8

Computers in Mission Control

Mission control begins when launch processing ends. At the point a missile is committed to flight—as when the Shuttle solid rockets are fired or a liquid-fueled booster rises an inch off the pad—responsibility for monitoring and control of the spacecraft shifts from the launch director and his crew to the flight director's team. Three major tasks occupy the flight controllers: sampling the telemetry stream to make certain everything is going well and to collect science data, doing navigation calculations, and sending commands. Manned and unmanned spacecraft require this support, with manned spacecraft having the advantage of carrying observers and decision makers to supplement what can be done from the ground. To successfully support both types of missions, digital computers must operate on massive amounts of data in real time. Mission control tasks are beyond the abilities of humans alone.

Mission control centers and their equipment are located far from the launch site. NASA's manned mission control began in 1961 with Project Mercury at the Cape Canaveral launch area, but its computers were at Goddard Space Flight Center near Washington, D.C. Since 1964, early in the Gemini program, both computers and controllers have been housed in Building 30 at the Johnson Space Center in Houston. NASA's unmanned near-earth missions are controlled mostly from Goddard, with most deep space missions handled through the Jet Propulsion Laboratory's (JPL) Spaceflight Operations Facility in Pasadena, California.

In addition to control centers, mission support requires numerous tracking stations to collect and format telemetry and radar data to help in monitoring and navigation and to transmit commands. These widely scattered stations and the control centers are linked together by the NASA Communications Network (NASCOM), headquartered at Goddard. The Space Tracking and Data Acquisition Network (STADAN), used to specialize in unmanned spacecraft but, having combined with the Manned Spaceflight Network (MSFN) in 1972, has become the general network. When all the specified Tracking and Data Relay Satellites are in place, they will take over much of the manned flight communications, yet tracking is still a STADAN responsibility. Lunar and planetary probes are the venue of the Deep Space Network, which operates three main stations at Goldstone, California, Madrid, Spain, and Canberra, Australia, each with a variety of antennas ranging up to 64 meters in diameter. The Deep Space Network helped with manned lunar missions when the Apollo spacecraft passed a distance of 10,000 miles from earth*.

In contrast with on-board computers, computer systems used in control centers and tracking stations have primarily consisted of off-the-shelf equipment. NASA could take this approach to procurement because, so far, adequate processing power to achieve mission objectives has been available in commercial systems. When mission control began in the late 1950s and early 1960s, software technology had not reached the necessary level of sophistication. The prime contractor had to develop completely new operating system software for the Vanguard, Mercury, and Gemini programs, but was able to incorporate large chunks of existing operating systems into those used for Apollo and Shuttle, as well as some later deep space missions. This was possible in part because experience and techniques learned from designing the original operating systems were used in new commercial products.

MANNED MISSION CONTROL COMPUTERS

As with manned spacecraft on-board computers, computer systems used in manned mission control are more sophisticated and larger than those used for unmanned missions. Even though unmanned satellites and space probes pioneered the use of computers in mission control, the need for quick response and redundancy, the inherent complexity of manned spaceflight, and the rigors of the race to the moon forced rapid improvements and innovations in systems used in manned mission control so that they surpassed the older systems.

The story of computers in manned mission control is largely the story of a close and mutually beneficial partnership between NASA and IBM. There are many instances of IBM support of the space program, but in no other case have the results been as directly applicable to its commercial product line. When Project Vanguard and later NASA approached IBM with the requirements for computers to do telemetry monitoring, trajectory calculations, and commanding, IBM found a market for its largest computers and a vehicle for developing ways of creating software to control multiple programs executing at once,

---

*For the story of the tracking and communication networks, see William R. Corliss, *Histories of the Space Tracking and Data Acquisition Network (STADAN), the Manned Space Flight Network (MSFN), and the NASA Communications Network (NASCOM)*, NASA CR-140390, June, 1974, and N.A. Renzetti, ed., *A History of the Deep Space Network From Inception to January 1, 1969*, Jet Propulsion Laboratory TR 32-1533, September 1, 1971. Each has considerable detail about the technical developments involved, including the decision to use computers at stations.

capable of accepting and handling asynchronous data, and of running reliably in real time. These things the company was able to do quite successfully, and the groups it assigned to the job impressed their NASA counterparts. When asked about IBM's performance in this field, one NASA manager said without hesitation, "IBM is the best"[1].

The company maintained its lock on mission control contracts through Gemini, Apollo, and the Shuttle. At each point, some experienced personnel were transferred to other parts of the company to share lessons learned. Several individuals contributed to OS/360, the first multiprogramming system made commercially available by IBM[2]. One became head of the personal computer division[3]. NASA also used successful managers from mission control work to help other programs. Howard W. "Bill" Tindall started with Mercury and Gemini ground software and later made a significant contribution to the quality of the Apollo on-board software. No other software system developed under NASA contract in the 1960s was as well thought out and executed as manned mission control.

Beginnings: Vanguard and Mercury

America's most spectacular contribution to the International Geophysical Year (1957–1958) was the Vanguard earth satellite, which, in ignorance of Russian preparations, was thought to be the world's first orbiting spacecraft. In June of 1957, Project Vanguard established a Real-Time Computing Center (RTCC) on Pennsylvania Avenue in Washington, D.C, consisting of an IBM 704 computer[4]. The 40,000-instruction computer program developed for Vanguard did data reduction and orbit determination[5]. Orbit calculations needed to be done in real time so that ground stations could be warned of the approach of the satellite in time to listen for its signals and know where in space the data came from. Thus, IBM gained early practical training in the primary skills needed for mission control. In 1959, when NASA was ready to contract for a control center for Project Mercury, IBM had experience it could point to in its proposal, as well as an existing computer system about to be freed from Vanguard work.

NASA awarded Western Electric the overall contract for the tracking and ground systems to be used in Project Mercury on July 30, 1959[6]. By late 1959, IBM received the subcontract for computers and software[7]. Washington remained the site for the computer system because it could benefit from centralized communications already in existence[8]. NASA founded Goddard Space Flight Center the next year, and since it was less than half an hour from downtown Washington, the same advantages would accrue from locating the computers there.

Combined NASA and IBM teams used the old computer system downtown until about November 1960, when the first of Mercury's new 7090 mainframe computers was ready for use at Goddard. James Stokes of NASA remembers the first time he and Bill Tindall went to the new computer center, they had to cross a muddy parking lot to where a "building" with plywood walls, window air conditioners, and a canvas top confounded the IBM engineers who were trying to keep the system up and running under field conditions[9]. That structure evolved to become Building Three of the new Space Flight Center and housed the system through the Mercury era[10].

IBM's 7090 mainframe computer was the heart of the Mercury control network. In 1959, the DOD issued a challenge to the computer industry in the form of specifications for a machine to handle data generated by the new Ballistic Missile Early Warning System (BMEWS). The 7090 was IBM's response. Essentially an improvement of the 700-series machines like the one being used as a development machine for Mercury, the 7090 adapted the new concept of I/O channels pioneered in the 709 and was so large that it needed up to three small 1410 computers just to control the input and output. The DOD's needs for BMEWS closely paralleled those of Mercury in terms of data handling and tracking. Thus, IBM was in a good position with its hardware.

To provide the reliability needed for manned flights, the primary Mercury configuration included 7090s operating in parallel, each receiving inputs, but with just one permitted to transmit output. Called the Mission Operational Computer and Dynamic Standby Computer, the names stuck through the Apollo program. This was NASA's first redundant computer system. Switching from the prime computer to the Dynamic Standby was by manual switch, so it was a human decision[11]. During John Glenn's orbital mission, the prime computer failed for 3 minutes, proving the need for active standby[12].

Three other computers completed the Mercury network. One was a 709 dedicated to continuously predicting the impact points of missiles launched from Cape Canaveral. It provided data needed by the range safety officer to decide whether to abort a mission during the powered flight phase and, if aborted, information about the landing site for the recovery forces. Another 709 was at the Bermuda tracking station with the same responsibilities as the pair at Goddard. In case of a communications failure or double mainframe failure it would become the prime mission computer. Lastly, a Burroughs-GE guidance computer radio-guided the Atlas missile during ascent to orbit[13].

Locating the computers near Washington while placing the mission control personnel at Cape Canaveral led to a communications problem that resulted in a unique solution. In early digital computers, all input data went to memory by way of the CPU. Large amounts of data that needed to be accepted in a short time often backed up, waiting for the central processor to handle the flow. A solution is direct memory access, which sends data directly from input devices into storage. Transfers of large blocks of data directly to memory are conducted through data channels, first used by IBM on its 709 and then on the 7090. By using channels, processing could continue while I/O occurred, increasing the overall throughput of the system. Mercury's 7090s were four-channel systems. Normally, the peripherals handling input and output would be connected to the channels physically close to the machine, but the peripherals (plotters and printers) driven by the Mercury computers would be about 1,000 miles away in Florida. The solution was to replace Channel F of the 7090 with an IBM 7281 I Data Communications Channel, a device originally created for Mercury that has had great impact on data processing[14].

Four subchannels divided the data handled by the 7281 device. One was an input from the Burroughs-GE guidance computer to provide data used in calculating the trajectory during powered flight. The second input radar data for trajectory and orbit determination. Two output subchannels drove the displays in Cape Canaveral's Mercury Control Center and locally at Goddard[15].

Connecting the two ends of the system was a land line allowing transmission at 1,000 bits per second[16]. Although this was a phenomenal rate for its time, now a simple microcomputer routinely transmits at 1,200 bits per second on nondedicated public telephone lines. The distance and newness of the equipment occasionally caused problems. Once in a while during a countdown, data such as the liftoff indicator, which was a single bit, would get garbled and give erroneous signals[17]. Most times such flags could be checked by other sources of information, such as radar data contradicting the lift-off message. Also, up to a 2-second time lag on the displays in the control center was common[18]. During powered flight, such delays could be significant; thus, the need for a separate impact prediction computer and another machine in Bermuda.

Software development for the Mercury program was another area in which IBM advanced the state of the art[19]. In the beginning of the computer era, operators ran programs on computers one at a time. Each program was assigned peripherals, loaded, run and, if errors occurred, stopped individually. As machines grew larger and the number of users increased, some way of making the process of loading and executing programs more efficient was needed. The result was the concept of "batch" processing, in which a set of several programs could be loaded as a unit and executed in sequence. A special control program called a "monitor" watched for errors and aborted programs trapped in loops or that spun off into corners. To handle the many jobs needed by manned spacecraft mission control, IBM set up a method for programs to be interrupted and suspended while other programs of greater priority ran, and then resumed when the high-priority jobs ended. Thus, a number of programs could be loaded into the machine land run, giving the illusion of simultaneous execution, even though only one had the resources of the central processor at any one time. This was the only way the processing of radar data, telemetry, and spacecraft commands could be accomplished in the split seconds of time allotted.

IBM called the control program the Mercury Monitor, but that is a misnomer in that it superceded the capabilities of the known monitors of the time. It was event driven, which means that certain flight events (lift-off, sustainer engine cutoff, retrofire) formed the basis of the starting times of certain processes[20]. The Mercury Programming System's primary functions included capsule position determination, retrofire time calculation, warning ground stations of the acquisition times, and impact prediction after retrofire. Three separate groups of processing programs, each stored on tape until needed, did these functions at different times: launch, orbit, and re-entry[21]. No matter which group of processors was loaded into the machine, the Monitor frequently checked a table listing processes waiting for input or output. Software placed entries in the table when the Data Communications Channel signaled that data were ready to be transferred[22]. The Monitor then handled the requests in priority order. Within a processor group, such as orbit, a set of different single-function processors would be defined. Thus, the entire mission control program was highly modular, allowing easier maintenance and change. In fact, some modules from the Vanguard programs could be adapted to Mercury use.

NASA wanted to take over the software as soon as possible, so 15 or so civil service employees were assigned to the IBM group while it was still in downtown Washington. However, the Space Task Group retained direct control over the software development, a somewhat frustrating situation for NASA engineers much closer to the actual project and in a better position to make suggestions[23]. At the time, NASA saw its role as that of a knowledgeable user and recognized it lacked the expertise to handle some of the calculating tasks involved. James Stokes, a NASA engineer, admitted that "we didn't know enough to specify the requirements" for the software[24]. IBM was not much better off and acquired its expertise by contracting for the services of Dr. Paul Herget, then director of the Cincinnati Observatory, who had privately published a book on orbit determination in 1948[25].

The Mercury network provided continuous height, velocity, flight path angle, retrofire time, and impact points. During powered flight the main computer center, the Cape impact prediction computer, and the Bermuda tracking station computer all would give GO/NO GO recommendations to the flight director. After engine shutdown, the system needed to give GO/NO GO data within 10 seconds, so that a safe recovery could be effected if orbit had not been reached. During the orbital cruise, the astronaut could be given updated retrofire times each time he came in contact with a ground station[26].

As the Mercury program wound down during 1962 and NASA began to accelerate preparations for Gemini and Apollo, the Agency decided to place both the computers and flight controllers for manned spaceflight mission control in a combined center in Houston. Goddard staff proceeded under the assumption that the new control center would not be ready in time for the first Gemini flights, which turned out to be correct. Gemini I, II, and III used Goddard as the prime computer center, with the new system in Houston acting in an active backup role for flight three. Beginning with flight four, the second manned mission, Houston took over as prime, with Goddard acting as the backup throughout the Gemini program[27].

For IBM and NASA, the development of the Mercury control center and the network was highly profitable. IBM's Mercury Monitor and Data Communications Channel were the first of their types[28]. Future multitasking and priority interrupt operating systems and control programs owed their origins to the Monitor. Large central computers with widely scattered terminals, such as airline reservation systems, have their basis in the distant communications between Washington and a launch site in Florida. For both organizations, the experience gained by staff engineers and managers directly contributed to the success of Gemini and Apollo.

Second System: The Gemini–Apollo RTCC

Before the first Mercury orbital flight was off the ground, NASA engineers working on mission control tried to influence the design of the new center in Houston. Bill Tindall, who worked on ground control for NASA from the beginning, realized that locating the Space Task Group management at Langley Research Center, the computers and programmers at Goddard, and the flight controllers at Cape Canaveral created serious communication and efficiency problems. In January 1962, he began a memo campaign to consolidate all components at one site, obviously the new Manned Spacecraft Center[29]. On February 28, just 8 days after John Glenn's flight, Tindall made his strongest case in a detailed essay in which he noted that IBM was the only company capable of creating real-time software. He wanted the Ground Systems Project Office, then in charge of oversight of the RTCC development, to allow representatives from the Flight Operations Division to assist in mission programming[30]. As the eventual users of the system, it made sense to include them.

Figure 8–1. IBM 7094s in the Gemini Real Time Computer Complex. (IBM photo)

In April, the Western Development Laboratories of Ford's subsidiary Philco Corporation began a study of the requirements for the new mission control center. One aspect of the study was to take numeric data and give it pictorial content, making the jobs of the flight controllers less hectic but necessitating much more sophisticated computer equipment[31]. As Philco worked through the summer, NASA Administrator James Webb announced on July 20 that there would be an expanded replacement for Mercury Control. A "request for proposal" was prepared, including concepts developed by Philco and documented by them in their final facilities design released on September 7.

Philco's design was broad in scope, covering physical facilities, information flow, displays, reliability studies, computers, and even software standards. Philco specified that modularity in program development was a must, as it would ease maintenance and allow the use of "lower caliber" people to code subprograms, leaving the real stars to do the executive software[32]. This organizational rule became standard for large program projects. Another specification required that the probability of successful real-time computer support for a 336-hour mission be 0.9995. Also, due to rendezvous plans for Gemini and the dual-spacecraft Apollo lunar missions, the center had to control two spacecraft at one time. To meet the reliability and processing goals, Philco examined existing computer systems from IBM, UNIVAC, and Control Data Corporation, as well as its own Philco 211 and 212 computers, to determine what type and how many would be needed. The calculations resulted in three possible configurations: five IBM 7094s (the immediate successor to the 7090, essentially a faster machine with a better operating system, IBSYS); nine UNIVAC 1107s, IBM 7090s, or Philco 211s; or four Philco 212s or CDC 3600s[33]. No matter which group would be chosen, it was obvious that the complexity of the Gemini–Apollo Center would be much higher than its two-computer predecessor. To help keep the system as inexpensive and simple as possible, NASA specified to potential bidders that off-the-shelf hardware was essential.

IBM moved quickly to respond to NASA's call for proposals, delivering in September a 2-inch thick, three-ring binder full of hardware and software bids, including a detailed list of personnel they would commit to the project, complete with employment histories. Although the company knew it was the leading candidate (Tindall's endorsement could hardly have escaped notice), it carefully matched the specifications, such as clearly stating that modularization and unit testing would be the norm in software development. One area in which they differed from Philco's calculations was the number of machines needed. Perhaps to keep the total bid low, IBM proposed a group of three 7094 computers. By splitting the software into a Mission Computer Program and a Simulation Computer Program, one machine could run the Mission Program as prime, another run it as the dynamic backup, and the third run the simulation software to test the other two, thus fulfilling requirements for redundancy and preflight training and testing. This forced IBM to explain its way around the 0.9995 reliability requirement. Three machines yielded reliability of 0.9712, slightly over four being needed to achieve the specification (thus, Philco's suggested number of five). IBM made a case that the reliability figures were misleading and that during so-called "mission-critical" phases the reliability of three machines would exceed 0.9995[34].

Eighteen companies bid on the RTCC, including such powerful competitors as RCA, Lockheed, North American Aviation, Computer Sciences Corporation, Hughes, TRW, and ITT. NASA assigned Chistopher Kraft, the eventual chief user, to chair the source board that studied the responses to the request for proposal. Tindall served also, with James Stroup, John P. Mayer, and Arthur Garrison, all of the Manned Spacecraft Center. They awarded the original contract NAS 9-996, covering the Gemini program, to IBM on October 15. Worth $36 million, it was to run until the end of August 1965. Extended to December 1966, the total cost came to $46 million[35].

With 6 weeks of preparation already done before the contract award, IBM's core of engineers were ready for business in Houston by October 28. J. E. Hamlin started as project manager and interim head of systems engineering. He had 12 years of IBM experience, first as a hardware engineer, later as a group leader for SAGE software, and then manager for the Mercury system implementation. He had barely started work at JPL's Deep Space Instrumentation Facility when the RTCC contract came up. In his first report in January 1963, he was able to announce the arrival of the first 7094 to be used for software development. The computer and, later, two others were installed in an interim facility on the Gulf Freeway. Each started with 32K words of memory and 98K words of auxiliary core storage, with a 1401 as a front end for input and output[36]. On the negative side, Hamlin's early projection of a peak staff of 161 had leaped to 228 by the time of the first report. Eventually, 608 IBM people worked simultaneously on the project, with 400 of them on software development. The magnitude of the task was greatly underestimated both by IBM, which made the bid, and NASA, which accepted it.

Hardware needs grew along with the staff. The original three machines moved from the interim center to Building 30 at the Manned Spacecraft Center. Two more were added, fulfilling Philco's prophecy. The size and rating of the machines was also increased to model 7094-IIs with 65,000 words of main core storage and 524,000 words of additional core as a fast auxiliary memory[37]. In the new configuration, one machine was the Mission Operational Computer, the second, the Dynamic Standby Computer, and the third, the Simulation Operations Computer as before, with the two new ones used as the Ground System Simulation Computer and a standby for future software development. The Ground System simulator acted like the tracking network and other ground-based parts of mission control to test software.

IBM's original proposal projected completion of the new system within 18 months. As time passed and problems occurred, the plan altered to begin with support of the Gemini VI mission. But slips in Gemini and steady progress on the software enabled the use of the Center for passive parallel computations during the Gemini II unmanned flight on December 9, 1964, just under 26 months after the contract award. On Gemini III, the Houston control center did its final test as an active backup. The results were so promising that from Gemini IV on, mission control shifted from the Cape to Houston.

Gemini Ground Software Development

NASA's requirements for the Gemini mission control software resulted in one of the largest computer programs in history. In addition to all the needs of the Mercury system, Gemini's proposed rendezvous and orbit change operations caused a near-exponential increase in the complexity of the trajectory and orbit determination software. Placing a computer on board the spacecraft made it necessary to parallel its computations as a backup and also necessary to devise a way to use the ground computer system to update the Gemini flight computer. Also, by the time the Gemini program matured, all data on the tracking network were in digital form, and thus computable, so the amount of data that passed through the ground system increased further[38].

IBM reacted to the increased complexity in several ways. Besides adding more manpower, the company enforced a strict set of software development standards. These standards were so successful that IBM adopted them companywide at a time when the key commercial software systems that would carry the mainframe line of computers into the 1970s were under construction[39]. IBM approached the more difficult areas by acquiring the services of specialist consultants and sponsored a group of 10 scientists pursuing solutions to problems in orbital mechanics. It included Paul Herget and some men from IBM's Cambridge, Massachusetts "think tank"[40].

Key to the flight system was the Mission Computer Program. It centered on a control program called the Executive, which took over the functions of the Mercury Monitor. Under the Executive, three main subprograms operated in sequence. NETCHECK performed automatic tests of equipment and data flow throughout the entire Manned Spaceflight Network, certifying it ready for the launch of the spacecraft. It succeeded the CADFISS (Computation and Data Flow Integrated Subsystem) program used in Mercury[41]. ANALYZER did postflight data reduction. However, the Mission Operations Program System remained the heart of the software, responsible for all mission operations, such as trajectory calculations, telemetry, spacecraft environment, backup of the on-board computer, and rendezvous calculations. It divided into a number of modules: Agena launch, Gemini launch, orbit, trajectory determination, mission planning, telemetry, digital commands, and re-entry, with several subprograms within each section[42]. Each subprogram was highly sophisticated and very powerful. The re-entry program, for example, could calculate retrofire times 22 orbits in advance[43].

IBM found it impossible to complete this complicated system with the tools used in the Mercury program. All of the Mercury control software was in assembly language. Aside from the assembler, software tools were minimal, reflecting the state of the art circa 1960. Partly inspired by the difficulties of developing a large system such as Mercury and SAGE and partly to help commercial customers creating new software to match the size and capabilities of the new line of mainframe computers, IBM provided a much better set of tools with its 7094 series machines than with earlier models. A fairly robust operating system, IBSYS, could be used with the 7094, and a modification of it gave the Gemini software developers a decent editor and compilation tools for high-level languages. Called the Compiler Operating System, it included a combination FORTRAN/Mercury compiler called GAC (for Gemini–Apollo Compiler), making it possible to do some programming in FORTRAN. The Mercury compiler contained all the functions of SOS, the Share Operating System, which was IBM's standard system of the late 1950s and the predecessor to IBSYS[44].

Besides using better tools, the Gemini programmers tried to keep the architecture simple and changeable. Using process control tables was an important design decision, as they could be changed to fit different mission requirements with some ease and without disturbing software in place. Their use continued throughout the Apollo and Shuttle programs[45]. The Executive was a further refinement of the real-time control program first approached in Mercury. A relatively spare 13,000 words in size, the Executive provided priority-based multiprogramming. It could transfer needed data to supervisory routines which, in turn, started processes[46]. At the lowest level, contention between cyclic processes and demand processes characterized the RTCC[47]. Its obvious success helped form NASA's ideas of what a good real-time operating system should be, which later influenced the nature of the operating system on board the Shuttle. NASA personnel were close to the Gemini–Apollo ground system development, sometimes defining test cases and duplicating programs to check whether requirements had been met[48].

Even with better tools and a more powerful computer, the processing needs of the mission control software quickly exceeded the capacity of the 7094. IBM recognized that the usual 32K memory of the machine would be insufficient when the company prepared its proposal. Therefore, it suggested the use of look-ahead buffering, which meant the next set of programs needed during a mission would be loaded over the ones going out of use[49]. The commercial practice of using tape storage for waiting programs became impossible due to the size and speed demands of the Gemini software. Thus, IBM added large core storage (LCS) banks to the original machines. These banks, even though not directly addressable, provided a higher speed secondary memory. Tapes would be loaded to the large core and then transferred to primary storage as needed[50]. An IBM engineer credited work in the use of LCS and paging memory as being influential in the development of IBM's version of virtual memory, the main software technological advance of its fourth generation 370 series machines of the early 1970s[51]. As the Gemini program continued, NASA grew more concerned about the ability of the 7094s to adequately support Apollo, considering the expected greater complexity of the navigation and systems problems. Kraft expressed concern that the "real time" in the RTCC needed enhancement[52]. As the large core filled, loading from tape for certain programs became common practice. Once, when

President Lyndon B. Johnson was visiting the control center, the NASA official leading the tour wanted to show the president a fancy display. Not fully conversant with the software, he chose one that ran off tape, so the entire party stood uncomfortably, minutes seeming like hours, while the machine dutifully found the program and put up the display[53]. NASA wanted a change.

It was about this time that IBM announced its System 360 series, a compatible line of several computers of different sizes using a new multiprocessing operating system that owed some of its characteristics to the company's NASA experiences. NASA thought the upper level machines of the new product, specifically the 360/75, would have sufficient power to replace the 7094s for Apollo, although the LCS would have to be continued due to the sheer size of the software. IBM's announcement, as is usual with the company, preceded the shipping dates of the machines by some months. It did not take long for NASA to realize this and become impatient. Control Data Corporation (CDC) released its 6600 line of computers in 1965 and was actually shipping to customers as IBM failed to deliver. Robert Seamans of NASA Headquarters suggested that the Manned Spacecraft Center buy 6600s and let IBM retain the software contract[54]. CDC's machine was actually faster and more powerful than the 360. Later, CDC sued IBM, claiming its premature 360 announcement sought to hold the market and that claims made for the 360 were not realized when the product actually came out. IBM settled out of court with major concessions totaling nearly $100 million, rushing delivery of the first 360 to Houston in time to stave off the movement to other vendors. NASA announced the conversion to the 360 in a news release dated August 3, 1966.

Transition to Apollo

Although the four remaining 7094 computers continued to support flight operations through the first three Apollo (unmanned) missions, IBM used the first replacement 360 to begin software development for the Apollo lunar flights. As in Gemini, two spacecraft, the command module (CM) and the lunar excursion module (LEM), needed support, with five computers each contributing to the overall system. Again, LCS provided added memory. Unfortunately, all the software could not be moved directly from one machine to the other due to the change in operating systems. The new operating system for the series, OS/360, had the multitasking capability developed during Mercury days but operated primarily in batch mode. Many programs could be entered, either by cards or through remote entry from terminals, and run together, but not in real time. The priority-interrupt provisions on the standard operating system were not sophisticated enough to handle the sorts of processing Apollo needed. Beginning in 1965, IBM modified the operating system into RTOS/360, the real-time version[55]. Extensive use of modularization helped in the transition. Separately compiled subprograms in FORTRAN, moreover, could be moved to the 360 with relative ease, but the assembler-based code had to be modified. This work continued for nearly as long as it took to get the original system operating, even though the architecture remained essentially intact.

One problem would not go away: memory. Each 360 had 1 million bytes of main memory, about four times the size of 7094 main store. A further 4 million bytes of LCS was added to each machine[56]. Even with some of the NETCHECK functions transferred to the new twin 360s in the Goddard Real-Time System (GRTS) and with seldom-used programs such as the radiation dosage calculator and ground telescope pointing program permanently located off-line, memory use rose to match the additional space. Simply meeting the requirements for ascent filled the main store[57]. At this time, NASA's Lynwood Dunseith, who had worked on the ground software since Mercury, realized that the worry over memory was causing programmers to develop idiosyncratic, "tricky" code in an effort to save a few words[58]. Dunseith knew the danger of that attitude, since it made the programs even more complex than their absolute complexity warranted. During the period he managed the software development, he tried to reduce the dependence on such expedients. It helped him that the 360s made it possible to develop significant parts of the software in FORTRAN[59]. Although FORTRAN is not as easily readable as some other procedural languages, it far exceeds 360 assembler in understandability.

As the Apollo system moved into the operations phase, the use of the Dynamic Standby Computer waned. During the first manned flight, Apollo 7, the Mission Control Center used a single computer for just under 181 hours of a 284-hour support period, which included countdown and postflight operations[60]. During Apollo 10, a dual spacecraft flight with LEM operations near the moon, the plan was to use the standby for 5 hours before a maneuver. Therefore, on only six occasions in an 8-day flight would there be two-computer support. To assist an off-line standby in coming to the rescue of a failed primary, operators made checkpoint tapes of current data every 1.5 hours. A failure of the Mission Operations Computer occurred at 12:58 Zulu on May 20, 1969. By 13:01, the standby had been brought up, using a checkpoint tape made at 12:00[61]. No significant problems resulted, which is actually a good summary of mission control operations throughout the Apollo era, Skylab, and the Apollo–Soyuz Test Project.

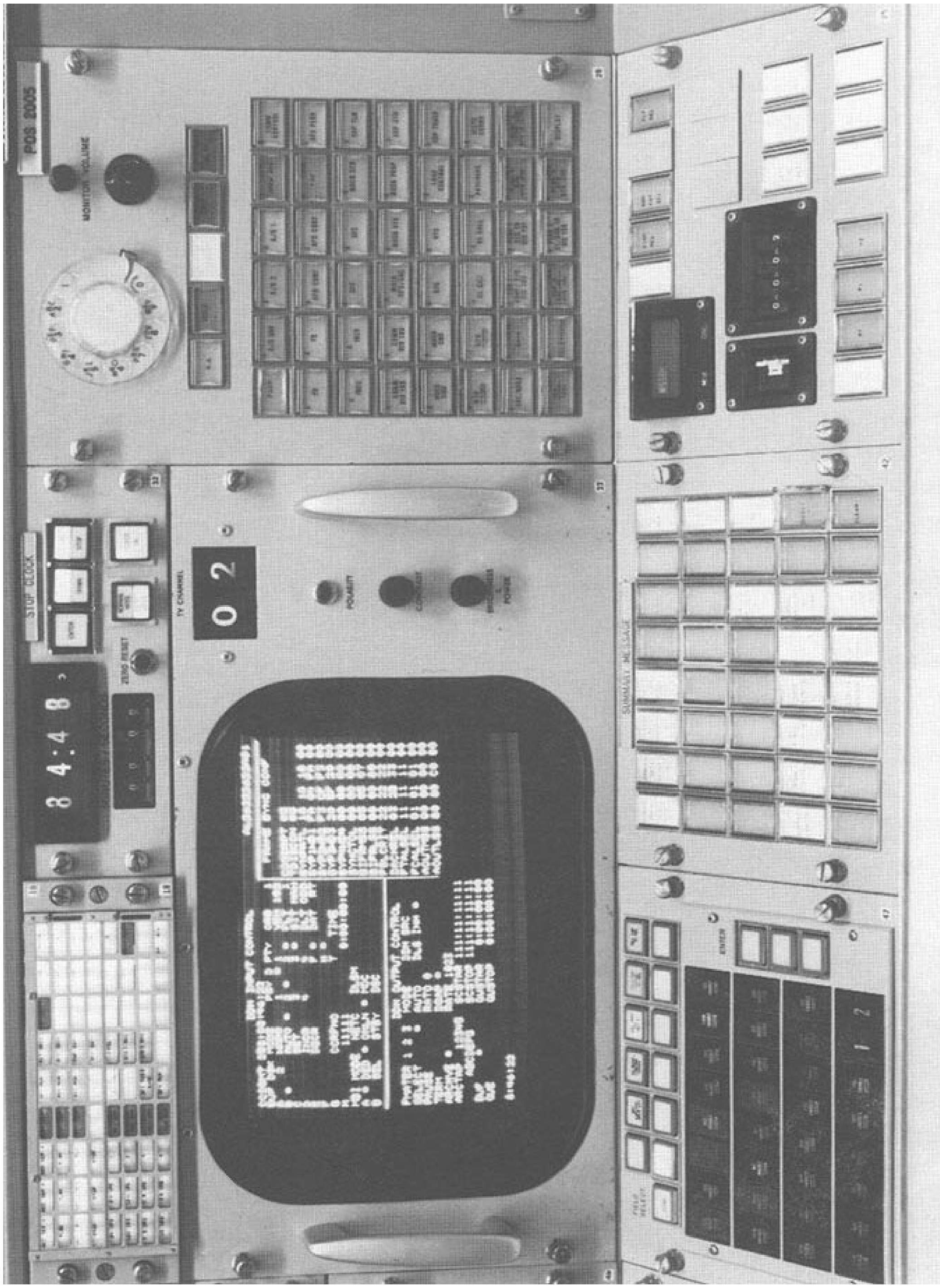

Figure 8–2. A display and control panel in Mission Control for the Shuttle program (NASA photos-80-26315)

Reducing Mission Control: Conversion to the Shuttle

During planning for the Space Transportation System, with frequent launches and multiple missions aloft expected, NASA studied ways to make the spacecraft more autonomous and thus reduce the functions of mission control. IBM again won the ground support contract, this time over primary competitor Computer Sciences Corporation[62]. Beginning in June 1974 and continuing into the 1980s, IBM worked on a new software system and mission-specific changes[63]. Five System 370/168 mainframe computers make up the Shuttle Data Processing Complex, the nominative successor to the RTCC. Each has 8 million bytes of primary storage, and, being virtual memory machines, do not need auxiliary storage of the LCS type. Disk is used instead. Three computers are involved during operations: One computer is the Mission machine, one, a Dynamic Standby Computer, and a third, the Payload Operations Control Computer. Now, in the late 1980s, these computers are being replaced by IBM 3083 series machines, marking Mission Control's fourth generation.

By this time, quite experienced and fairly knowledgeable about what would be needed, NASA and IBM approached the ideal of thorough design before coding began[64]. Reflecting the structure of the on-board software, the requirements documents proceeded through different levels of complexity. For the first time in ground software development, a quality assurance group from outside the development organization watched over software production[65].

The efficiency of the software developers increased with the conversion from batch processing to interactive processing. During Mercury, Gemini, and Apollo, programmers tested new software in batch. With the main IBM Federal Systems Division office nearly a mile from the actual computers housed in Building 30, it was necessary for a courier to pick up card decks, deliver them to the Computing Center, and later return the results. In this manner, an average of only 1.2 runs per programmer per working day was possible. During 1974–1976, NASA commissioned a study of batch versus interactive programming, in which programmers using terminals could prepare jobs and run them from the IBM building. Using IBM's Time-Sharing Option (TSO) system, interactive processing clearly won out over batch in terms of effectiveness. NASA accordingly ordered all Shuttle ground software to be done under the time-sharing system[66].

Regardless of the intentions of the Shuttle managers to shrink the ground operations software, the ground support functions provided by the Data Processing Complex have not been reduced. Some parts of the original tasks are handled more completely on-board, but the continued addition of new equipment and concepts increased the size of the software. It supports over 40 digital displays and 5,500 event lights. The total size of the system is 600,000 lines, roughly 26% larger than Gemini and rivaling Apollo[67]. Shuttle missions are approaching the complexity that a single computer can no longer support[68]. In addition, high between-flight change traffic delayed the transition to the operations era. As late as 1983, 8% of the total code changed each mission, keeping 185 programmers busy. New and more powerful computers can always be added, but the process of changing software must be automated or the expense of labor intensive maintenance will continue to the end of the Shuttle program.

UNMANNED MISSION CONTROL COMPUTERS

Mission control of unmanned spacecraft is significantly different from that of manned spacecraft. Most important of the differences is the long duration of many unmanned flights. Except for Skylab, no American manned flight has lasted more than 2 weeks. In contrast, when Voyager 2 encounters Neptune in 1989, it will have flown for 12 years. During that time, the Voyager Project staff must monitor the health of the spacecraft and gather and interpret the data it is collecting. Few of the original engineers will still be associated with the mission, so conceptually mission planning for a long-duration unmanned flight must concentrate on an extended view of operations and the development of detailed documentation[69]. Another difference is that the manned mission control centers are used for one project at a time, whereas the unmanned centers may be controlling a wide variety of missions. So far, there has been no overlap in the manned *programs* in the sense that no Mercury flights continued after Gemini flew, and so on. In contrast, the Jet Propulsion Laboratory (JPL) commanded Surveyors, Lunar Orbiters, Pioneers, and Mariners all at once in the mid-1960s, and has continuously been responsible for multiple missions.

Control of Near-Earth Missions at the Goddard Space Flight Center

NASA formed Goddard Space Flight Center with the Naval Research Laboratory's Vanguard Project team as a nucleus. After Vanguard ended, use of the IBM 704 in downtown Washington ceased, and a model 709 was installed at Goddard on May 23, 1960, as a replacement machine for use in working with earth-orbiting satellites. Within 2 months, the first of six 7090 computers also arrived. Folklore has it that Goddard soon housed 1% of the total computing power in the entire United States. Although two of the 7090s and later other computers supported Mercury flights, Goddard's most substantial customer base has been the plethora of scientific, navigational, communications, mapping, and weather satellites launched in the last quarter of a century.

Goddard pioneered the use of dedicated small computers for specific missions, thus eliminating the complexity of handling multiple missions on a single mainframe. This occurred in spite of the presence of large numbers of big computers. Some command and control and definitely navigation calculations are carried out on large machines, but each project has a small computer to handle data reduction and the day-to-day operation of the spacecraft. As examples, the Nimbus weather satellite program used Control Data 160A computers, the Orbiting Geophysical Observatory had Scientific Data Systems SDS 910 and 920 computers, and so did the Orbiting Solar Observatory[70]. These machines could be sent on to another project when their current job ended, and in fact some of the SDS machines had rather long lifetimes of nearly two decades. In addition to using small computers at the control center, Goddard installed UNIVAC 1218 computers in the Manned Spaceflight Network ground stations, originally for control of Gemini and Agena and later for Apollo. Both the 160As and 9105 were among the first products of their respective fledgling companies, and, with the 1218 and Digital Equipment Corporation's PDP series, the forerunners of the minicomputer boom of the 1970s.

Relatively little changed in the general techniques of mission control at Goddard for about two decades. As the 1980s continue, the trend is for the majority of unmanned satellites to be commercial rather than scientific in nature. Commercial satellites are controlled by their owners, although NASA provides orbit determination and some command services on a reimbursible basis. However, sufficient missions exist, such as the expected 17-year duration Hubble Space Telescope, to keep Goddard involved in ground control activities for some time, along with its continued commitment to NASCOM and STADAN.

To the Sky's Limit: Mission Control at JPL

As Goddard strove to standardize earth orbital operations and distribute its functions, JPL approached the similar problems in a different way, centralizing operations as much as possible. In many respects, Goddard and JPL are fraternal twins. Each has a set of ground-tracking stations, plus on-site control centers for a variety of missions. The difference is that JPL is responsible for deep space exploration. In fact, the lower limit of its responsibilities is set at 10,000 miles. For a short period, it did satellite work. JPL developed the guidance system and propulsion for the Sergeant battlefield missile and studied adapting clusters of the motors as upper stages to the Redstone missile. The resulting Jupiter-C launch vehicle put America's first satellite into orbit on the night of January 31, 1958. Called Explorer I, the satellite carried JPL-developed instrumentation. A room near the office of laboratory director Dr. William Pickering became an active unmanned mission control center since it contained communications equipment connected to the tracking network that confirmed Explorer reached orbit. That same year NASA was formed and JPL became closely affiliated, changing its mission to deep space work.

In 1959, the early Pioneer flights aimed at the moon. JPL built a series of tracking stations, beginning at Goldstone in the high desert of California, to track the missions[71]. Unlike earth orbiters, whose closeness to the planet make it necessary to have a large number of stations to stay in contact, deep space probes needed only three stations spaced so that one would always face the spacecraft. Initially, the stations were located in Australia and South Africa as well as at Goldstone, but later one in Madrid replaced the African station and the Australian one moved from Woomera to Canberra[72]. The stations were collectively named the Deep Space Instrumentation Facility.

From the beginning, JPL considered using computers in the stations as data-gathering devices. One 1959 report suggested using IBM 650 machines, which were small computers[73]. In 1962, Dr. Eberhardt Rechtin, head of the Instrumentation Facility, sent Paul Westmoreland and Carl Johnson to evaluate the computers of Scientific Data Systems, a new company[74]. Westmoreland and Johnson thought that the SDS 910 could be used as the data gatherer, with the slightly more powerful 920 as a data processor. Accordingly, Rechtin directed that the machines be ordered and got the first 920 built and the second 910. The 910s and 920s still functioned in similar tasks as late as 1985![**]

Functionally, the SDS computers took data received from the spacecraft and formatted and recorded it on magnetic tape. A computer at JPL processed the data more completely. Initially, an IBM 704 similar to the one used for Vanguard did the work. JPL installed the computer in late 1958 to use with Pioneer 3 and 4[75]. Early Ranger lunar impact flights later had all data reduction done off tape on that machine. Data in analog form on the tapes would be translated into numbers that spewed out on teletypes and punched paper tape. Aerojet-General Corporation also owned a 704 that JPL used as a backup[76].

[**] After 1968, the SDS machines were known as XDS 910 and 920. Xerox bought out SDS and renamed the products "Xerox Data Systems."

Planning for the first Mariner missions revealed that more computing power would be needed at JPL to handle the increased data generated both by more instrumentation and longer mission lifetimes. Dual 7090 computers similar to those installed at Goddard were bought for data reduction. To provide flight controllers with more up-to-date information about spacecraft telemetry, a Digital Equipment Corporation PDP-1 computer served as a near-real-time data processor. Data could be displayed on teletypes from 4.5 to 7 minutes after it was received[77]. By this time the Deep Space Instrumentation Facility could transmit data via NASCOM instead of having to wait for airmail to deliver the tapes. Operations with this equipment taught JPL at least one useful lesson: Power fluctuations in September and December 1962 caused both 7090s to go down at once, eliminating the redundant capability[78]. As a result, JPL built an auxiliary power generation facility, perhaps leading the manned Mission Control Center, under construction at this time, to do the same.

Centralizing the Effort

During the 1960s, NASA found itself about to be involved in a large number of critical deep space projects. Ranger would be followed by the Surveyor series of lunar landing missions. Mariners would continue to fly to Venus and Mars, with several targeted for Martian orbit and imaging duty. Lunar Orbiters would look for Apollo landing sites and Pioneers were aimed at deeper space. JPL did not have primary responsibility for all of these programs. Lunar Orbiter came from Langley Space Flight Center, and Pioneer from the Ames Research Center. If each responsible organization had to set up a control center for its spacecraft, considerable overlap and duplication would occur. Accordingly, in 1963, NASA decided to have JPL track and command all deep space missions, with the help of project personnel from home centers stationed at JPL[79]. On December 24, 1963, JPL's director William Pickering formally established the Deep Space Network[80]. Managed by William H. Bayley, with Eberhardt Rechtin as technical head, it would serve all of NASA, just like NASCOM did from Goddard[81].

JPL was already building a Space Flight Operations Facility to house new, more powerful computers and the various teams from its own projects. Anticipating NASA's decision, Eugene Giberson, then of the Surveyor project, directed some of his money to help develop the centralized computer center[82]. The combination of the Operations Facility and the Deep Space Instrumentation Facility was the Deep Space Network. After opening on May 15, 1964, the Operations Facility supported Mariner Mars 1964 as its first flight[83].

Even though expected to handle all deep space missions, some organizations fought to retain mission functions. Ames set up the Pioneer Off-Line Data Processing System (POLDPS) in 1965 to handle non-real-time data recorded by the SDS 910s at the stations[84]. Both the Lunar Orbiter and Surveyor projects also wanted to record their telemetry data at the stations, so the Network bought dual SDS 920s for each site. Later, Pioneer 10 and 11 data were processed with these systems[85]. Langley originally wanted to control the Viking Lander, but costs and common sense forced that job back to JPL.

Figure 8–3. The Space Flight Operations Facility central control room at the Jet Propulsion Laboratory. (JPL photo P23358BC)

Evolution of the Space Flight Operations Facility

JPL's Space Flight Operations Facility has had three generations of equipment. Beginning in 1964, two strings of solid-state computers formed the basis of the system. Each consisted of an IBM 7094 mainframe, an IBM 7040 medium-sized computer, and an IBM 1301 disk storage system placed between them. Later, a trio of System 360/75 computers replaced this configuration. More recently, the control center adopted a distributed computing strategy similar to Goddard's.

As in the manned programs, during critical mission phases both strings of the original generation of equipment would be running at the same time but with the data from the stations only routed to one of them. If a 7094 failed, its associated 7040 could be connected to the other 1301 (and, thus, the second 7094), leaving the second 7040 as another layer of backup[86]. Later upgraded to a 7044, the smaller computer acted as a traffic cop on the incoming data. All inputs (teletype, telephone, microwave) went to the machine before they went anywhere else, and the software in the 7040 routed the data to active programs, inactive programs, or administration stations[87]. George Gianopolis of JPL, one of those charged with the responsibility of getting the system to work, remembers that the 7040s were especially difficult to install[88]. The 7040s deposited data on the 1301 disk storage system. A 54-megabyte hard disk, the 1301 served both the 7040 and the 7094 from the middle, so both could access data at identical addresses. This concept presages the network file servers in the modern office and the Common Data Buffer in the Launch Processing System. Airline reservation systems and other large data base operations utilized the same configuration beginning at about the same time. Using a smaller computer to handle resource-hungry input and output tasks and a common storage area is a standard network concept today. As for the 7094, the flight operations director could control its use by "percentage time sharing" in which higher priority jobs simply got more machine time[89]. The primary functions of the 7094 were telemetry analysis, tracking, predictive work for the stations, and maneuver commanding. UNIVAC computers in the JPL institutional computer center did the navigation calculations as batch jobs, separate from the Operations Facility computers[90].

Although a powerful system, the 7040/7094 combination had to stretch to meet mission requirements. Upgrading it to 7044/7094 Model II status helped some, but the system could handle only a Mariner mission (two spacecraft) or a Surveyor but not both[91]. Surveyor project officials even had to add a PDP-7 as a front end computer to the front end computer, putting it between the stations and the 7044 and driving strip chart recorders[92]. More assistance came during the Mariner Project when engineers realized the UNIVAC 1218 computers used in preflight testing of the spacecraft could also do engineering telemetry analysis[93]. This was not done until Mariner Mars 1971. Soon, though, the acquisition of more powerful 360 series machines ended the reign of the 7094s.

Monolithic Computer Systems

In October of 1969, JPL installed its first System 360/75, a gift from the Manned Spacecraft Center, where it was considered surplus. A second machine arrived in April 1970, this one left over from the demise of NASA's Electronics Research Center in Cambridge, Massachusetts. JPL bought a third machine, which survived until August 1983[94]. Each 360 had 1 megabyte of primary core storage and 2 megabytes of LCS, half that of an Apollo-configured machine[95]. Two of the 360s controlled missions as a redundant set, with the third used for development work. A special switch connected the 360s to the institutional UNIVAC 1108 mainframe computers so that tracking data could be directly transferred for use in navigational computations[96]. But the gift from Houston was not entirely welcome at JPL, for along with it came the Real-Time Operating System (RTOS) developed by IBM for the Apollo program. As Gianopolis saw it, "what we picked up from Houston was good for Houston, but not necessarily for us"[97].

Unmanned spacecraft missions needed to create large data bases capable of handling the long series of telemetry signals that might go on for months or years. IBM's RTOS tried to keep all data in core memory, using disk storage as read-only devices. JPL needed to be able to write to the disks. Also, each Apollo computer concentrated on real-time functions and did not do development work. JPL wanted to run FORTRAN jobs on the machine, but RTOS could not handle it[98]. A crisis of sorts arose with the Mariner Mars 1971 orbital mission. During the cruise period to the planet the ground software failed every 5 hours. By the time Mariner reached orbit around Mars, the failure rate fell to once every 20 hours[99]. Still, something had to be done, so JPL contracted for an overhaul of the operating system, culminating in 1972 with the JPL/OS, which incorporated the needed changes.

Since the 360s lacked a small computer for a front end (original thinking being that the machine could handle the load by itself), JPL implemented the idea of using the preflight testing computers in mission support for Mariner Mars 1971[100]. Incoming telemetry went to the UNIVAC 1230/1219 set first. Then the 360s did commanding, tracking data evaluation, predictions for the stations, and engineering work. Besides the UNIVAC test set, the UNIVAC 1108s provided navigational data and, by then, the Image Processing Laboratory at JPL had its own 360/44 for processing planetary imaging[101].

Viking, a much more complex project than Mariner, and with essentially four spacecraft (two orbiters, two landers) to control, stretched the 360s and their helpmates to the limit. JPL assigned the small UNIVACs to handle the Viking Orbiter data, since the spacecraft were built and tested at JPL and the software was in place. System 360s controlled the landers[102]. At peak, 700 controllers worked the Viking mission, more than on any other space program to date (The count was double that of Skylab, the largest manned support group)[103]. Facing dual Voyager missions and Galileo, with the prospect of continuing Viking far past the original mission life estimates, JPL was again looking for a way to upgrade mission control.

Distributed Computing Becomes the Answer

As JPL discovered planning the computers for the Voyager onboard systems, functionally distributed small computers offered more reliability and cost savings than large single processor systems. The Laboratory implemented a distributed system to fill its Voyager ground control needs as well. Viking was the last mission to be supported from a large mainframe computer. By the time Voyager neared Jupiter, two strings of dedicated minicomputers performed the telemetry, tracking, and engineering monitoring functions. A single minicomputer shared by several projects did the commanding. Why did the change occur? First, the Deep Space Network was unhappy with the level of support it derived from a centralized system. Second, even though centralizing deep space mission control at JPL was a sound idea, putting too many missions on a single computer system was less so. No matter how much JPL tried to standardize things, each mission had its unique characteristics, calling for changes in the support software. With a distributed system, changes could be made without affecting other software. When missions neared critical phases, such as launch or encounter, software had to be frozen until the phase passed. With enough spacecraft aloft, the amount of time available to change software became quite short[104].

NASA provided an additional impetus to switch to a distributed system. Acknowledging the Deep Space Network's concern over using the 360s in the JPL control center and worried that the Network could not monitor its performance when supporting projects originating at other centers (such as Pioneer), the Agency directed the Network to develop monitoring capability in separate computers. Between 1972 and 1974, a set of ModComp 2 minicomputers was connected in a local area network at JPL to implement this directive[105].

In 1976, the control center itself converted from 360s to ModComp 2s and 4s in preparation for Voyager. Later the Laboratory added ModComp Classics and retained some of the UNIVAC 1218s and 1230s (renamed 1530s after upgrades)[106]. These computers are arranged in redundant sets. Each project (Voyager, Galileo, etc.) has its own telemetry machine and shares a command machine. A routing computer in the basement of the Space Flight Operations Facility building is the entry point of all data from NASCOM, sending the data to the appropriate computer. The command computer reverses the process for outgoing data.

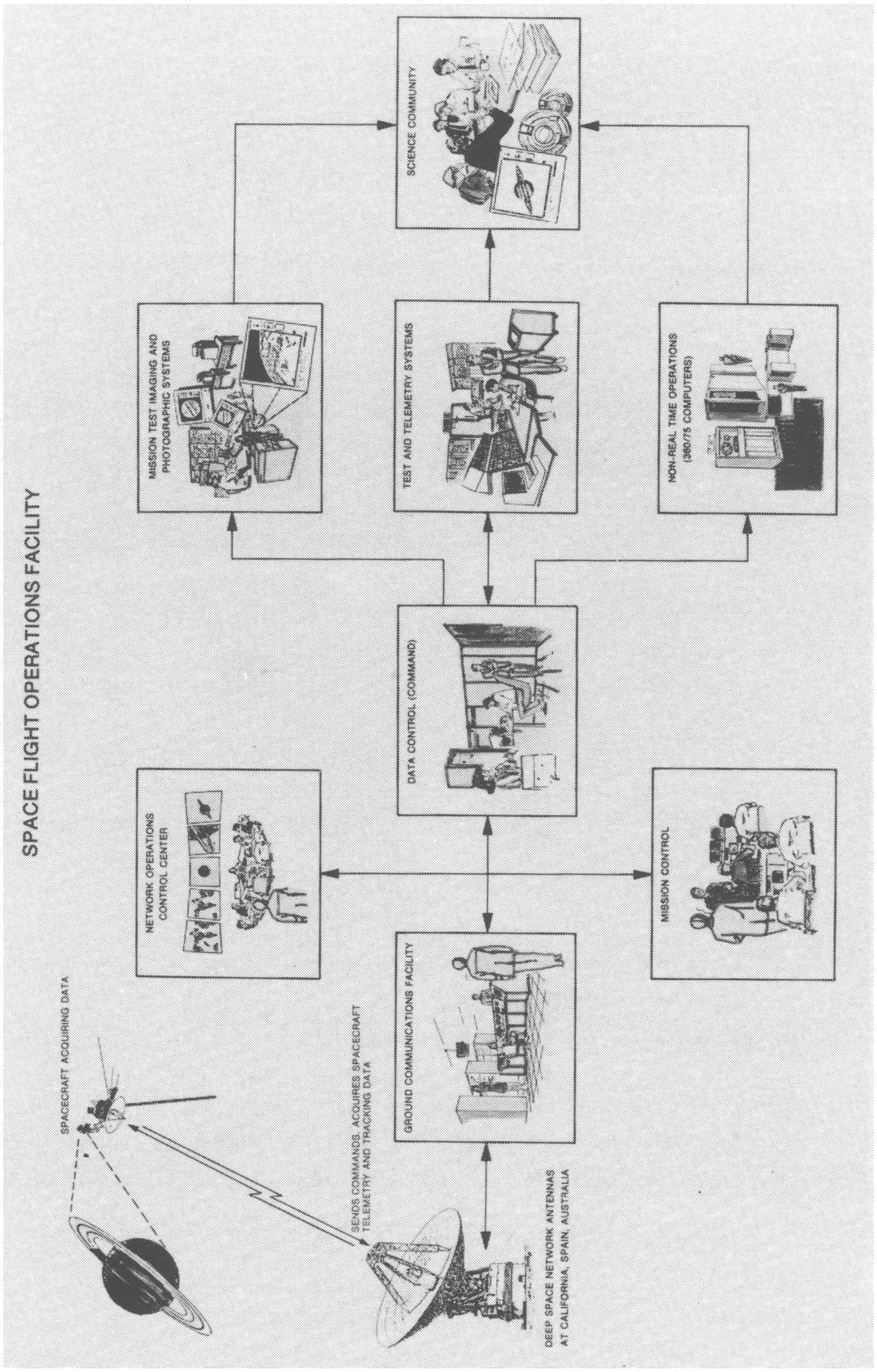

Figure 8–4. A schematic of the components of the Space Flight Operations Facility. (JPL 333-660)

By the early 1980s, the Deep Space Network was heavily into distributed computing. It converted from the 920s to ModComp 2s at the stations and ordered three Digital Equipment Corporation 11/780 VAK superminicomputers for use at JPL. Nearly 100 minicomputers were connected on an Ethernet. The use of high-level languages became the rule rather than the exception[107]. Key to the future success of the Deep Space Network is the inherent flexibility of distributed computing centers. They mirror the use of modules in software: interchangeable parts in a changing field.

Software Development in the Deep Space Network

Software development for the control center and the stations has always been a challenge, as programmers have struggled to use machines built primarily for commercial use in the arena of real-time control. In keeping with the centralization of the computers in 1963, the original software developers worked under Frederick Lesh in a "program analysis group"[108]. JPL separated software development for mission operations from that of the network stations just before Mariner Mars 1969[109]. Also, at that time emphasis began to be placed on making the software more parameter-based and, thus, more flexible and capable of use on multiple missions[110]. A new management concept led to the assignment of a program cognizant engineer to each software system engineer. The software engineer would define requirements, prepare test cases, and oversee the program engineer, who would produce the code. This turned out to be quite successful and avoided the difficulties encountered when an engineer thought (wrongly) that he could do both jobs alone[111]. In microcosm, this is the "outside verification" concept used extensively in programming now.

Martin-Marietta Corporation, the Viking Lander contractors, had to do some dangerously unique software development when NASA decided to move control of the Lander from Langley to JPL. Since Orbiter software development and giving support to other missions tied up JPL's computers, Martin took the chance of developing the Lander software in a "minimal higher order language," specifically a hopefully transportable subset of FORTRAN. Martin's solution reflected its recent migration to IBM 370 series and Control Data 6500 series computers at its Denver plant. These were technologically more advanced than the JPL computers and could not be trusted to produce directly transportable software[112]. The idea worked, but Martin admitted that the requirement for delivering mission support software 10 months before the flight provided strong motivation[113].

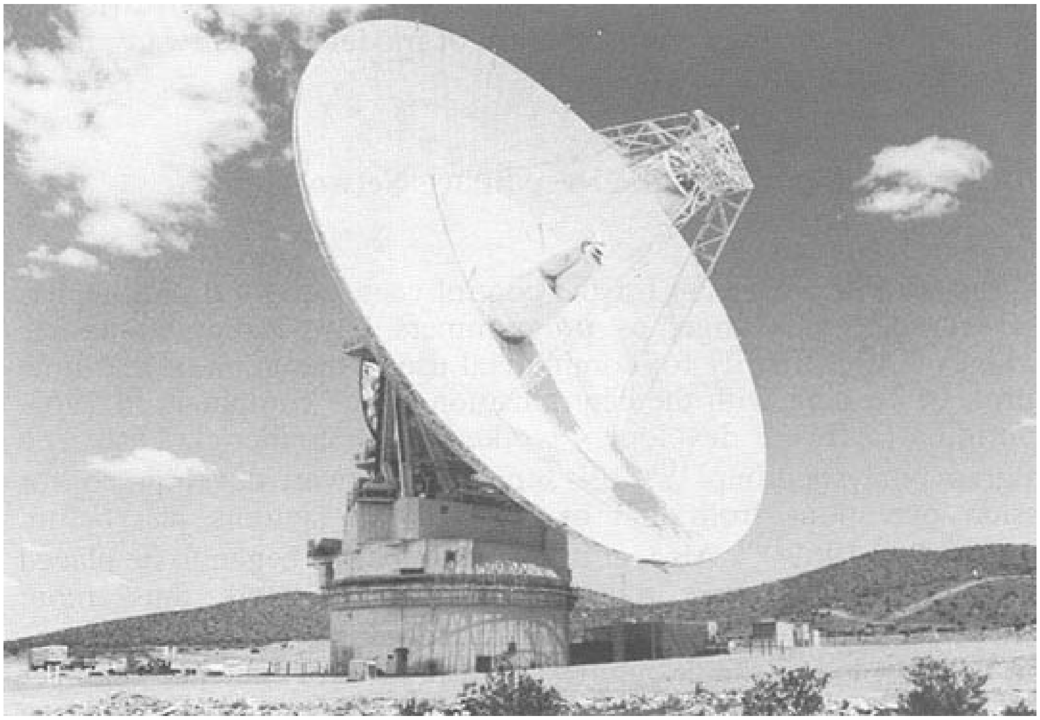

Figure 8–5. The 64-meter antenna of the Deep Space Network at Goldstone, California. (JPL photo 333-5967BC)

As JPL moved to a distributed system, a concerted attempt at establishing software standards has resulted in a state-of-the-art set of documents[114]. Based on structured programming and software engineering principles, these documents and the decision to use more high-level languages such as HAL, C, and Pascal make the Deep Space Network one of the most sophisticated software organizations within NASA. A further decision to no longer change commercial operating systems (possible now that computers are more general purpose), will help ensure continued cost reduction and consistency[115].

Mission control is the most computer-intensive part of spaceflight operations. From the beginning of both the unmanned and manned programs, the computer industry has been constantly forced to stretch the capabilities of both hardware and software in order to meet NASA's needs. In this way, NASA was a driving force in the development of multiprocessing operating systems and large computer complexes.