For: Interface, May/June 1974

CP-V DESIGN

An interview with five members of the Xerox programming development team

It has been said that operating systems -- the control programs that oversee the running of computers with a minimum of human intervention -- are between 30 and 100 times more complicated than the hardware they run on. Yet this complexity is largely transparent to the user, whose work they simplify by many orders of magnitude. To make this modern miracle happen is the everyday job of the Xerox programming development team headed by Al Bongarzone. In this interview, Bongarzone and four members of his staff -- Ed Bryan, Peter Heinrich, Gene Kinney, and Dick Litschgi -- share some insights on the Xerox Control Program - Five (CP-V) operating system: What it is and what it does.

Q: In order to show what kind of an operating system CP-V is, could we begin by relating it to the hardware it runs on?

Heinrich: I think a key point here is how thoroughly integrated the hardware and software are. You've not only got to have good hardware, but to use that hardware properly all the time is a pretty tricky business, and takes a substantial effort in software. And here's where the fact that this product has been developed over a long period of time pays off handsomely. We've had a lot of opportunities to understand how both the software and the hardware really work.

Bongarzone: Right. Because CP-V is based on our UTS operating system, which has been in the field since December 1970. When we brought out the first release of CP-V last August, we already had in-depth knowledge of the performance of the operating system, and how it utilizes the key Sigma and 560 hardware features.

Litschgi: Of course, another result of our building CP-V on the base of UTS is that all language processors and user programs that run under UTS are compatible with CP-V.

Bryan: But it's interesting that in each of CP-V's five basic operating modes -- timesharing, batch, remote processing, realtime, and transaction processing -- we're taking advantage, today, of key hardware and software decisions we made back in the 1960s.

Q: What are some of these key hardware features?

Bryan: Certainly the memory map is one of the most important.

Q: What is the memory map?

Bryan: In our Xerox 560 and Sigma 6,7, and 9 computers -- which, incidentally, are the ones for which CP-V was designed -- we have a hardware function that allows CP-V to make very efficient use of core memory, using up all the odds and ends resulting from the going and coming of the user jobs. The memory map puts this fragmented real core back together so that to the user it appears as a single contiguous "virtual" program space. Furthermore, each and every program may use the same virtual space -- CP-V just reloads the map to change from one program to another. Some systems use software overhead to do the job. To illustrate one effect of this, we can get the same things done with, let's say, 128,000 words of memory that competitive systems -- using software overhead to manage the memory job -- would need a megabyte or even two megabytes to do.

Kinney: The point is, there's no wasted core in a CP-V system.

Bryan: Another key hardware element -- essential for high-performance timesharing -- is our high-speed RAD rotating memory. This fixed-head disk storage system provides us with a virtual extension of memory in secondary storage. Actually, by one definition of "virtual memory", the whole high-speed swapping RAD is an extension of the core memory. And the very high speed with which the RAD swaps programs in and out of core makes it possible for us to have a very large virtual-to-real ratio in the sense that IBM talks about it.

Q: Can CP-V run on a removable-disk storage system in place of a RAD?

Kinney: You can do that, but not as a high-performance system. But a disk-pack system does provide a budget-entry vehicle -- to allow a customer to get started on CP-V. As he grows, and takes advantage of more of the many CP-V features, he'll eventually want to use the higher speed RAD for swapping.

Bongarzone: Another aspect of the system architecture that we exploit is the multi-port memory. The fact that the system has an extremely high bandwidth makes it possible for us to perform better than much of the competition in the same price class, whose machines are still cycle-stealing -- that is, performing I/O through the CPU. With our independent ports to memory, we can perform I/O simultaneously with high-speed processing.

Litschgi: Actually we can look at both the Sigma and 560 computers as having a multiprocessing architecture with the IOPs operating essentially independent of the CPU, controlling the peripherals. The 560 has gone a step beyond Sigma on this. What it lets us do is drive higher performance peripherals than before, while working the CPU all the time.

Heinrich: And, given this kind of architecture, it's up to us software designers to figure out how to keep all parts of the system as busy as possible. Because an operating system is so complex, there is always a lot of room for improvement. In release after release, we've been able to get more and more bang out of the system. As an example of this, we were initially running CP-V with "wait times" (that is, when the CPU has to wait on the peripherals and can't be used by anyone) as high as 25 percent. In a subsequent release it came down to around 15 percent. And now, running the recent B00 release of CP-V on our timesharing system, here in El Segundo, we often see wait times as small as 1 percent. Which means we now routinely deliver between 90 and 95 percent of the total system capacity to the user.

Bongarzone: An additional hardware feature -- one that makes it possible for us to incorporate realtime as one of the five modes of the operation of CP-V -- is the interrupt structure.

Q: That differentiates us from a lot of the competition, doesn't it?

Bongarzone: Yes, both in terms of the number of interrupts that we allow, and the fast context switching when responding to interrupts through use of extra register blocks for example. The hardware capability is there, and we've used it to incorporate realtime processing, as a standard CP-V feature in our B00 release. That, incidentally, is the fourth of CP-V's five operating modes to be implemented and released to the field. The fifth, transaction processing, will be released in the fourth quarter of this year.

Q: What are modes one, two, and three?

Bongarzone: They were available with the initial release of CP-V last August: single-stream and multiprogrammed batch; timesharing; and the remote processing mode, including intelligent remote batch.

Heinrich: That last one is especially interesting. What this allows you to do is talk computer-to-computer. You can link up a small system to a CP-V, or two CP-V systems, or a CP-V to another large system. The University of Saskatchewan, for example, has their CP-V system talking to an IBM 370/158 in another city.

Bongarzone: We have an increasing number of customers doing that sort of thing today.

Q: What is there about the CP-V software -- its design, or architecture -- that makes all this multi-mode activity possible?

Kinney: One of the most important reasons is an "integration" concept. It's basic to the CP-V design, and means that the operating system treats all jobs that come in very much the same, regardless of where they originate. Whether they're working in batch mode, timesharing, remote batch, transaction processing, or whatever, users all build the same files, use the same processors, the same services of the operating system ....

Heinrich: Its also important in achieving the system efficiency I mentioned before -- because of the uniformity of the jobs CP-V has more choices and therefore can better adapt to changing load conditions.

Bryan: It's hard to over-emphasize this business of treating all programs alike, regardless of what environment they're operating in -- the same kinds of programs run in all modes, and that's rather unusual in an operating system. It's one of the things we tried to go after in the early design days, and it was simply not what people at that time were doing when they designed operating systems. They were designing the different operating modes as distinct entities -- separate and not even always equal in the same machine -- and we were trying to make them all the same, so that problem programs could float between one environment and another without the need for re-programming or file conversion.

Q: Why is it important to be able to do that?

Litschgi: For obvious reasons, programmers like to develop their programs on-line and debug them interactively. (Their bosses like it too because the job gets done faster.) Once checked out, the unaltered program is turned over to operations for periodic production runs in a batch or remote processing mode. This flexibility in CP-V allows the user to achieve the full benefits of each mode.

Bryan: One other point, too. An elegant aspect of CP-V that's based on that key integration concept is that the system has set up default access to files and peripherals and so on, so that programs are automatically connected to the peripheral devices appropriate to the mode of execution. That is, if a batch job calls a program, the output will go to the line printer while if the same program is called on-line then the output will be directed to the user terminal -- all without need for special instructions from the programmer.

Heinrich: But it's not just the user that benefits. The fact of having an integrated operating system has made it much easier for us to add new modes of operation. Without that key design decision, way back in the beginning, it would have been impossible for us to develop new capabilities for the system as we've done -- and we haven't yet reached the end of the potential.

Q: What do you see in the way of untapped potential in CP-V?

Bryan: Transaction processing is, of course, the potential we're tapping most immediately. In the short-term,efforts are underway to increase the reliability of the file system, and to enhance its capability to handle very large data bases. And I'd say two of the major areas of untapped potential are inter-machine communications and multi-processing.

Bongarzone: I think that in the computer business there's been one constant, and that is that those of us who build and deliver systems really don't understand their full potential -- it's the customer who constantly teaches us new ways to use these machines. He continually shows us ways to improve them, to get more productivity out of them, to apply them in new and innovative ways. Really, the best product planners, the best designers we have are our customers. What we must do is continue our exchange with them, to learn what they're doing and to identify their needs -- and more times than not, we find they're using the system in a way we never envisioned.