of the number of internal IBM sites depending on VM/370 for their own development. Extensions such as the VNET code in RSCS perhaps took a while to get out in product form, but even that made it eventually.

Measured Performance, Measured Reliability, Measured Integrity: These aspects were in line with IBM's stated goals, but the VM/370 structure made the goals more than usually achievable. There is enough of a tale associated with each one to justify discussing them separately.

1. *Reliability:* We started monitoring the system reliability even before we shipped Release 1, and simple good luck had a lot to do with VM's early reputation. Sometime shortly after the internal release of the "July 5th System" in 1972, we started numbering the development versions of VM/370-CP and keeping track of the mean time between failures. The code which we shipped as VM/370-CP Release 1 was "System 54"; there had been that many or more different systems between mid-July and the code freeze for FCS (October?). The normal range of reliability for CP systems from System 40 or so onward was 3-1/2 to 5-1/2 hours between crashes—one or two crashes per working day, on average. CP System 53 was on the low end of the range at about 3.6 hours MTBF, but we knew that we only had time for one more system build prior to FCS. VM-CP System 54 was built in exactly the same way as the previous versions, with more or less the same number of fixes. Miraculously its measured reliability was something over 38 hours MTBF, an improvement of a full order of magnitude!

   Both the measured performance and the measured reliability of CP were accepted by the group as requirements for ongoing releases. The system test procedures for the first couple of years included a full regression test and MTBF determination, even for monthly PLC tapes. Sometime after the first 18 months or so, the MTBF testing was dropped from a requirement to a goal; it was seldom possible to measure it because the system wouldn't fail in a full week of continuous testing. The move to Burlington in 1973 and the second S/370 system allowed us to do "non-intrusive" testing; we used one machine as a terminal simulator and measurement vehicle so that we could run a completely standard VM/370 system in the other.

2. *Performance:* The ongoing research work in the Cambridge Scientific Center and IBM Yorktown Research was a significant benefit in the pursuit of VM/370 performance. Even before the VM/370 effort started, Don Hatfield had developed a "profiler" capability which would measure and record the address references of a virtual machine during execution. By plotting the results over time against the load map of the program, he could identify opportunities for rearranging the program to improve working set characteristics. As you might imagine, a full run of the profiler was a significant effort; the CalComp plotter on the 1130 system would run for several hours to produce a single output map. During the VM/370 Release 1 period we profiled VM-CMS several times and adjusted the module loading sequence to improve its locality of reference. We also profiled VM-CP, running in a virtual machine, to identify which code paths were the most critical. About 11% of the CP resident nucleus accounted for a large portion of the CP execution time; many of those code paths were the target of Carl Young's first scheduler update.

The well-known scheduler work of Lynn Wheeler was another example of successful "applied research"; his triumph was a remarkable balance between sophisticated algorithms and the code execution time needed to do the analyses. Many of the algorithms which perform best in theory cannot be implemented in practice—the system time needed to execute the algorithm is greater than the time saved by greater precision. This principle was demonstrated quite well by OS/VS2-SVS; their code for real page frame management was much more sophisticated than anything we had in VM/370, but we could sustain a paging rate more than 10 times their maximum! The MVS people took advantage of our experience, but they were saddled with a much less efficient real I/O path.

3. *Integrity:* Dick Jensen, our "resident malicious user", should get a lot of credit for VM/370's resistance to penetration. His self-assigned goal (at least originally; it became official later on) was to find ways of either crashing CP or gaining access to system resources which were outside of his virtual machine definition, using only the Class G "general user" privileges. He came armed with a full set of the CP listings and a wicked imagination. As he found the problems, we had to find ways to fix them. I ended up with the unlovely task of reworking the ISAM self-modifying channel program support to close one of the better-known "windows" into CP. Dick was also involved, I think, when IBM commissioned the Systems Research Institute in California to do a formal penetration study of the VM/370 system. Their study turned up about 15 areas which needed fixes or improvements. Over a period of a couple of years we were able to fix everything except the ones which fell into the "denial of service" category. There was an IBM Research version of VM/370 which handled even that problem, but it was based on the principle of delivering each user a specified level of performance—no more as well as no less, even if you were the only user on the system. The notion of running slowly as a single user on a S/370-168 limited the popularity of that system!

Things That Might Have Been

Through the early years of VM/370 there were a number of projects which either never got to the drawing board, never got truly started, or never got finished. Some fall into the "wouldn't it have been nice" category, while others raised stronger feelings of hunger or desire. I surely can't remember all of the ideas and notions, but I can tell tales of a few of the larger ones. The prototype support for Program Event Recording (PER) is a tale you've already heard; here are a few others.

Remote 3270-BSC Support for NCP Version 2: VM/370 support for display terminals was a round-about development which started with Dick Newson. After VM/370 Release 1 was shipped, he retreated *(via* a promotion to Advisory Programmer) from managing the CP Development group to a technical position in the newly-created CP Advanced Development group. One of his first tasks was to develop the system console support for the new S/370-158 and S/370-168 systems, both of which came with display-mode-only consoles. He created the infamous VM READ/CP READ/RUNNING paradigm for the 3066 console of the S/370-168, then adjusted the display size parameters for the 3158 console.

[Let me digress for a cute tale.] The first time that Dick had a chance to test on a real Model 168, he came back from Poughkeepsie with wonder and greed in his eyes. The 168 was at least three times as fast as any of the machines which had ever run CP-67 or VM/370. Tasks like running VMFLOAD to gather the CP object files into a load deck, which ran for several minutes on our machine, completed so fast that Dick found himself double-checking the results—he couldn't believe the speed! [Back to the main tale.]

After Dick Newson had the display support working for the system consoles, Charlie Weagle (CP Development) took it over and added support for local (channel attached) 3277s. The next step, this time by George Saunders, was to add the BSC communications to support remote 3270s. George started working on the remote support some time after I had started on the 3704/5 EP, PEP, and NCP support, and we shared both an office and an architecture issue. The 3704/5 NCP, even in its pre-SNA Version 2 days, multiplexed the I/O activity of many lines over a single channel address. The existing control block structure for real I/O had to be extended to provide both another level in the device hierarchy and a different kind of device inter-dependency. The status of devices controlled by the 3704/5 was dependent on the status of the controller program, but the 3704/5 "native subchannel" address could be any convenient value. There was no automatic correlation of device and control unit addresses.

George Saunders had much the same kind of problem with the remote 3270 support, since each BSC line could support one or more 3272 cluster controllers, and each controller could support up to 32 display terminals, printers, or both. Somewhere in the shuffle we got only part of it right. The remote 3270 support took advantage of the four-digit DEV references which I created for the 3704/5 NCP support, but we didn't solve the problem of handling a really large terminal population. The four-digit reference used the first hexit (four bits) to identify which 3704/5 or which BSC line the device belonged to. Unfortunately, that limited us to a maximum of 15 or 16 lines and/or 370Xs, and it made it very difficult to combine the two capabilities—*i.e.,* to support BSC 3270s through the 370X NCP. The CP overhead of BSC polling has been a significant performance/response time factor ever since, and it is one thing that could certainly be done better by a smart comms front end.

Direct channel programming of synchronous data communications is messy; it requires a lot of I/O operations, a lot of interrupts, and it is dependent on real-time response. CPREMOTE and all of its successors existed partially to keep the mess out of the CP nucleus. One consequence of that goal was the need for some very "fault tolerant" I/O programming in the service virtual machines, bearing little resemblance to IBM's recommended error detection and recovery techniques. The 3270 remote display support, unfortunately, needed to be within the scope of CP. Having the system kernel depend on the operation of a user-level virtual machine was something that we did not (and I still don't!) believe in. There might have been an alternative, however...

The SYSTEM Virtual Machine: One of the things which never got past the "blue-sky" stage was a notion to actually implement the SYSTEM virtual machine in VM/370-CP. There were a variety of functions which belonged in CP architecturally but did not really belong in the interrupts-disabled, critical path, real address environment of the basic kernel support. Unit record spooling and terminal I/O (distinguished, perhaps, from system console I/O) were the first likely candidates, along with some subset of the CP user commands. Later opportunities might have included SNA support for virtual machines, disk volume management for the 3850 MSS, page migration, shared minidisk support, a file system for CP, etc.

The underlying notion was to create a virtual address space which included the resident CP nucleus and free storage areas, but was otherwise pageable and runnable like a virtual machine. With not too much internal finagling, it should have been possible both to reduce the size of the resident nucleus and to make more of the CPU time available for dispatch competition. The modular structure of CP, and the regular use of an SVC-based calling convention, should have made it possible to get the best of both worlds—a small, fast, highly reliable kernel with a lot of extended services for virtual machine systems.

True "Warm Start" After CP Abend: The possibility of a "selective" restart after certain kinds of CP failure was another idea which didn't make it much past the discussion stage. I should remember the one or two people who were the most vocal advocates, but I can't be sure if it was Larry Estelle, Ed Murray, or one of the CP support people. The idea came up originally in 1974, I think, and I was one of the major skeptics, perhaps unfairly. At the time CP was fairly reliable and, because of the quick dump-to-disk and restart features, highly available. Nonetheless it was very disruptive to lose everything in the virtual machines, especially for large user populations.

The idea was to add another function to the chain of programs which were invoked by a CP failure. It would first examine the CP Abend code to determine if a "warm start" could be considered. Machine checks, I/O errors which affected the channel activity, and certain internal errors could never be considered due to a loss of system integrity. Some number of failures, however, could probably be isolated to a single virtual machine. In those cases it would be (theoretically) possible to "crash" just the failing user, refresh the CP storage areas modified by the dump process, and restart the system after a very short interruption. Since the warm start would not be attempted until after a full dump was taken, it should still have been possible to track down the original problem.

It was a glorious idea, but the complexity of an on-the-fly system integrity check scared me then, and it still does. The design coordination required between the warm restart process and any new function would have put even more strain on the system development cycle. We would also have created the possibility of long-term operational degradation. Each failure and successful restart might require "abandoning" some amount of CP free storage (virtual machine control blocks, I/O chains, etc.) which could not be safely recovered. The more successful the implementation, the more time would likely elapse between "cold" starts, and we might have had to deal with some very interesting long-term effects! (Much of this is perhaps self-justification; I wanted it to be possible but I couldn't believe it. Ironically the system which I use regularly now, PRIMOS on the Prime 50-Series, has incorporated a similar "warm start" capability, successfully, for many years.)

Native SNA Support in VM/370-CP: This is perhaps one of the tales which you have been most waiting for, due to the "wonders" of the original VCNA and the subsequent GCS-VTAM support. The original attempt at SNA support started in 1974, as a natural follow-on to the initial 3704/5 EP and pre-SNA NCP support. I spent the middle of 1974 learning about SNA and some of IBM's other communications futures—some which never saw the light of day, others which developed into such things as the Twinax interface and the IBM Token Ring.

As a personal note, that period included my first contact with Jim Cannavino.[177] He was working with the VTAM communications group in Kingston as an educator/trainer, and he was one of the primary internal teachers for SNA and the other advanced communications architectures. His energy and obvious enthusiasm made him a dynamic speaker, and he had a very good grasp of both the subjects and his audiences. We got together on a few other occasions on a personal level, trading "tall tales" of our early days in the industry and with IBM, but our career paths led off in two very different directions. I think IBM is very lucky to have kept him.

[177] Editor's note: Dave has confused Jim Cannavino and Jim Cavet here. See his February, 1991, addendum. [MWV],

Overlapping the technical exploration and evaluation that I was doing, others within the VM/370 organization (Ed Christiansen, certainly, and probably Dick Newson as well) were fighting the "company line" which mandated (at the time) that there would be only one SNA access method—VTAM. Since a stripped-down version of VTAM required a working set which was larger than all of CP and most of CMS, combined, we were less than thrilled about the prospects of porting it into CP. Nonetheless, we were convinced that the support had to be in CP, for architectural reasons, rather than in a virtual machine. We got the reluctant go-ahead to develop a "native" SNA capability in VM/370-CP sometime late in 1974 or early in 1975. The combination of VTAM's size, its dependence on an underlying DOS/VS or OS/VS system, and the fact that it was coded in PL/S, all were factors somewhat in our favor.

Along the way, in October, 1974, I started attending the internal SNA Architectural Maintenance Board (the "AMB") as the official VM/370 representative. The AMB consisted of a chairman with dictatorial power and a changing group of people who were empowered only to "assist" his decision making. If discussion could not resolve an issue, the chairman could, by fiat. The board met every three weeks or so for two or three days, generating an inch or more of minutes, action items, status, and correspondence from each meeting. Attendance was seldom as few as 20 and occasionally as many as 35, consisting of several SNA Architecture people (Jim Gray was one of the original architecture "heavies") and one or more representatives from each development group, hardware and software, which was working on SNA-related projects.

The purpose of the AMB was to make decisions and compromises on a timely basis. Any member could surface an issue—architectural, implementation, or a "variance" request—with a guaranteed resolution period of about two months. New issues or requests would be included in the minutes for meeting "N", presented at meeting "N+l", discussed at meeting "N+2", and a decision was guaranteed in the minutes for that meeting. It was a demanding experience, but it was also an invaluable connection to the other activities within IBM. The meetings alternated between Kingston, New York, and Raleigh, North Carolina, with a once-yearly excursion to one of the other IBM locations doing SNA work. The AMB regulars came to be very familiar companions; at least half of the group was staying in some nearby hotel, regardless of the meeting location.

The SNA support which we had mapped out for VM/370 included native CP support for all of the necessary System Services Control Point (SSCP) functions, SNA terminal support for the 3270, 3767, and 3770 (in 3767/LU-l mode), and a DIAGNOSE interface for virtual machine access to NCP-SNA resources. My five-minute, off-the-cuff estimate was 25,000 lines of code (assembler) for the CP effort; Ed Murray and Mark Dunn's module-by-module detailed analysis pegged the effort at something like 25,154 lines, added, modified, or deleted. The vast majority of the "Utility" support—NCP configuration, generation, image build, downline load, etc.—could be carried over intact from the earlier NCP-2 support. [Remember that we were dealing with an SNA which was significantly simpler than it is today; VTAM had not yet been shipped, TCAM SNA support was just getting started (also in defiance of the one-and-only-one SNA principle!), and there was not yet any real allowance for multi-host SNA networks.]

While I was spending a lot of time on SNA architectural issues and keeping track of the rest of implementation efforts, the other four on the project team (Ed Murray, Mark Dunn, Charlie Johnson, and one other whose name I can't recall) were working hard on the specifications, design, and coding. Through the first half of 1975 we completed a full set of functional and preliminary design specifications, validated our sizing estimates, and got approval to continue the project. Along the way I had to educate a fair number of the AMB people about the unusual problems or "viewpoint" arising from a multi-system system like VM/370. They hadn't yet considered the separation of the SSCP functions from the data-stream functions, dynamic

reassignment of SNA resources between subsystems, or the addressing problems inherent in a large SNA network. For almost all of the "standard" SNA host support it was "one terminal, one application, forever and ever" in the minds of the developers.

In July of 1975, after a month or so of actual coding with no technical surprises, the whole project was cancelled. It was one of the cyclical periods when IBM was under pressure to show quarterly earnings growth, and they had been investing heavily in the infamous "Future System" or "FS" projects. Many of the FS efforts were in trouble; they were targeting very ambitious technology and it was not always possible to do "invention on demand". As a result, a significant amount of the VM/370 group's resources were "temporarily" redirected to helping out with the next generation of machine and system architecture.

In January of 1975 I had missed the AMB meeting in Boca Raton, Florida, because I was called into a two-week task force in Palo Alto, California. The task force was commissioned to come up with an advanced interactive computing capability for office automation, ideally incorporating the best features of CMS, TSO, and ATS (IBM's mainframe-based "OA" offering). During the second half of 1975 and the first part of 1976, many of the "Virtual People" were busy on a series of task forces and architectural reviews. IBM was trying to salvage what it could of the FS technology and hammer it into some implementable products.

The good news was that they recognized that a virtual machine capability would be required in any new system architecture, and that the virtual machine expertise was in the VM/370 arena. Part of the bad news was that we couldn't be doing two or three things at once; VM suffered from the diversion of effort. The other bad news was that the Poughkeepsie technical and management establishment were not particularly comfortable depending on a small group in Burlington, Massachusetts, which preferred to do things their own way, and there was a lot of travel expense involved in our working together. To a large degree our own success was one of the elements which lead to the Burlington closing in the summer of 1976.

MHCP—The M/H Control Program: One of the exciting things to come out of the task force efforts in late 1975 and early 1976 was a dream called the "MHCP"—a ground-up combination of the best of OS/VS2-MVS, VM/370-CP, and VM/370-CMS. Its name came from the internal code-names for the next machine generation; IBM envisioned a series of entry-level "E" machines, mid-range "M" machines, and high-end "H" machines. The E machines were to be the province of an extended DOS/VS and an appropriate VM capability; the M and H machines were the province of the "big boys"—OS/MVS and some form of VM/CMS, VM/TSO, MVS/TSO, MVS with virtual machines, etc.

The E machines were released eventually as the IBM 4300 series. The initial design of the system did not include a S/370 compatibility mode at all. The first pass at the architecture called for a dramatic extension of the Virtual Machine Assist (VMA) concept to a fully microcode-controlled single virtual address space, such that the DOS/VS system would never "see" the real memory space. I personally led two brief task forces which were asked the question "Can you build a VM system for a machine like this?" The first examination took about a week; the second, with a slightly modified E-System architecture, took us less than a day. The answer in both cases was a simple "No." Thus the 4300 systems came out, some time later, with a full S/370 capability in addition to the DOS/VSE ECPS mode.

For the mid-range and high-end systems, IBM was searching for opportunities to take a step beyond the basic S/360 and S/370 architecture. The very high density packaging technology of the Liquid Cooling Module (LCM) and the continuing decrease in semiconductor memory costs were leading quickly to hardware capabilities that could not be exploited by existing hardware

architectures and software. At the same time, users were reaching the limits of the S/370; a sizeable VTAM network on OS/VS2-MVS would use up a full 16M-byte logical address space, while the combination of large "disk farms" and multi-CPU configurations were pushing the bounds of the I/O address space. I don't remember exactly how many different task forces we participated in; there were several passes taken at evaluating different virtual address space control architectures, system software designs, and I/O control architectures.

During this period, many of the inter-group rivalries were set aside or ignored, but we could not escape all of the organizational distinctions. We were able to work directly with the top people from the Poughkeepsie and Endicott system software groups, but there was little if any direct give-and-take between the software people and the primary hardware architecture groups. The hardware design versus software design issues were generally handled by an exchange of documents rather than working discussions. Not too surprisingly, the first version of the DAT architecture for the M/H machines looked like it was designed to run OS/VS2-MVS and not much of anything else.

We were given two questions to start with: "Can you build a VM system on this machine which runs S/370 virtual machines?", and "Can this architecture create a virtual M/H machine?" The first question came out as a qualified "Yes", but the second one was very tough to determine. I remember some active talks with Charlie Salisbury, Bob Adair, and others where we really couldn't decide if the software required to virtualize the M/H DAT architecture would work. The detailed hardware design was based on assumptions about the way it would be used—much like the S/370-135 timer microcode—and prototyping the entire system in our heads or on paper, in the context of a two- or three-week task force, was a major stretch even for the best of us.

After a couple of inconclusive attempts at the problem, another group was convened to come up with a better picture of the operating system which we wanted to run on the "big" machines. The necessary ingredients would include an interactive environment like CMS or TSO, a dedicated application environment like CICS/VS, compatibility for S/370 batch applications and utilities, S/370 virtual machine capability, and enough "native" virtual machine capability to support development, test and non-disruptive installation of software for the new machines. The usefulness of virtual machine technology was by that time well recognized within IBM. Some form of virtual machine support had been added to TSS/370 on an experimental basis, and there were rumors of a similar experiment in VS2-MVS. The first few rounds were not too productive, with an "exchange" of ideas something like these:

- "We'll just run MVS under VM."
- "No, we'll add one virtual machine to MVS and run VM in it."
- "Why do we need MVS? We'll run CICS naked under VM."
- "Forget VM, we'll run TSO under MVS."
- "Forget TSO, run CMS under MVS."

Eventually we settled down to some real work. The idea of the "M/H Control Program" or the MHCP was to apply the separation of function which worked so well for VM/370 to the larger problem of handling multiple address spaces with shared memory, in the style of MVS, in addition to well-defined virtual machines. (VM/370 was great at isolating virtual machines, but it was never very good at letting virtual machines work together. Virtual CTCAs, VMCF, and IUCV are all pretty clumsy and specialized.) Conceptually the idea was to isolate the low-level

I/O and system handling functions into a kernel much like VM-CP, but provide more than one "interface" into that kernel. One interface would be a regular virtual machine, but another would be a dramatic expansion of the DIAGNOSE-style enhanced interface, unconstrained by the hardware architecture definition.

The fairly short period of time we had to work on the high-level design limited the detail we could go into, but everyone involved thought that we could make it work. With the mechanics of system control and I/O device dependencies isolated in the MHCP kernel, it would have been theoretically possible to support new devices, and perhaps even new CPUs, much more easily and more quickly. By the same token the "intelligence" about how many address spaces to create and which areas of storage to share would remain in the MVS control realm. Faint echoes of the MHCP design, I suspect, led to the Group Control System (GCS) component. The original MHCP notion made a lot of converts in the Burlington VM group and in the Poughkeepsie groups, both technical and marketing/strategic. I can't articulate many of the ideas very well, but we were all convinced that there was a real opportunity.

Our timing was less than perfect unfortunately. Putting the MHCP together would have required a lot of work and a massive coordination effort between several different organizations. The combination of expense and earnings pressure, and the fact that we were (unknowingly) on the eve of the Burlington shutdown, probably contributed to the "not yet" decision. Various pieces of the advanced architecture have since been released as "XA", but I haven't worked with IBM systems closely enough in the past ten years to know what did make it and what didn't. It's always a bit of a thrill to see something which you've worked on or contributed to announced as a product even if it takes a while to surface.

Conclusion

Dave Tuttle, CS February, 1970

Somehow I get a funny feeling when you refer to my "memoirs"—those are written by old people, aren't they? I have been doing computer system development since early in 1967, but I won't be even 41 until next October.

—Dave Tuttle
—July, 1989

Addendum

One of the things which I can probably clear up for good is my earlier confusion between Jim Cannavino and Jim Cavet. Jim Cavet was indeed the SNA/Data Comms training person, working out of Kingston, and we did share several evenings trading tales on the road and at least one very pleasant dinner at his home in Kingston. Jim Cavet might be the one who started with IBM as an 029 keypunch CE in the Chicago or Detroit area and told me a wonderful tale about getting his first manager in trouble by not cashing six months worth of paychecks... but the tone of the tale fits Jim Cannavino just as well if not better.

Jim Cannavino, on the other hand, I met in the early part of 1973 while he was in charge of some aspect of SDD Development (probably related to OS/VS2-MVS, but I'm not too clear on the details) in Poughkeepsie. He was involved in trying to put together a network of the many different VM/370 systems which were being used within SDD to develop OS/VS2. It was typically ironic that the VS2 marketing people were our worst enemies, but the VS2 developers had bet their schedule on using VM's capabilities.

Through some negotiation that I know very little about, Jim had gotten a couple of days of my time made available to work on a version of CP2780 which had been modified, not yet successfully, to provide file transfers between VM systems. The time was late January or early February of 1973. VM/370 had been out in the field for a couple of months. The VM/370 group had transferred from DPD to SDD (January 1, 1973). I had moved from the CP Development Group to the Advanced Development Group, but I was still doing some bug-fixing (major, as it turned out) on the DMKVAT code, and I was working on the DMKCTC virtual CTCA support

The first trip down to Poughkeepsie was something of an adventure. The initial plan was for me to go down on Wednesday night work Thursday and Friday, then come back Friday night—I had some crucial personal plans for the weekend. Best-laid plans fell afoul of New England weather—I couldn't get out of Logan until Thursday night. I took Command Airways ("IBM Airlines") from Boston to Poughkeepsie, rented a car, and somehow found my way over to the big Building 701-705 development complex. I'm pretty sure it was my first visit to Poughkeepsie, and I arrived well after normal hours (9:00?). I met Jim there, and he gave me a quick briefing on what they were trying to do, showed me the current state of the code, and gave me a tour of the two-acre machine room at the center of Building 701.

Such strange circumstances beget other strange events; I ran across Bob Wright, an acquaintance from my days as a computer hacker in the MIT Computation Center, running stand-alone on a system in the back comer of the machine room. I also noticed, in passing, a very frustrated programmer trying to get some VS2 testing done in a VM virtual machine. Coincidence, again; he was experiencing a serious bug in the virtual DAT support which showed up only with VS2, and I had—earlier the same day!—just found out what it was and how to get around it. We didn't have a load map of his CP, but I was able to find DMKVAT by going through the Program Interrupt New PSW; I worked up an in-core patch, put it in via STCP, and had him up and running in less than 20 minutes. The fix worked fine, and I remembered it well enough to write it down for him from memory, when the VM system went down for some other reason an hour or so later. (Simple to do, in fact, but I may not have been the only one to tell tales about it)

[On with the main story.] Those of us who remember CP2780 know that it was a truly worthy example of "spaghetti code". Very few people, even then, knew enough about BSC communications to write a reasonably structured line I/O handler. It was so badly organized, in fact, that I thought I had to straighten it out before I could put in the extra features that Jim Cannavino was looking for. Ever the optimist, as a cocky young engineer, I set about doing just

that. Time was not freely available, however; the full day lost to weather couldn't be recovered. Jim was pushing me to stay and work the weekend, because he was not successful in getting more of my time through official channels. My personal plans were not very flexible; that weekend I was scheduled to take my *fiancee* (Susan, my first wife) to upstate New York to meet her parents for the first time. Little did I appreciate the skills of a wheeler-dealer such as Jim, in the pursuit of his goals.

Through some hidden magic with credit cards and expense accounts, Jim arranged the following scenario: I would stay in Poughkeepsie through Saturday, working on the modified CP2780. Susan, on Sunday, would take a pre-paid taxi to Logan Airport, pick up a pre-paid airline ticket, and fly from Boston to Albany, New York. With the rental car that I was supposed to return to the Poughkeepsie airport on Friday, I would drive to Albany, pick up Susan at the airport, then continue on to Syracuse to meet her family. *Mirabile dictu,* this all came to pass. On Monday, a holiday, Susan and I drove back to Boston in the rental car, which I subsequently turned in at Logan Airport. The AVIS Wizard choked, coughed, and spit up a mild note saying, "Return Station Other than Expected"—off by about 200 miles and three extra days.

The magic with expense accounts came next. Jim had told me to put the rental car on a separate expense form and send it to him. That was fine, and I got reimbursed in the normal course of events. However, our regular expense mechanism couldn't handle the fact that I flew down to Poughkeepsie but didn't (visibly) return; they made me add in 210 miles of personal car expense to explain the return trip. While Jim was hatching this scheme, he told me some tales and tricks about expense account handling from the time when he was an SE or CE working out of New York City. (After a holiday-party-interrupting crisis trip got Jim and a co-worker stranded in a hip-deep snow storm in Connecticut, he managed to hide the cost of a ruined $750 cashmere overcoat in a three-day expense report, even though his manager knew it was there, somewhere...)

There was a second trip down to Poughkeepsie on the same mission, but it was an unauthorized one. Jim successfully hooked my "savior" image, and I drove down on a Friday night to work the weekend. As luck would have it, of course, I got snowed in and didn't get back to work until the following Wednesday. This led, inexorably, to one of the best straight lines, from my manager, which I have ever had to pass up. The Advanced Development Group, at the time, was officially being managed by Ken Brooke. The people in the group, however, included both myself, Dick Newson, and some others who were not very manageable. Ken was generally more concerned about the appearance of control than he was adept at handling the technical issues. When the weather kept me in Poughkeepsie for two extra days, I had to call to tell Ken where I was, and I earned a tongue-lashing when I finally got back to the office. Ken was apparently the one who had refused more help for Jim's project, and he came out with the line, "Why do you think I'm here, just to sign your paychecks?" At the time, it was very hard to resist answering his question.

The sad part of the story is that I never did get to finish the cleaned-up, networking version of CP2780. We might have had a VM/370 network of some sort much sooner than we eventually did. The other "unfinished symphony" from my early VM-CP days was a ground-up rewrite of DMKIOS; I worked out the I/O initiation path, but I couldn't free up enough time to figure out a clean interrupt service structure. My fast-path version could take an IOBLOK from DMKIOS entry to the actual Start I/O in about 55 instructions, if the requested path was free.

Ah, the good old days....

—Dave Tuttle
—February, 1991