R-918

December 1971

The JOSS Years: Reflections on an Experiment

Shirley L. Marks

Rand
SANTA MONICA, CA. 90406

R-918

December 1971

The JOSS Years: Reflections on an Experiment

Shirley L. Marks

Rand
SANTA MONICA, CA. 90406

Bibliographies of Selected Rand Publications

Rand maintains a number of special subject bibliographies containing abstracts of Rand publications in fields of wide current interest. The following bibliographies are available upon request:

Aerodynamics • Africa • Arms Control • Civil Defense
Combinatorics • Communication Satellites • Communication Systems
Communist China • Computer Simulation • Computing Technology
Decisionmaking • Delphi • East-West Trade • Education
Game Theory • Health-related Research • Human Resources
Latin America • Linguistics • Maintenance • Middle East
Policy Sciences • Pollution • Population
Privacy in the Computer Age • Program Budgeting • Public Safety
SIMSCRIPT and Its Applications • Southeast Asia
Space Technology and Planning • Statistics • Systems Analysis
Television • Transportation • Urban Problems • USSR/East Europe
Water Resources • Weapon Systems Acquisition
Weather Forecasting and Control

To obtain copies of these bibliographies, and to receive information on how to obtain copies of individual publications, write to: Communications Department, Rand, 1700 Main Street, Santa Monica, California 90406.

Published by The Rand Corporation

PREFACE

JOSS,* Rand's conversational computing system for the computer novice, was initiated in May 1963 as part of the Air Force-sponsored Information Processor Project. When formally proposed in 1961 for implementation on the JOHNNIAC computer, JOSS was described as an experiment in improving communication between man and machine. Designers felt that a simple language and a system applicable only to small numerical problems would enable them to learn in a reasonably short time how intimate interaction could ease communication.

The experiment was expanded in a second version of the system implemented on a commercial computer in 1965, and has continued to indicate the value of a personalized approach to computing power. Evidence of the influence of JOSS on modern conversational time-sharing systems has appeared in the wide use of the term "JOSS-like" to refer to such systems created for the nonprogrammer.

This Report for the general reader is the final publication in a series of documentation covering the design and use of JOSS in its two versions. The experience of eight experimental years is presented with respect to hardware, software, and users. Emphasis is on the personal nature of the service, and the role of the user both as an individual and as one of a group competing for the system's resources.

*JOSS is the Trademark and Service Mark of The Rand Corporation.

SUMMARY

JOSS, the JOHNNIAC Open-Shop System, was proposed in March 1961 as "an exploration into continuous and intimate contact between a human user and a computer." This personal time-shared computing service initiated the first phase of the Air Force-sponsored Information Processor Project, whose goal was to improve communication between man and machine. The simple language of JOSS, its familiar typewriter console, and its conversational environment brought to the computer novice for the first time both the elementary features of a desk calculator and the sophisticated skills of an electronic computer.

Prior to JOHNNIAC's retirement to the Los Angeles County Museum in February 1966, a second version of JOSS was implemented on a Digital Equipment Corporation's PDP-6 computer. JOSS II provided thirty times the speed of JOSS I, five times the storage, three times the number of consoles, and powerful new language features. The modern capabilities of the PDP-6 hardware not only added to the power of JOSS as a problem solver, but also enabled the system to monitor its own performance.

The JOSS experiment began in May 1963 with one console at the JOHNNIAC and four installed in offices of Rand staff selected to evaluate the system. By the end of 1970, there were 500 to 600 users at Rand and at Air Force sites across the country, many of them the occasional user for whom the system was intended. Mobile JOSS consoles permit easy access to the system from Rand offices equipped with special plugs; teletype terminals provide most of the remote service via standard data communications devices.

JOSS was designed for solution of the small numerical problem. Typical JOSS interaction is displayed by the physical indicators on the console and exemplified by the process of creating and modifying programs. The readability of the English-like commands and messages contributes to easy creation and modification. Such console and language characteristics have enhanced the personalized, open-shop approach to computing power for the nonprogrammer. However, the experimental years have illustrated the need to consider service not only to the individual user but also to the many users competing for the limited system resources. JOSS resources can probably be managed more formally than at present with little complaint from users as long as there is never interference with the essential friendliness of the conversational environment.

CONTENTS

FROM JOSS I TO JOSS II

Shortly after The Rand Corporation settled into its present location in Santa Monica in early 1953, the JOHNNIAC was completed and installed as Rand's first stored-program computer. Eight years later, it became the basis for JOSS,* the JOHNNIAC Open-Shop System, Rand's experimental time-shared service designed for the computer novice.

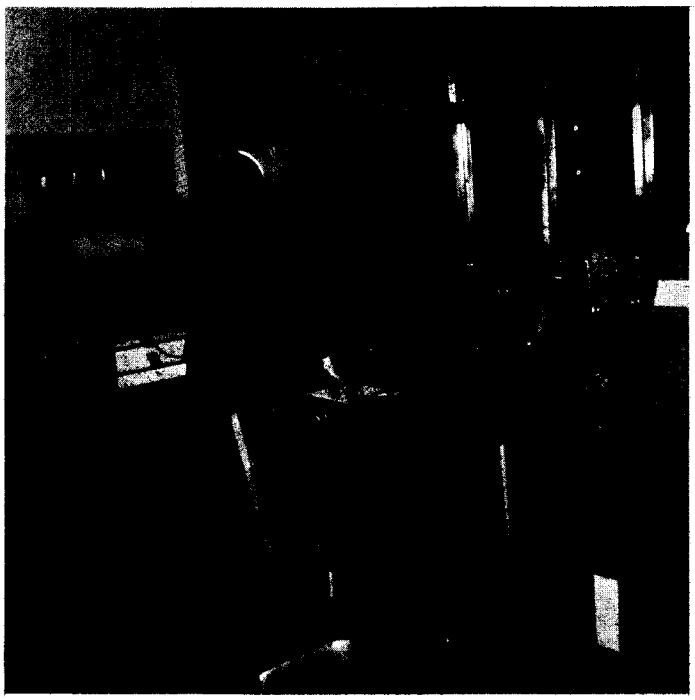
JOSS in its JOHNNIAC implementation began the first phase of the Air Force-sponsored Information Processor Project, whose goal was to improve communication between man and machine. In 1955, some years before the initiation of the project, J. C. Shaw expressed concern for the Rand open shopper, whose small numerical problems sometimes required the elementary features of a desk calculator, and other times the sophisticated skills of an electronic computer. He also noted some demand from the Rand staff for the ability to intervene in the calculation process.

A 1959 memo from W. H. Ware suggested that future information processors would require "a multiplicity of personal input-output stations, so that many people can interact with the machine at the same time." Shaw recommended in 1960 that the JOHNNIAC be used full time to service the open shop via hard-copy stations. He looked forward to the challenge of resolving communication and monitoring problems on a small scale.

In March 1961, JOSS was formally proposed as "an exploration into continuous and intimate contact between a human user and a computer." This personal computing service would offer a simple language compatible with the limitations of the JOHNNIAC hardware. By restricting the scope of problems to those that were numerical and relatively small, the designers hoped to learn in a reasonably short time how an informal exchange could ease the relationship between man and computer.

In addition to the JOHNNIAC, JOSS hardware would include ten remote typewriter consoles in fixed locations, and a communication system to mediate between the computer and the consoles. Shaw would design the language, console, and conversational environment. T. O. Ellis and M. R. Davis would direct the construction of the multitypewriter communication equipment.

*JOSS is the Trademark and Service Mark of The Rand Corporation.

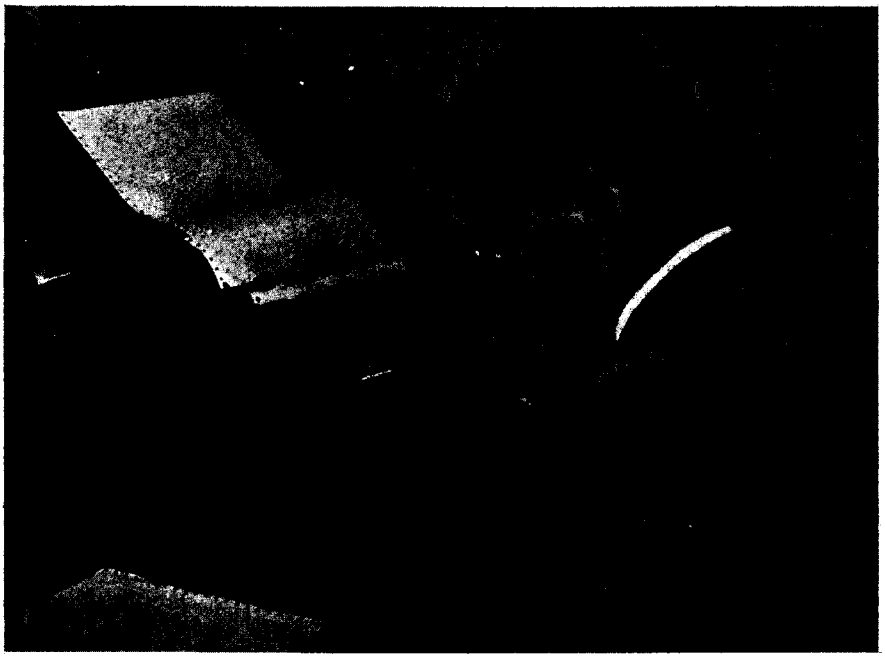


J. C. Shaw at JOHNNIAC

In May 1963, the JOSS experiment began with an initial five consoles, each consisting of an IBM model 868 typewriter and a small box to indicate the status of and control the functions of the console's communication electronics. One console was installed at the JOHNNIAC and four in the offices of Rand staff selected to evaluate JOSS. The first schedule of operation was announced on 17 June 1963: 9 to 12 each weekday morning.

In the years before JOHNNIAC's retirement, the lengthening JOSS schedule and the increasing number of consoles demonstrated users' growing addiction to this personal computing aide. However, the designers were anxious that JOHNNIAC's faithful but dated hardware should not prejudice the evaluation of JOSS. They emphasized that a modern computer, teamed with numerous user-oriented consoles, would considerably advance the system's interactive character. The additional speed and storage would not only add to its power as a problem solver, but would also enable the system to monitor its own performance. In July 1964, anticipating JOHNNIAC's enshrinement in the County Museum, the designers proposed a new JOSS.

Rand invited several computer manufacturers to replace JOHNNIAC. Each of the nine bids received was rated on such factors as component response time, arithmetic computation speed, ability to store internally the exact representation of



Mathematician John Williams at JOSS I Console

user-typed input, inclusion of consoles built to Rand specifications, and date of delivery.

Vendors for all elements of the system were selected, and Air Force funds were approved. The contract signed with Digital Equipment Corporation promised installation of a PDP-6 computer by 31 October 1965. In addition, DEC would build 30 consoles according to Rand plans. Each would include a modified IBM Selectric typewriter and a Rand-designed paging mechanism—a mechanical device associated with the typewriter that permits spacing in a single motion to the top of the next page.

Data Products would supply a disk file for the important between-session storage of user programs. Vermont Research would furnish a magnetic drum for temporary storage of user programs during sessions. James G. Robins Electronics agreed to build a line-finder, a device to locate an available channel to the computer for each console connected to a special office plug. Standard data communications equipment would couple JOSS to terminals outside the Rand buildings.

Following installation of a minimum configuration of the PDP-6 computer in late July 1965, JOSS II designers began to check out the software via teletype terminal. By October, they had completed checkout of a minimum system. In November, a prototype JOSS console arrived, and in December typed "JOSS at your service. Initials please:" for wives attending the Fall Joint Computer Conference in Las Vegas. This demonstration was the first remote use of JOSS.

C. L. Baker wrote to JOSS users on 9 February 1966:

> As all of you are by now aware, Friday, February 11 will be the last day on which JOSS service from JOHNNIAC will be available—"The end of an era."

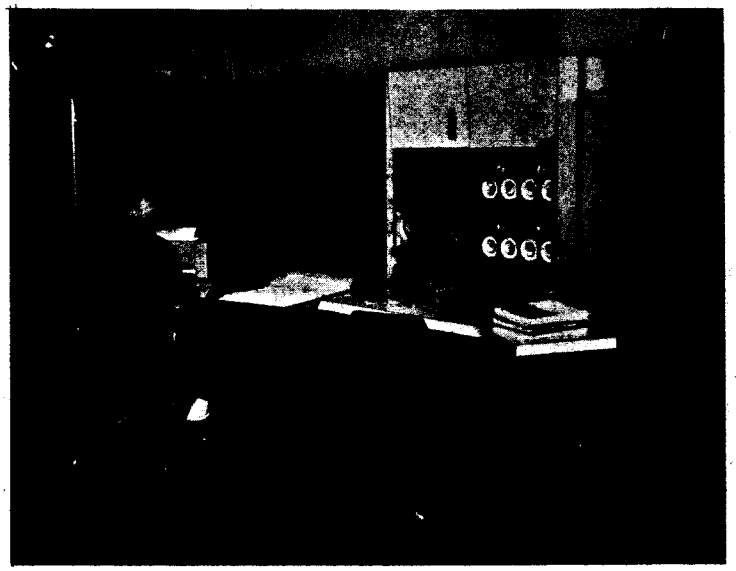
On February 18, W. H. Ware delivered a eulogy to JOHNNIAC before a gathering of friends. J.C. Shaw loaded a final JOSS I program, which counted down the seconds to JOHNNIAC's retirement.

JOHNNIAC Retirement, 18 February 1966

During the following months, the PDP-6 was moved to its permanent home in the Rand basement, production consoles were delivered, and the drum and disk and line-finder took their places in the new system. JOSS II operational hours were gradually lengthened as the system became more reliable. By the end of 1966, service was available to the Rand staff on a 24-hour-per-day, 7-day-per-week basis, minus hours for maintenance.

PDP-6/JOSS, with thirty times the speed of JOHNNIAC/JOSS, five times the storage, three times the number of consoles, and powerful new language features, broadened Shaw's original design. The enlarged programming group was headed by C. L. Baker, who contributed many human-engineering notions to the JOSS II console. J. W. Smith created the central processing routines, those machine-language programs that interpret and respond to requests in the JOSS language. I. D. Greenwald programmed the arithmetic and function evaluation routines, as well as the software that handles console-computer communications and enables the user to access files on disk. G. E. Bryan designed the JOSS monitor, the supervisory unit of the software, which assigns priorities to user requests and states, allocates resources, and schedules tasks. Within the monitor are routines to accumulate accounting information and to record performance statistics.

In April 1967, management of JOSS was transferred from the development group to a small staff under G. W. Armerding. R. L. Clark took charge of software upkeep and user satisfaction. A. C. Lucero attends to console and communications needs and orchestrates vendor activity. Shirley Marks assumed administrative du-



G. E. Bryan at PDP-6 Computer

ties from August 1967 until June 1971, when personal services were diminished and redistributed.

Greenwald had advised early JOSS users in "An Informal JOSS-II User's Manual":

> In the event of trouble, if you should "hang" in some inexplicable fashion, try the following (in order)—1. Rock the typewriter on-off switch. 2. Depress the "Interrupt" button-lite. 3. Call for help.

So began the JOSS II operational experience.

HARDWARE

Figure 1 shows the physical arrangement of the elements of the JOSS computer. Visitors to the PDP-6 room in Rand's basement inquire most often about these components—core memory, magnetic drum, magnetic disk file, communications equipment, and consoles.

The entire system software occupies just over half the 32,768 words of PDP-6 core memory. The remaining words of high-speed core are devoted to work space shared among users currently on line. The user programs not immediately requiring the system's attention are temporarily switched to the magnetic drum, which also contains a copy of the system for self-restoring under certain error conditions. The magnetic disk file, accessible to users on command, provides long-term storage for programs and data. Other peripheral devices, available only to the system or to maintenance personnel, are two magnetic tape units (which record accounting information) and an on-line teletype (which monitors the system's operation and permits its manual management).

Users sit at 31 JOSS consoles and numerous model 33 and 35 teletype terminals. Twenty-six of the 31 consoles are in the Rand Santa Monica buildings, where they can be connected to any of 200 JOSS plugs, each linked to the internal telephone network. In addition, a test panel in the PDP-6 room simulates a console for maintenance of the Selectric typewriters. The line-finder associates a connected plug with one of the 27 in-Rand console channels in the computer's multiplexer. The five remote consoles and the remote teletypes are connected to the multiplexer over both private and shared data lines, with and without voice capability.

The console is an integral part of the user-oriented system. The familiar typewriter keyboard, the small control box, the two-color "conversation" recorded amid lights and sound and touch indicating "speaker" and "listener"—all were meant to attract the researcher who is repelled by computer gadgetry. It was in introducing JOSS to the Air Force that the teletype became an alternate for the console in use outside the Rand buildings.

Three teletypes were used to check out the new system in the summer of 1965, prior to delivery of the first production consoles. After all the consoles had been accepted from DEC, the facility to accommodate teletype terminals was retained in

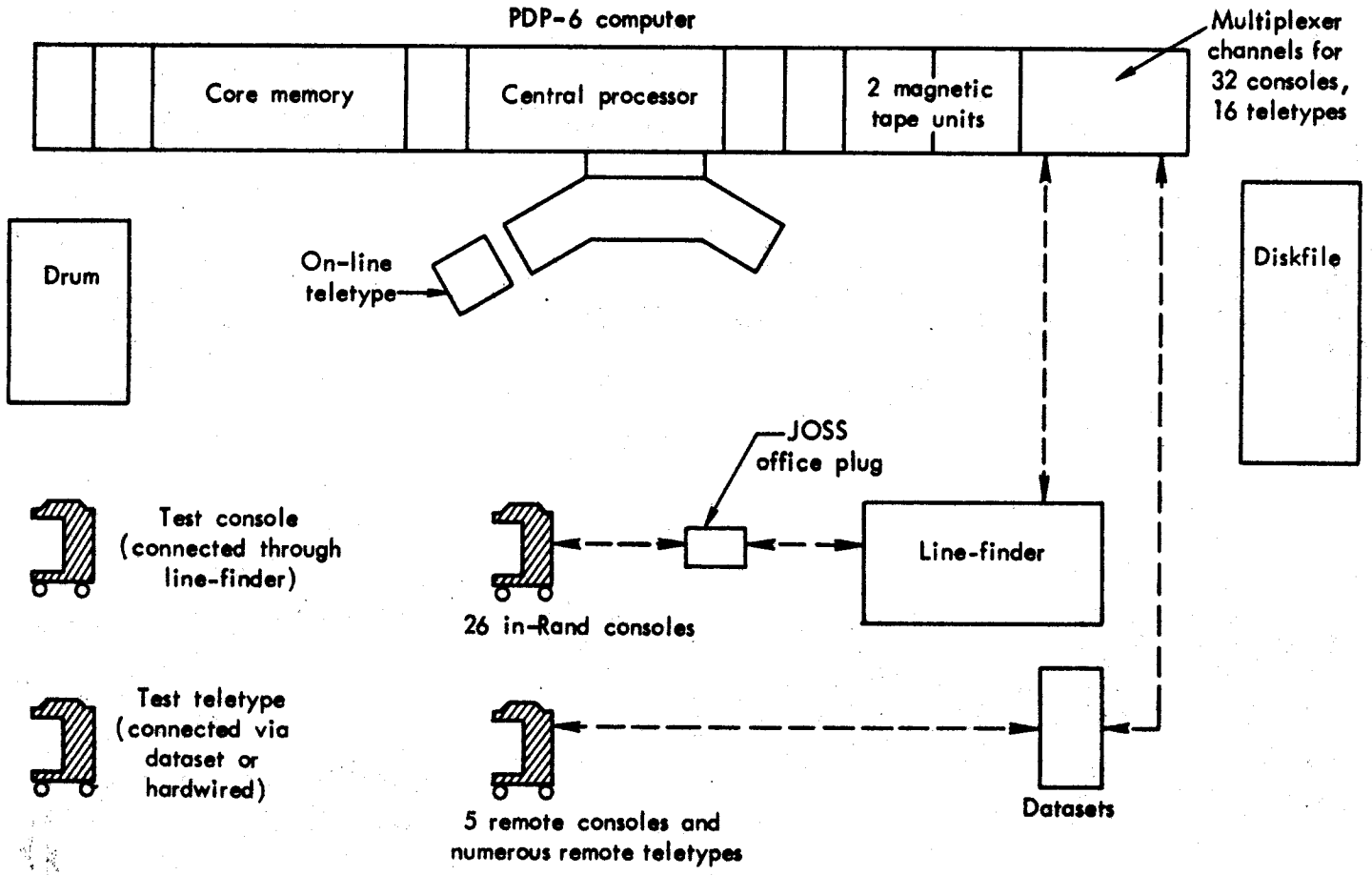


Fig. 1—JOSS hardware elements

the system. One of the two model 33s used for system checkout now plays a role in maintenance, and the sturdier model 35 has become the on-line monitor.

In November 1965, the Air Force Logistics Command at Wright-Patterson Air Force Base accepted Rand's offer of a JOSS station for a trial period of 30-90 days, to acquaint AFLC with the new interactive time-sharing approach to computing. Thus in February 1966, the first remote JOSS console was installed at AFLC's McClellan Air Force Base in Sacramento, California.* Three more were subsequently dispatched—one in August to the Air Force Academy in Colorado, and in September one each to an Air Force studies and analysis section and ARPA in the Pentagon. In December, DEC sold its prototype console to a systems analysis group in the Department of Defense; in August 1967, a sixth console was delivered to SAC in Nebraska.

In February 1967, the first remote teletype connection to JOSS was installed at Langley Air Force Base in Virginia. Originally a temporary substitute for the then-scarce console, this model 35 teletype has remained as Langley's permanent JOSS terminal primarily because of its flexibility in connecting to other systems. Other Air Force sites soon followed Langley's example—namely, Nellis Air Force Base in Nevada, Air Force Cambridge Research Laboratory in Massachusetts, and two Pentagon groups, Air Force Budget and the Air National Guard.

Earlier, in December 1966, a Rand Recommendation to the Air Force proposed that the JOSS experiment be spread to additional USAF sites for the mutual benefit of the Air Force and Rand. Specifically, the Recommendation advocated the procurement of 16 additional JOSS consoles and a corresponding expansion of the PDP-6 multiplexer to handle 16 new console channels. By March 1969, however, negotiations between the Air Force and Rand had reduced recommended consoles to more economical teletypes, and eight teletype channels were appended to the multiplexer. Table 1 shows the Air Force organizations that have participated in the JOSS experiment. Table 2 lists all channel assignments for February 1968 and June 1971.

The November 1966 maintenance schedule fixed hours throughout the five-day week for attention to hardware, files, software, and general system errors. Six hours weekly were assigned to PDP-6 preventive maintenance, with a DEC engineer on call for emergencies. In October 1967, the DEC contract was revised to ensure a resident engineer during the normal workday, as well as a continuation of its two-hour preventive maintenance routine on three weekday evenings. In 1971, the PDP-6 contract reverted to its original on-call provisions for emergencies.

Data Products has continued to service the disk file monthly. The consoles are the joint responsibility of IBM, DEC, and Rand. IBM regularly cares for the Selectric typewriters at all locations. DEC tends to the console electronics where and when needed. Rand repairs paging mechanisms and resolves minor difficulties, including analyzing cross-country complaints by telephone and occasionally exchanging components by Air Express.

*The loyalty of these early Air Force JOSS users is demonstrated by the indefinite extension of the trial period. See Table 1.

Table 1

AIR FORCE JOSS USERS

User	Terminal	Installed	Status, June 1971
McClellan Air Force Base, California	Console	Feb. 1966	In use.
Air Force Academy, Colorado	Console	Aug. 1966	Returned to Rand Santa Monica, May 1969.
AFCSA, Pentagon	Console	Sep. 1966	In use.
ARPA, Pentagon	Console	Sep. 1966	Returned to Rand Washington, Feb. 1970.
OASD, Pentagon	Console	Dec. 1966	Owned and in use.
Langley Air Force Base, Virginia	Teletype	Feb. 1967	In use.
Nellis Air Force Base, Nevada	Teletype	Jul. 1967	In use.
AF Cambridge Research Laboratory, Massachusetts and Arizona	Teletype	ear. 1967	Dormant.
Offutt Air Force Base, Nebraska	Console	Aug. 1967	In use.
AF Budget, Pentagon	Teletype	Sep. 1967	Dormant.
Air National Guard, Pentagon	Teletype	Feb. 1968	In use.
Ent Air Force Base, Colorado	Teletype	Apr. 1969[a]	In use.
Wright-Patterson Air Force Base, Ohio	Teletype	Apr. 1969	In use.
Offutt Air Force Base, Nebraska	Teletype	Jun. 1969	In use.
AFCSA, Pentagon	Teletype	Aug. 1969	In use.
Eglin Air Force Base, Florida	Teletype	Sep. 1969	Terminated June 1971.
AFRDQ, Virginia	Teletype	Sep. 1969	Dormant.
Kirtland Air Force Base, New Mexico	Teletype	Oct. 1969	Terminated June 1971.
OASD, Pentagon	Teletype	Nov. 1969	In use.
AFDSDC, Washington, D.C.	Teletype	Nov. 1969	Dormant.
AF/XOXFA, Pentagon	Teletype	Feb. 1970	In use.
West Coast Study Facility, California	Teletype	Jul. 1970	Dormant.
AFSZ, Wright-Patterson Air Force Base, Ohio	Teletype	Sep. 1970	Terminated June 1971.

[a]The PDP-6 multiplexer was expanded in March 1969 to accommodate 8 additional teletype channels, some of which are shared by several users.

Table 2

PDP-6 CHANNEL ASSIGNMENTS

Channel[a]	February 1968	Channel[a]	June 1971
	Teletype		*Teletype*
00	Hardwired, JOSS staff	00	Hardwired or acoustic coupler, JOSS staff
01	Hardwired, JOSS staff	01	DDD, Rand New York
02	Leased line, Langley AFB	02	Leased line, Langley AFB
03	Prime-TWX, AF Cambridge Research, others	03	DDD rotary[b]
04	DDD, Rand Bethesda, AF Budget, Air National Guard	04	DDD, Rand Washington
05	Prime-TWX, AF Cambridge Research, others	05	DDD rotary[b]
06	Leased line, Nellis AFB	06	Leased line, Nellis AFB
07	DDD, Rand Bethesda, AF Budget, Air National Guard	07	DDD rotary[b]
		10	DDD rotary[b]
		11	Western Union, Air Defense Command
		12	Leased line, Offutt AFB
	Console	13	Western Union, Kirtland AFB, Eglin AFB
10-31	JOSS plug, Rand Santa Monica	14-16	DDD rotary[b]
32	Leased line, Offutt AFB	17	DDD, Rand Santa Monica
33	Western Union, McClellan AFB		
34-36	JOSS plug, Rand Santa Monica		*Console*
37	Western Union, AF Academy	20-32	JOSS plug, Rand Santa Monica
40-41	JOSS plug, Rand Santa Monica	33	Western Union, McClellan AFB
42	DDD, OASD	34-36	JOSS plug, Rand Santa Monica
43	DDD, AFCSA	37	Leased line, Offutt AFB
44-46	JOSS plug, Rand Santa Monica	40-41	JOSS plug, Rand Santa Monica
47	DDD, ARPA	42	DDD, OASD
50-77[c]	--	43	DDD, AFCSA
		44-46	JOSS plug, Rand Santa Monica
		47	DDD, Rand Washington
		50-57	JOSS plug, Rand Santa Monica
		60-77[c]	--

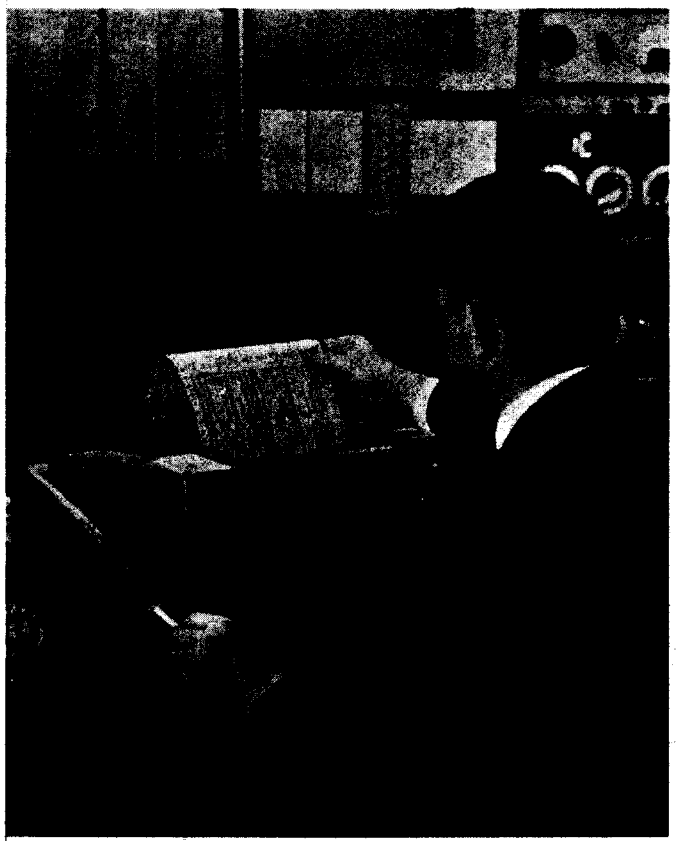
[a]Channels are numbered in octal rather than decimal.

[b]DDD teletype rotary assignments include AF Cambridge Research, AF Budget, OASD, AFCSA, Air National Guard, AF Avionics Laboratory, AF/XOXFA, AFRDQ, AFDSDC, West Coast Study Facility, and ASZV.

[c]Available for expansion to 64 channels.

Hardware failures have been memorable probably because of their relative infrequency. In 1967, multiplexer errors were relieved by adding cooling fans. Chronic PDP-6 ailments during the last half of 1968 culminated in a disastrous loss of files, but led to improved maintenance procedures. Several elements of the system were rejuvenated during 1970. The disk file received a major overhaul. A third of the Selectrics were rebuilt, and all were slowed to an output pace more becoming their years. A thermal cutoff was installed to turn off PDP-6 room electricity when a rising temperature threatens damage to computer circuits. Thus far in 1971, the hardware has resisted damage from large quakes and small visitors.

A more typical record of the daily JOSS routine is displayed at the PDP-6 teletype monitor. Each minute that JOSS is operating, the system prints out a line of figures describing the activity of users, consoles, and system during that minute. Once every four hours, a paragraph of statistics appears. This "on-line log" informs the observer of an occasional data transmission error, a lone user at an early



G. E. Bryan at the On-Line Teletype

morning hour, the length of queues awaiting rush-hour access to disk file or computation, the number of users thinking in "green state" (that period during which the console's green light indicates the typist is in control).

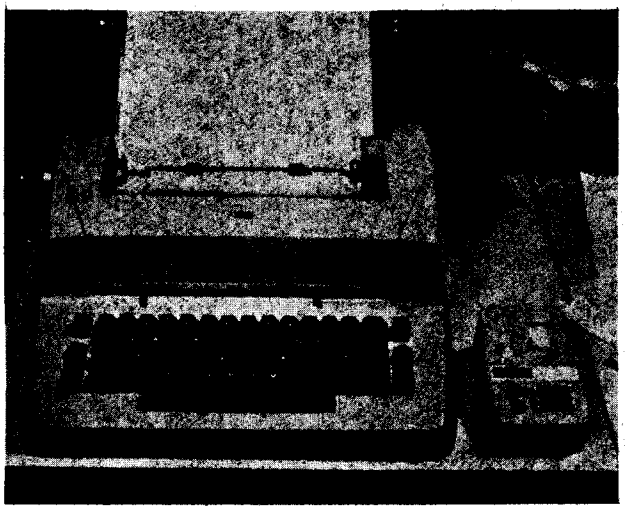
On a rare day, lines of printout may indicate that one or more users have been "gronked," that is, that a major error of console or system has discontinued their service. G. E. Bryan explained the origin of "gronk" as the earth-and-sky-shattering utterance of a brontosaur in the comic strip "B. C." This agonizing expression seemed appropriate to the unexpected interruption of a long calculation. The system usually recovers of its own accord after such a disruption, typing at each affected console the courteous JOSS greeting. Repeated gronks appearing on the log signal an unscheduled maintenance period.

Good system design prevents most failures from being catastrophic. For instance, software and hardware combine to protect areas of memory from user attack, so that no one can enter either another user's work space or the section of memory occupied by JOSS itself. Certain disk failures are sensed by the software in time to inhibit access to files and to notify users at their consoles of this action. Such a condition must be reset manually by the engineer. Other disk file defenses are furnished by software and operational procedures.

THE CONSOLE

Versions I and II of the JOSS console were created for the occasional user, who has little time to spend learning or relearning complex controls. The console was intended as a personal instrument located handily in an office, available for problem solving on line or simple typing off line. Much care went into choosing and arranging the characters on the keyboard to take advantage of average typing skills. Although the teletype is readily obtainable and relatively inexpensive, it was rejected as a possible terminal principally because its telecommunication functions are a mystery to the untrained person.

Meredith Westfall at the JOSS I Console



The JOSS II Typewriter and Control Box

Various features were incorporated in the JOSS II console to enhance the user's comfort and convenience. It was made both mobile and stable, passing easily through office doorways and fitting nicely into typical office areas. To encourage mobility, special JOSS plugs in more than 200 offices and public rooms connect a console, through the basement line-finder, to an available computer channel. A small trapezoidal table can be attached on the right or left, in either of two directions, or can hang vertically when not needed. Additional space accommodates pencils, coffee cup, and ample room for knees and feet.

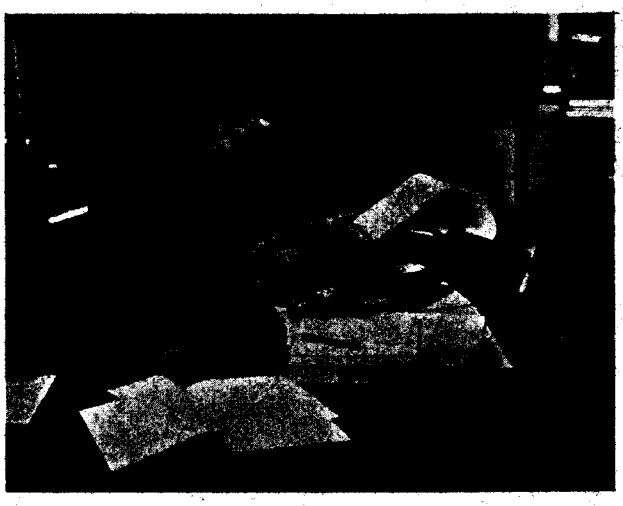
To satisfy input and output requirements, a hopper and stacker for standard fan-fold paper project from the rear. When tear strips containing sprocket holes are removed, the resulting 8½-by-11-inch page is suitable for binding or reproducing. A Rand-created paging mechanism can be operated manually by key or automatically by command to space to the top of the following page in a single motion. Tabbing facilitates the typing of tables, and an elite type font blends with the text style of most Rand publications.

JOSS users generally find the typewriter familiar, its touch responsive, its working environment comfortable. They also occasionally have had cause for complaint. Although the removable side wing is convenient for holding a small amount of reference material, such as a single notebook, it has proved inadequate for a large book of computer listings or some other bulky volume. Since the console can easily be moved near a desk or table, this detachable work space may be more a luxury than a necessity in an economy model. It is also a hazard when children are tempted by its solid appearance to sit on it.

The routine for changing paper is not self-evident in the JOSS tradition. The paper hopper and stacker are certainly well constructed to feed and accumulate

C. L. Baker Wheeling Console into Office

C. L. Baker Connecting JOSS Console



C. L. Baker Working at JOSS Console

paper. Yet someone who has not referred to the step-by-step guide in the *JOSS Notebook* may neglect to restore a lever or knob, or insert the paper incorrectly or carelessly. Actually, the paper might as well be supplied directly from its shipping carton resting behind the console. It would be changed less often; and a less obtrusive stacker would permit the console to stand closer to a wall. The present device also necessitates moving the console away from a wall when opening the hopper to the rear.

The paging mechanism has worked well in most of the JOSS typewriters, but a few have malfunctioned chronically. The mechanism was added to the typewriter as an improvement over the line-by-line advance of the Selectric's index feature, but has been plagued by worn microswitches and idler gears, misaligned shafts, accumulated grease, and incorrect paper pressure. Most JOSS users have learned to turn a platen knob to aid a slipping clutch trying to catch. Other paging problems have generally been solved by simple maintenance adjustments. The question of whether the typewriter vendor makes these simple adjustments to this noncommercial, nonstandard mechanism, or the console vendor, or a Rand technician, has been a point of controversy.

A poll of potential users favored an elite type font over pica, not only to crowd more characters into a single-line JOSS statement, but also to match the type used in most Rand publications. However, the elite font type ball proved to be sensitive to worn ribbons and misadjustments, so that care is required to prepare good reproducible output. A new ribbon and type ball help, as does carbon-backed vellum substituted for the standard JOSS paper. The green of the JOSS two-color ribbon reproduces as a mottled grey. Alternatively, an all-black ribbon provides a vellum from which a printer can make separate plates for two-color reproduction.

The console's mobility has brought the computer's power where it is needed, either by moving the terminal from one room to another or by relocating it within a room. Mobility has been less of an advantage when it has enticed users to make unlimited demands on the limited number of consoles.

SOFTWARE AND OPERATIONAL PROCEDURES

One pride of JOHNNIAC/JOSS was the language that was simple enough to be summarized on a single 8½-by-11-inch page. PDP-6/JOSS still adheres to the principles that the language be readable, learned quickly, and used easily even if infrequently. Some of the original language elements were expanded, as in the commands for *Do* and *Type* and *Cancel*, the use of additional logical and conditional expressions, the concept of the sparse array, and a more flexible format for the *max* and *min* functions. This format became an option in the new functions of *sum, prod, conj,* and *disj*. The lengthened function list was completed by *first* and *tv*. The verb *Let* was added to permit the user to define his own functions as formulas. A set of new verbs enables the user to communicate with the disk file to store and retrieve programs. Appendix A illustrates the expansion from an 8½-by-11 page to one of 16½-by-11.

Software modifications to PDP-6/JOSS have been deliberately minimal, partly because of the limited programming support, but also to avoid additions that might violate the principles of the "easy-to-learn and easy-to-use" language. Changes generally resulted from operational pressures. For example, the need for a program library led to the introduction of the "read-only" library file, whose items can be recalled as from personal or public files, but which are guarded from alteration by users. Another software change permits annotation of a stored program by including steps to describe its operation. Such descriptive steps appear when the program is typed at a console, but are ignored by the system during the program's execution, and are particularly helpful in using library programs.

A modified Rand accounting system required a revised JOSS log-on and an accounting algorithm that no longer calculates charges for items stored in files. Although files are now free of charge, users are still encouraged to purge dormant items to conserve the limited disk space. To help identify dormant items, two columns were added to a file's item-list, which exhibit on command the date each item was filed and the number of times it has been used since being filed. These software changes are summarized in Appendix A.

Because the system requires no operator in constant attendance, manual procedures are limited essentially to shutdown, restart, and file maintenance. Each time JOSS is shut down for servicing, the accounting tapes are removed for later process-

ing by Rand's main computer center, which prepares a variety of computer usage reports. From time to time, the operating statistics produced at the on-line teletype are bound in monthly volumes.

Three evenings per week after the two-hour hardware maintenance, the contents of the disk file are transferred to a set of magnetic tapes. After the tapes are verified as being a correct copy of the disk, they are entered in the collection of the five most recent sets. These backup tapes are used to restore the disk file whenever it has been altered, either intentionally or inadvertently. The remainder of the file maintenance period is used to satisfy user requests to assign or close files, to punch individual items on cards, or to reload items previously punched.

One purpose of the punched-card backup for file items is to preserve dormant programs, which can then be removed from the crowded disk. The owner may later request that the items be reloaded from his cards. The punched-card backup also protects programs that are difficult to recreate at the keyboard in the event of a catastrophic loss of files—that is, if the disk file and all sets of backup tapes prove unusable.

Such a severe loss has occurred twice, both times unfortunately before the punched-card service had been implemented. On 3 November 1966, a memo from I. D. Greenwald informed JOSS users that files had been ruined the previous evening —not exactly a rare event in those early disk days. More critical, he reported, was the fact that all magnetic-tape copies were bad. Most of the night had been spent attempting to reconstitute the files from the best of them. Greenwald described additional software error checks that would be implemented to avoid future errors.

On 26 November 1968, after several months of hardware troubles, JOSS users were notified of another major file loss. All items stored in files since 7 November had been destroyed, and items stored prior to the 7th were most likely incomplete. Programs for which users had saved printed copies were laboriously revived by retyping into JOSS. The human and procedural errors were carefully examined and steps were planned to avoid a recurrence.

LANGUAGE

Learning JOSS often proves simpler for the nonprofessional than for the professional programmer. The latter is tempted to look for analogies between, say, the JOSS *part* and the FORTRAN subroutine. Not only do such comparisons bring forbidden jargon into the open shop, but they can be misleading as well.

The verbs *Set* and *Let* illustrate the confusion. The JOSS *Set* assigns values to scalars or elements of an array. The JOSS *Let* defines formulas, usually as functions of parameters. The programmer who has used the conversational language BASIC observes that the BASIC *Let* works somewhat like the JOSS *Set*. He may also be aware that numerous programming languages define a subroutine of many statements much as the JOSS *Let* defines a single-line formula. But none of these similarities will particularly help him to understand the fundamental organization of a JOSS program—that is, the hierarchy of parts, each initiated by a *Do* command.

The JOSS programming style seems elusive to some because it is so seldom on view. Alone with his personal tool, each user composes his program unaware of any standard of "good" JOSS. It is only when he voluntarily seeks help or distributes his program to others that he may reveal a poor use of the grammar or an inefficiency in construction. Problem areas thus discovered were often discussed for the benefit of all users in the *JOSS Newsletter* until its discontinuation in June 1971. *Newsletter* items differentiated between *Set* and *Let*, *Do* and *To*, and *Done* and *Quit* and *Stop* and *Cancel*. Others simplified space and size, arrays, and functions of functions.

The popular readability of the language—the English-like punctuation, the full words, the mathematical symbols arranged in the pattern of a handwritten formula —improves typeability as well as comprehension. Brackets, parentheses, and spaces set off the words and symbols to clarify meaning for the typist as well as for the software.

A newcomer from the world of computer-oriented languages sometimes asks why he must end a statement with a period, instead of merely pressing the RETURN key, which can accomplish the same end within the software. For one thing, the period helps the nonprogrammer to distinguish a command from an algebraic expression. For another, the similarity of JOSS language rules to those of the English language makes it easier for the computer novice to learn them.

The single line of a JOSS statement seems limiting at times when compared

with the facility of other languages to continue statements on following lines or to allow more than one instruction per line. The single line proves invaluable, however, when a user needs to know exactly where an interruption has left his calculation. A statement that is too long for a single line can be divided into several statements, and a lengthy function can be defined as a simpler one in terms of several others.

The reasons for certain spacing rules may not be as self-evident as JOSS was meant to be. For instance, a space is forbidden between a function name and the left parenthesis preceding the argument; a space is required between a file number and the left parenthesis preceding its identification code. Violations produce an admonishing *Eh?*. However, the freedom to space between words in a statement improves the line's appearance and reduces eyestrain in complex sequences.

Another favorite JOSS characteristic is ease of modification. Not only can steps, forms, formulas, and values be readily replaced, deleted, or added at any time during the creation of a program, but the current state of the program can be displayed in selected portions or in its entirety at a convenient interruption. After modifications are completed and control is returned to the system, JOSS reinterprets the interrupted step from its left-most character.

The language includes a collection of built-in functions that aid in the solution of numerical problems. There are functions that dissect a number into its component integer, fraction, digit, exponent, sign, and absolute value. Others calculate for an appropriate argument the square root, sine, cosine, logarithm, exponential, and arc tangent. Still others compute over a prescribed range of variable the sum, product, minimum, or maximum of an expression, or the first value in the range for which a condition is satisfied. The format of these last functions follows that of standard mathematical notation, with adjustments for expressing subscripts and superscripts in the single-line JOSS fashion.

Three other functions—*conj, disj,* and *tv*—depart from Shaw's initial JOSS philosophy. The JOSS I design intent was that all language features would be understood by all users, whatever their computer experience. Shaw realized that there would be elaborations of the language in transferring the system from the JOHNNIAC to the PDP-6. But he urged that extensions maintain the spirit of the original, so that no user would be excluded from acquaintance with any JOSS feature. JOSS II designers favored an elementary language as a subset of a larger grammar, thus providing an extended JOSS for those who required its capabilities. These three extended functions are concerned with the truth values *true* and *false,* which make possible user-defined functions that involve both propositions (which have truth values) as well as expressions (which have numerical values). This logical capability has seen limited use.

Everyday JOSS usage combines JOSS-supplied functions with the usual mathematical symbols to define a special-purpose function or an expression in a program step. Such compact descriptions of calculational procedures are not only easy to read in the normal left-to-right manner of mathematics, but are also simpler to comprehend at a glance than the corresponding multiple instructions in a machine-oriented language. When such functions or expressions have been of general interest, they have been publicized in the *Newsletter* or the *JOSS Program Library*

Catalog, and have included formulas for modular and complex arithmetic, numerical integration, and rounding of intermediate values.

The demonstration file of the library provides a programming guide with examples of the conditional expression, compact formula definitions, and some of the less frequently used functions and commands. The command to *Type all* reveals not only elements of the language but also the organization of a program.

One feature missing from the language and most often requested is a function to generate a random number. A built-in random number generator would have introduced ambiguities whenever the function was referred to more than once in a single statement. Other difficulties would have arisen from the necessity to repeat a run of random numbers in certain calculations. Instead, JOSS users have attempted to generate random numbers by combinations of commands and formulas, usually involving a multiplicative procedure. Some maintain that only a small sample of approved numbers can be produced by such methods, because JOSS multiplication rounds to a single precision result. Others have suggested a scheme that they find satisfies statistical tests and provides ample numbers for most studies. Still other users prefer to equate intricacy with randomness in choosing a method, and even employ such variable quantities as the current session time to add a "random" note of mystery.

The facility for manipulating strings of characters would have been a major programming development when JOSS II was being implemented, because it would have required a redesign of the language and manpower beyond the scope of the project.

USERS

The typical JOSS user may never have solved a problem by computer before, or perhaps has done so only with the help of a professional programmer. He is attracted by the availability of the JOSS system, the simplicity of its language, and the informal interaction that permits prompt correction of errors. As he creates a program in the familiar language of his discipline, he is shielded from the unfamiliar sight and complexity of the computer.

JOSS also appeals to the professional programmer, who knows well the pitfalls of machine language and the frustrations of formal procedure. Both novice and professional consider JOSS a helpful assistant, a tool that serves the user rather than demanding a rigid obedience. Various methods of communication among users and staff augment this informal service.

The most immediate and direct communication is the message, by telephone or through JOSS itself. One telephone number connects a user-in-need with a technician in the PDP-6 room or with an electronic secretary in his absence. Another number activates a recording that recites the current status of the system—e.g., up, down, marginal, expected recess. A third number rings if JOSS is operating, and gives a busy signal if not. These three responses are available at all hours to all users.

One of the ten public JOSS files is designated as the Mailbox, which is reserved for messages. Every file may contain up to 25 items, each of which may represent an entire program or any combination of program elements. A Mailbox message can be addressed to an individual or to all users or the JOSS staff by appropriate choice of the item's five-character identification code. For example, the index of items for the Mailbox file might indicate a message for John Jones as item 5 (JONES). To read the message, Jones would recall item 5 and instruct JOSS to *Do part 1.* When JOSS had typed the message, Jones would remove the program steps that produced it by discarding item 5 (JONES) from the Mailbox file. JOSS personnel check the Mailbox daily, responding to trouble reports or requests, and occasionally expediting messages between remote users and colleagues at Rand.

The standard heading that appears at the top of each page of JOSS printout includes space for a short administrative message, which a technician inserts at the on-line teletype in the PDP-6 room. A message continues to be printed in the heading until it is manually removed at the PDP-6. For instance, normal evening recesses

are publicized in the page heading three afternoons per week, beginning perhaps two hours before the 5 p.m. shutdown. There is a shutdown procedure in the JOSS software that prevents new users from logging on and prepares accounting tapes for removal. Once the procedure has been manually initiated for normal or emergency maintenance, JOSS transmits the message *System shutting down* to the page heading, and beeps every connected console five times at the start of each minute to urge users to advance to the next page of their printout to read the heading. Occasionally, a heading note alerts users to look for a more elaborate message in the Mailbox file.

The monthly *JOSS Newsletter* was a more general means of exchange, covering topics ranging from language to hardware, many of them arising from questions asked by users. The first issue appeared in November 1967, and subsequent issues were bound each six months in a cumulative volume to simplify distribution of back copies. In June 1971, the final issue of the *Newsletter* was published; in August, a cumulative binding of the entire series appeared [1].

Another document, the *JOSS Program Library Catalog*, was introduced in November 1969 to inform users about programs stored in the library files. Annotated programs accepted for the library are abstracted in the *Catalog*, which has been updated and distributed with the *Newsletter* whenever changes warranted. The March 1971 edition is the final version of the *Catalog*, although programs submitted for the read-only protection of the library files are still accepted, with the understanding that publicity for the programs is the individual's responsibility.

A series of JOSS publications dates from the earliest years to the present. J. C. Shaw described his design principles for JOSS I in 1964 and 1965 [2-5]. C. L. Baker, J. W. Smith, I. D. Greenwald, and G. E. Bryan detailed JOSS II in 1966 and 1967 [6-15]. Also in 1966 and 1967, several user documents complemented those intended mostly for system designers [16-21].

Beginners follow a primer while seated at a console, and progress according to their interest and experience. JOSS users with questions are encouraged to explore on their own, to discover "what happens when" by experimenting and reading. Always at hand are more experienced users to suggest an answer or a path to one. In particular, JOSS Representatives have been appointed in most Rand departments and at each remote installation to aid local questioners.

In 1966, Bryan reported that 300 different people used JOSS each month. By the end of 1970, there were perhaps 500 to 600, a quarter of them some distance from Santa Monica. One-hundred-fifty sessions per day in 1966 compare with about 140 in 1970, with peak usage still at mid-morning and mid-afternoon. A casual survey of the session logs shows a repeated use by a few and occasional use by many. The average number sharing JOSS during any minute of a normal workday remains at about 13. The time all users spend actually computing (as opposed to thinking about it while JOSS is servicing other console requests) has risen from 130 hours per month to about 200. However, computing has fallen from 18 to 7 percent of total session time.

The figures seem to indicate an increase in the number of occasional users, and also in the amount of "green time," or that period at a console when the user rather than JOSS controls the typewriter. More people may be thinking at their consoles,

or possibly some teletype users are running up their session time by forgetting to log off JOSS before disconnecting their data communications. Regardless of the total usage figures, however, 5 percent of a typical 45-minute session is devoted to JOSS computation.

In 1967, approximately 20 percent of the disk file was occupied by items in 330 personal and public files. By 1971, items in 770 personal and public files and 13 library files filled 91 percent of the disk. A vigorous campaign to edit personal and department files forestalled a space crisis.

Before files were implemented in the fall of 1966, every program had to be reproduced at a console keyboard each session it was used. Once programs could be saved between sessions, lengthy logic and more intricate input and output were practical. It became reasonable to construct a program that conceals its convolutions from the program's user much as the intricacies of the JOSS software are hidden from and inaccessible to the JOSS user.

For example, a person having no computer experience can interact with a program in his field of interest entirely in the terminology of that discipline. Such a program might be a model of a real-life situation, permitting the user to investigate selected actions under a variety of circumstances, all in a conversational manner. Just as JOSS responds helpfully to errors on the programming level, the well-planned JOSS model permits the modeler to correct his typing errors or to modify his decisions, and then continue.

JOSS users were invited in September 1968 to submit programs for the newly established program library. By the spring of 1971, the read-only files contained routines to fit curves and fly rockets, to invert matrices and calculate probabilities, to compute costs and find roots, to simulate campaigns and evaluate systems. Remaining in personal files are specialized routines known only to the file owners and their confidantes.

The first library file is reserved for demonstration programs. Among them, a quiz supplies 20 questions on JOSS topics, a choice of answers, and an explanation of each correct one. A mortgage amortization schedule illustrates JOSS report-quality printout. For the mathematically inclined, JOSS rapidly factors selected integers into primes or, more slowly, plots the shape of y as a specified function of x. A calendar is printed for an input month and year; a formula in the calendar program represents "Thirty days hath September . . ." in logical meter. Displaying JOSS interaction are the games of blackjack, hangman, and tic-tac-toe in two and three dimensions.

Early visitors to JOSS were mainly computer scientists anxious to observe the latest in man-machine communication. As conversational computing gained acceptance, visitors included more potential users and fewer creators of such systems. From foreign dignitaries to children of all ages, they have been strangers to the computer and, in many cases, have had a limited acquaintance with English and mathematics. All have been delighted with the JOSS *Eh?* when it doesn't understand what they have typed.

Student groups have come from universities down through the elementary grades. Young people have represented science clubs, computer courses, honor socie-

ties, scouts, classes at an advanced level and some receiving remedial tutoring. During each visit, JOSS is introduced as a helpful tool accessible via simple English and familiar mathematics. The youngsters take turns at an array of consoles, discovering for themselves the informality permitted within the prescribed rules of language.

JOSS has proved to be an excellent educational tool because of the immediacy of answers, and the fact that almost every user action, whether correct or not, can stimulate questions or discussion or exploration at the console. Topics range from mathematics to language, and vary interestingly with the age of the visitor. For example, the seventh-grader learns new words like "delete" and "imperative," and satisfies himself with a square root by squaring it. The ninth-grade student discovers "precision" and "accuracy" by example, and why JOSS is "fast" conversationally and "slow" computationally. The 18-year-old wonders how to generate random numbers and experiments with recursion.

MEETING AND INTERACTING WITH THE SYSTEM

JOSS appears physically to the beginner as an electric typewriter and a small control box. He hunts for a way to enter the system, and discovers either a blue rocker switch at the keyboard (labeled ON-OFF), or a white rocker switch on the control box (labeled POWER ON-POWER OFF). If he presses the first, he soon finds that he has an ordinary typewriter at his disposal, and nothing more. If he presses the second switch, the JOSS greeting initiates the log-on procedure. In this way, the newcomer learns the first JOSS lesson, that interaction by trial and error is safe and rewarding.

To log on, the user responds to English language requests for initials, department name, and project number. If he wonders whether he should identify himself in upper or lower case, with or without periods or spaces, he discovers that any combination of these is accepted by the system. However, if he types a nonexistent department or project number, JOSS objects firmly, with a hint of how to improve.

Logging on to JOSS thus presents the beginner with a first taste of the system's discipline, and the freedom within its boundaries. One should be able to greet the computer simply and briefly, without having to learn or remember a complicated protocol. On the other hand, the user should have to conform to a minimum set of rules that protect him from complexity and ambiguity, while enabling him to proceed reasonably from error to correction.

Conversation involves an interchange in which each participant takes turns speaking and listening. JOSS indicates conversational turns in several ways: the system is in control when the red light is on, the printing is black, and the keyboard is locked; the user is in control when a beep sounds, the green light is on, the printing is green, and the keyboard is unlocked. The user may regain control manually by pressing the INTERRUPT button, which JOSS then back-lights in amber. The status of errors or intentional interruptions is indicated by appropriate typed messages.

Quick response is fundamental to conversational JOSS, where a user judges speed by how long it takes him to regain control of the typewriter, rather than by how fast JOSS calculates. The interpretive nature of the JOSS language sacrifices computational speed for the ability to interrupt, modify, and continue. Specifically, a JOSS command is stored in the user's work space essentially in its typed form, and interpreted character by character each time it is executed. A language not designed

for interactive use translates each typed command into machine language instructions for more efficient execution. The repeated interpretation of a JOSS step adds to the program's execution time, but makes possible the typical JOSS interaction.

The manner in which a program is created by the discovery and correction of errors illustrates even more than do the physical indicators how a human "converses" with the machine. Commands are typed and obeyed immediately and forgotten, or are stored as labeled steps to be initiated later by a direct command. Interaction seems most responsive when all commands are direct, so that results or error messages appear promptly on the next line. The give and take of a session is more typical, however, when a user experiments with his program's logic—trying this, learning of an error, correcting that, and continuing his exploration from the point of error.

The readability of the commands and messages contributes both to understanding and ease of correction. Except for the laconic *Eh?* indicating an error in spelling or spacing or vocabulary, an error statement is a single English sentence that identifies exactly what and where the error is. The offending statement is either a direct command on the previous line, a stored step identified in the message by its label, or a formula. In any case, control is returned to the user for whatever action he chooses. He may retype a line and continue from the point of interruption, resume at some prearranged place, or simply begin anew. Most often he types *Go* to continue, knowing that JOSS will reinterpret the interrupted step. He may voluntarily halt his program by pressing the INTERRUPT button, then examine his calculations and proceed, stopping and starting without ever becoming entangled in obscure messages or being lost irretrievably within the software.

JOSS users seem more sensitive to interruptions of service than do those who submit computer runs over a counter. The professional programmer who reads a sign saying the computer is down merely adds some hours to the normal turnaround time, and goes away without further question. He understands computer malfunctions and how they can affect the running of his job.

The typical JOSS user, on the other hand, brings to the console an innate distrust of the machinery beyond his view. He has been encouraged to experiment, assured that his explorations will never lead him into unfriendly territory. Then suddenly the INTERRUPT button does not interrupt. Or the JOSS greeting appears in the middle of his computation to tell him that either the system or his terminal has momentarily failed. Or a repeated beeping directs him to press the PAGE button to read an emergency heading message.

The user can handle such situations in several different ways, depending on the information available to him. If the red light is on, JOSS is presumably still working for him. If he is accessing a file, he knows that he might be in a long disk queue, so he generally waits awhile before pressing INTERRUPT and losing his place in line. If he really suspects an error, he immediately presses INTERRUPT to return to green state.

The green light, a beep, and a message would inform him that he has control to investigate his suspicions. However, if the INTERRUPT button does not turn to amber when pressed and the red light persists, the trouble is in the system. In this

case, he would dial extension 233 for a recorded status message or, more likely, extension 415 to ask a technician if JOSS is down. He worries about whether the system will be down long or if files are threatened. Whatever his experience, he requires some human reassurance.

If the maintenance technician already suspects a general problem, because of user reports or observation of the on-line log, he inserts a SYSTEM MARGINAL message in the page heading, as a low-key warning. Users understand that they should keep their sessions short and file their programs at intervals until the all-clear announcement. When evidence accumulates, the technician must decide whether to wait until the engineer performs the next scheduled maintenance or to take the machine down during prime time. He knows that the latter is unnerving, but that an unreliable system weakens users' confidence even more.

If shutdown is appropriate, he places a suitable message in the heading, activates the emergency series of beeps to attract users' attention to it, then records a telephone message giving the current status and predicted delay. He is now free to diagnose and repair.

During an extended period of downtime, the recorded status message is updated every few hours, or whenever new information is available. Facts are presented briefly with a minimum of technical detail. Prior to June 1971, especially serious malfunctions were explained more elaborately in a *Newsletter* item, with a summary of work performed and planned.

Preventive or corrective maintenance is scheduled with the users' needs in mind. Occasionally someone will request an adjustment of the normal evening schedule, but all such deviations from custom are advertised as far in advance as possible. For example, a *JOSS Flash* might be distributed a week ahead of time to announce that JOSS will be down for the day to service the drum. An afternoon's heading message might alert users to an early recess for monthly disk maintenance. In return, users are urged to notify the staff of any priority needs, so that maintenance can be arranged compatibly.

Although every effort is made to minimize downtime, it is apparent that the system and its hardware elements were designed to support an experimental research environment rather than an operational one. For instance, JOSS is without the hardware redundancy common to many commercial installations. Thus the system is totally down while a major component is being serviced.

Further, diagnosis may require more than one of the several vendors, so that determining responsibilities and coordinating repair activities consumes valuable time. The PDP-6 and disk file engineers may have to consult when it isn't clear which of these two pieces of hardware was the source of error. A console failure may involve the vendors of the typewriter and console electronics, as well as Rand representing the noncommercial paging mechanism.

Moreover, the experimenters were unconcerned with the small number of PDP-6 installations elsewhere. During an operational crisis, however, when machine problems persist despite the efforts of the local vendor office, an extra source of engineering expertise is essential. The electromechanical line-finder plays a rela-

tively minor role in the experiment, yet the absence of solid state electronics in its design has required considerable human intervention to unstick sticky relays.

Despite the innumerable ways a complex system can fail and the many sources of delay in restoring service, JOSS has had remarkably little downtime, the year 1968 notwithstanding. That year and the years since have demonstrated the value of good communication among users and staff.

ALLOCATION OF RESOURCES

When JOSS I was implemented on the aging, home-built JOHNNIAC, the cost of operation was essentially that of the electricity to run the computer. Not even a minimal charge was made to any of the selected Rand researchers who helped evaluate that first version. It is not surprising, therefore, that the word "free" became associated with JOSS and has continued to influence its use.

The JOSS II hardware is owned by the Air Force and maintained by contracted vendors and Rand. JOSS II rates, which are calculated to recover yearly overhead costs, are charged only for actual computation. There is no charge for the green time during which the user thinks at his console or for space used to store his programs between sessions. Thus his typical 45-minute session costs less than $2.00 for an average 2 minutes of computing by an average-sized program—although if he is not in Santa Monica, he must add the price of data communications.

Certainly the person who considers JOSS essentially free, as many Rand and Air Force personnel do, will use the system differently from those who receive monthly bills from a commercial time-sharing service. Because he doesn't stand in line at a window to turn in his computer run, the JOSS user is seldom aware of the needs of others sharing JOSS, and senses their presence only when he has to wait in a disk or compute queue. His attitude toward time and space and equipment, and sharing these resources, depends more on whether his program runs too long or whether he can't find a console than on the modest month-end dollar summary.

Since even a maximum-sized program doing an average computation costs so little to run, the JOSS user has little incentive to reduce his program's magnitude. Instructions and data may well fill up the 4000 machine words of his maximum core block and spill over into the files if he decides to overlay portions during execution. He then pays a penalty in speed, since a large program is usually switched to the drum more often when competing with small programs for core space. Time is wasted for all those in the disk queue when a large program must pull itself in by bits and pieces from files.

Some large programs are initially a manageable size, and grow by elaboration to tax the available space and service. Then arises the dilemma of whether to struggle with the handicaps of a too-spacious code, or begin again on another roomier system, perhaps without an interactive language. If the program is a model,

the user will probably remain in interactive JOSS, even with its storage limitations and slow file response.

Not charging for file storage discourages the housecleaning habit. As evidence, of the 44 issues of the *Newsletter* distributed between November 1967 and June 1971, exactly one-fourth pleaded with users to edit personal and public files. The fraction of disk occupied reached 91 percent in February 1971, declining by 11 percent only after an all-out campaign to transfer dormant items to punched cards. Public files, intended for temporary storage as a supplement to personal and department files, have become rest homes for aged programs. Personal files of departed staff are seldom cleared and closed.

The console's mobility is one of its attractive features, except when it contributes to console scarcity. In addition to the 11 terminals designated as public in advertised locations, there are 15 assigned to various departments to move from office to office. These officially mobile consoles are allocated on the basis of past usage and expected need, and are under the supervision of the department JOSS Representatives. Each Representative establishes his own priority scheme. Some require a signup list and a known parking place for the console between office uses. Others prefer informal agreements and no fixed locations.

Allocation problems arise when there are pressures of deadlines, special demonstrations, or simply a program that must be run in a private office. If a department cannot fill a request, the JOSS staff tries to borrow a terminal from another department. The danger of "bulb-snatching" is obvious, unless each request is examined relative to the requirements of all users. This may involve an unwelcome inquiry into JOSS use. Must a demonstration occur during a peak JOSS hour? Why won't public terminals serve as well as a special setup? Why does a program need the privacy of an office? Will the console remain idle for long periods between uses, and thus out of circulation in a place unknown to many?

An inappropriate JOSS application can also strain the supply of consoles. Ordinarily a user has the freedom to program JOSS in his own, perhaps inefficient way. However, he who selects JOSS instead of another system also elects to share its facilities. Suppose he has written his program in such a way that a typist must supply a value once every 10 minutes for an extended period. Then he not only monopolizes a console unfairly, but also places the typist in a mechanical, noninteractive role. Moreover, a code so obviously devoted to computation rather than conversation will run less economically in the interpretive language of JOSS than in the more efficient compiled machine language of a production-oriented system.

Only because the number of consoles is limited does anyone ever examine a program to see if it is JOSS-like—small, interactive, and numerical. Only when a user complains that his program seems to run more slowly than at a previous time are compute and disk queues examined on the on-line log. When three magnetic tapes instead of one are needed to back up the disk file, then file owners are urged to edit. It is the pressures of allocating resources that lead to occasional violation of the JOSS user's privacy.

THE IDEAL AND THE REAL

Familiar console, simple language, and conversational approach are the basis of experimental JOSS, created to probe the relation between human and machine. JOSS was designed as a personal tool to please an individual, without considering the real-world competition for service among many users. Ideally, there should be as many consoles and corresponding computer channels as there are people wishing to calculate at the same time. In fact, there should be a sufficient number of consoles to allow one to remain idle in a user's office until he next requires it. No JOSS "expert" should ever examine the propriety of an application or the efficiency of a program or the correctness of the grammar. Periodic editing of files should be unnecessary. Such a utopia exists for the single JOSS user only when space, time, and equipment are ample for all.

JOSS was designed for solution of the small numerical problem. Since small programs infrequently call on files, most users are willing to accept an occasional wait for file service. Therefore, JOSS II design ignored speedy access to files; requests are queued rather than time-shared. The same design philosophy minimized the core space occupied by the file service routines, in favor of more room for user programs and less time spent switching programs to the drum. This emphasis on speed in core instead of quick access to files delights the user who cares more for conversational response than for chaining portions of large programs.

JOSS can readily monitor its own space and time, but management personnel must decide whether accounting rules will merely recover costs, as at present, or also create incentives to be frugal. Whatever the decision, the scheme should be easy to alter when conditions change—a capability not permitted by the current JOSS logic. Current JOSS accounting avoids a charge for session time and file space, concentrating on core space and compute time, those resources most in demand. Few users would resent a fair charge for file space, but most would probably object to automatic editing of files, which would discard items that had not been referred to in some specified length of time. Most restrictive of the informal atmosphere would be a bill for session time, since the ability to relax at the console contributes to a thoughtful JOSS conversation.

In any specific computing context, there is an appropriate combination of controls to distribute resources fairly while still supporting the goals of the system. The

principal JOSS intent is to maintain a personalized, open-shop approach to computing power. JOSS resources can probably be managed more formally with little complaint, as long as there is never interference with the essential friendliness of console and language.

In an ideal maintenance world, hardware vendors should be kept to a minimum, major system components should be redundantly designed, and the computer itself should be more common in the marketplace. Yet experience has shown again and again that it is the personal services, not the perfection in maintenance procedures, that encourage use of the system.

Who is allowed access to a computing system, and for what purpose, are dangerous topics to raise when the system is open-shop, free, and friendly. JOSS has been available at Rand to anyone who could find a terminal and follow the log-on protocol. Only the month-end accounting summary has informed an administrator that strange initials have credited JOSS computing to his department. Although session records show the hour and channel associated with a set of initials, only the user knows if he has been calculating correlation coefficients or trying to beat JOSS at blackjack.

The system monitor has counted as many references to library demonstration programs as to more serious routines. If such items are overused, they can easily be removed; the average JOSS user is not likely to reproduce their sophisticated logic in his personal file. The freedom to play computer games may or may not be cost-effective, even if considered a fringe benefit of a research activity. Yet it is apparent that game-playing can lead to a rapport with the computer that is a goal of JOSS-like systems, designed for the problem solver who hesitates to tangle with a machine.

Evidence of the influence of the JOSS philosophy on modern conversational time-sharing appears in the wide use of the term "JOSS-like" to describe interaction with a computer by a novice. Evidence of its influence on future users of time-sharing systems can be noted in comments by young visitors to Rand. A junior high school class that visited Rand in 1970 was asked to record impressions of everything seen and heard. Some responded with a list of word associations:

COMPUTER = terminal, box, complex, programmer, complicated, machine, confusion.

JOSS = friendly, fun, friend, kind.

Others commented on their introduction to JOSS:

- It seemed as though we were working with a human being. He was almost like a grammar teacher the way he corrected us every time we made one little mistake.
- JOSS seemed like a real person with a sense of humor.
- JOSS was acting as if we had known each other for years.
- The JOSS computer was "out-a-site." It isn't just a tool one would work with, it's more like a friend who helps you with your work. He's also polite.
- I never knew that a computer could seem so friendly.

If the JOSS years offer a lesson for designers of future systems, it may be this. The power of the modern computer lies not only in its speed, size, and accuracy. Its strength is also in its accessibility and ease of use. The novice and specialist alike will be attracted by a friendly computer.

Appendix A

JOSS LANGUAGE SUMMARIES

JOSS I One-Page Summary

DIRECT or INDIRECT

```
Set x=a.

Do step 1.1.
Do step 1.1 for x = a, b, c(d)e.
Do part 1.
Do part 1 for x = a(b)c(d)e, f, g.

Type a,b,c,_.
Type a,b in form 2.
Type "ABCDE".
Type step 1.1.
Type part 1.
Type form 2.
Type all steps.
Type all parts.
Type all forms.
Type all values.
Type all.
Type size.
Type time.
Type users.

Delete x,y.
Delete all values.

Line.

Page.
```

INDIRECT (only):

```
1.1 To step 3.1.
1.1 To part 3.

1.1 Done.

1.1 Stop.

1.1 Demand x.
```

DIRECT (only):

```
Cancel.

Delete step 1.1.
Delete part 1.
Delete form 2.
Delete all steps.
Delete all parts.
Delete all forms.
Delete all.

Go.

Form 2:
dist. = .....  accel. = __.___

x=a
```

RELATIONS:

= ≠ ≤ ≥ < >

OPERATIONS:

+ - · / * () [] ||

CONDITIONS:

if a<b<c and d=e or f≠g

FUNCTIONS:

sqrt(a)	(square root)
log(a)	(natural logarithm)
exp(a)	
sin(a)	
cos(a)	
arg(a,b)	(argument of point [a,b])
ip(a)	(integer part)
fp(a)	(fraction part)
dp(a)	(digit part)
xp(a)	(exponent part)
sgn(a)	(sign)
max(a,b)	
min(a,b,c)	

PUNCTUATION and SPECIAL CHARACTERS:

. , ; : ' " _ # $?

___.____ indicates a field for a number in a form.
..... indicates scientific notation in a form.
# is the strike-out symbol.
$ carries the value of the current line number.
* at the beginning or end kills an instruction line.
Brackets may be used above in place of parentheses.
Indexed letters (e.g., v(a), w[a,b]) may be used above in place of x, y.
Arbitrary expressions (e.g. 3·[sin(2·p+3)-q]+r) may be used above in place of a, b, c, ...

JOSS II Commands and Functions

In the following list of commands and functions, these symbols are used:

L = letter	int = expression with integer value
S = subscripted letter	num = expression with numerical value
P = proposition	rng = range of values of a letter,
F = formula	such as *1(.1)3,3.64,4*

In JOSS, an expression, e.g., *3+sqrt(2)*, has a numerical value. A proposition, e.g., *a≤b*, has a truth value (true or false). An *if* clause can be appended to any command except a short *Set* command. A conditional expression may be used wherever an expression is allowed; e.g., define f(x) by a conditional expression: *Let f(x)=(x<0:0;0≤x<1:x*2;1)* defines f(x) to be 0 for x<0, x*2 for 0≤x<1, and otherwise 1.

JOSS II Command List

TYPE COMMANDS
Type num.
Type S.
Type S(int,int).
Type P.
Type "any text".
Type _.
Type form int.
Type step num.
Type part int.
Type formula F.
Type F(num).
Type F(P).
Type all steps.
Type all parts.
Type all formulas.
Type all forms.
Type all values.
Type all.
Type time.
Type timer.
Type size.
Type users.
Type item-list.

- Several individual *Type* commands may be combined in one, except for *Type "any text"* and *Type item-list*.
- The words *in form* may be appended to *Type* commands that specify individual values.

DELETE COMMANDS
Delete L.
Delete S.
Delete S(int,int).
Delete form int.
Delete step num.
Delete part int.
Delete formula F.
Delete all steps.
Delete all parts.
Delete all formulas.
Delete all forms.
Delete all values.
Delete all.

- Several individual *Delete* commands may be combined in one, such as *Delete* L, *form* int, *all parts*.

SINGLE-WORD COMMANDS
Page.
Line.
Go.
Stop.
Done.
Quit.
Cancel.

SET COMMANDS
Set L=num.
Set L=P.
Set S(int,int)=num.
Set S(int,int)=P.

SHORT SET COMMANDS
L=num
L=P
S(int,int)=num
S(int,int)=P

LET COMMANDS
Let F=num.
Let F=P.
Let F(L)=num.
Let F(L)=P.

DO COMMANDS
Do step num.
Do step num, int times.
Do step num for L=rng.
Do part int.
Do part int, int times.
Do part int for L=rng.

TO COMMANDS
To step num.
To part int.

PARENTHETICAL COMMANDS
Cancel or any *Do* command may be enclosed in parentheses.

SPECIAL COMMANDS
Reset timer.
Let S be sparse.

DEMAND COMMANDS
Demand L.
Demand S(int,int).
Demand L as "any text".
Demand S(int,int) as "any text".

FILE COMMANDS
Use file int (ident).
Use library int.
File ... as item int (code).
Recall item int (code).
Discard item int (code).

- The file number and identification are assigned by the Information Sciences Department.
- Library files are numbered from 1 to 250 and do not have an identification.
- The item number and code are assigned by the user at time of filing. $1 \le int \le 25$. The code, if used, contains at most 5 letters and numbers.
- Any combination of elements that can be deleted by a *Delete* command may be filed by a *File* command.

JOSS II Function List

```
                    ip(x) = integer portion of x
                    fp(x) = fraction portion of x
                    dp(x) = digit portion of x
                    xp(x) = exponent portion of x
                   sgn(x) = -1 if x<0, 0 if x=0, +1 if x>0

                  sqrt(x) = square root of x for x≥0
                   log(x) = natural logarithm of x for x>0
                   exp(x) = e^x

                   sin(x) = sine of x, x in radians, |x|<100
                   cos(x) = cosine of x, x in radians, |x|<100
                 arg(x,y) = angle in radians formed by the positive
                            x-axis and the line from the origin to (x,y)

  sum(x,y,z) or sum[i=rng:num] = sum of values in argument list
prod(x,y,z) or prod[i=rng:num] = product of values in argument list

  min(x,y,x) or min[i=rng:num] = minimum of values in argument list
  max(x,y,z) or max[i=rng:num] = maximum of values in argument list

  conj(p,q,r) or conj[i=rng:P] = true if and only if P is true for all i
  disj(p,q,r) or disj[i=rng:P] = true if any P is true

              first[i=rng:P] = first i for which P is true

                      tv(P) = the truth value of P; 0 if P is false,
                              1 if P is true
```

- num or P is evaluated for each value of i in the range rng; i usually appears in num or P.

JOSS II Software Changes

New Use of the Asterisk. Originally, the asterisk had two interpretations in JOSS. It could mean (a) exponentiation, as *2*3* for *2 raised to the third power;* or (b) it could begin or end a line of typing, to instruct JOSS to ignore the line. The latter use was extended to permit annotation of stored programs. Thus a step that begins or ends with an asterisk, or is entirely blank, will be ignored during execution, but will appear when the step is typed.

Time in Page Heading. The colon was removed from time as printed in the page heading, to match the result of the command *Type time.* This latter result of time printed without the colon is required when typing time in a form.

Library Files. The command *Use library n,* where *n* is an integer from 1 to 250, was added to access the new read-only library files. *Recall* and *Type item-list* perform as for nonlibrary files. *File* and *Discard* produce error messages.

Interconsole Communication. Commands to permit game-playing among two or more consoles were added on an experimental basis only.

Accounting Changes. The log-on protocol was rearranged in the order of initials, department, project number. A user's department may now be identified from any terminal, whether in Santa Monica or at a JOSS site. In addition, the accounting algorithm no longer charges for storage of file items.

Item-List Format. The original format of a file's item-list included the following five columns:

ITEM CODE RPN DATE SPACE

Two columns were added to indicate the date last used, and the times used since first filed. The new item-list appears like this:

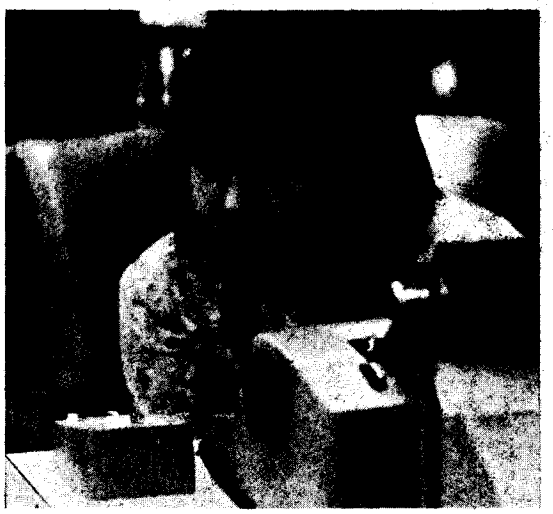
ITM CODE RPN SPACE FILED USED TIMES

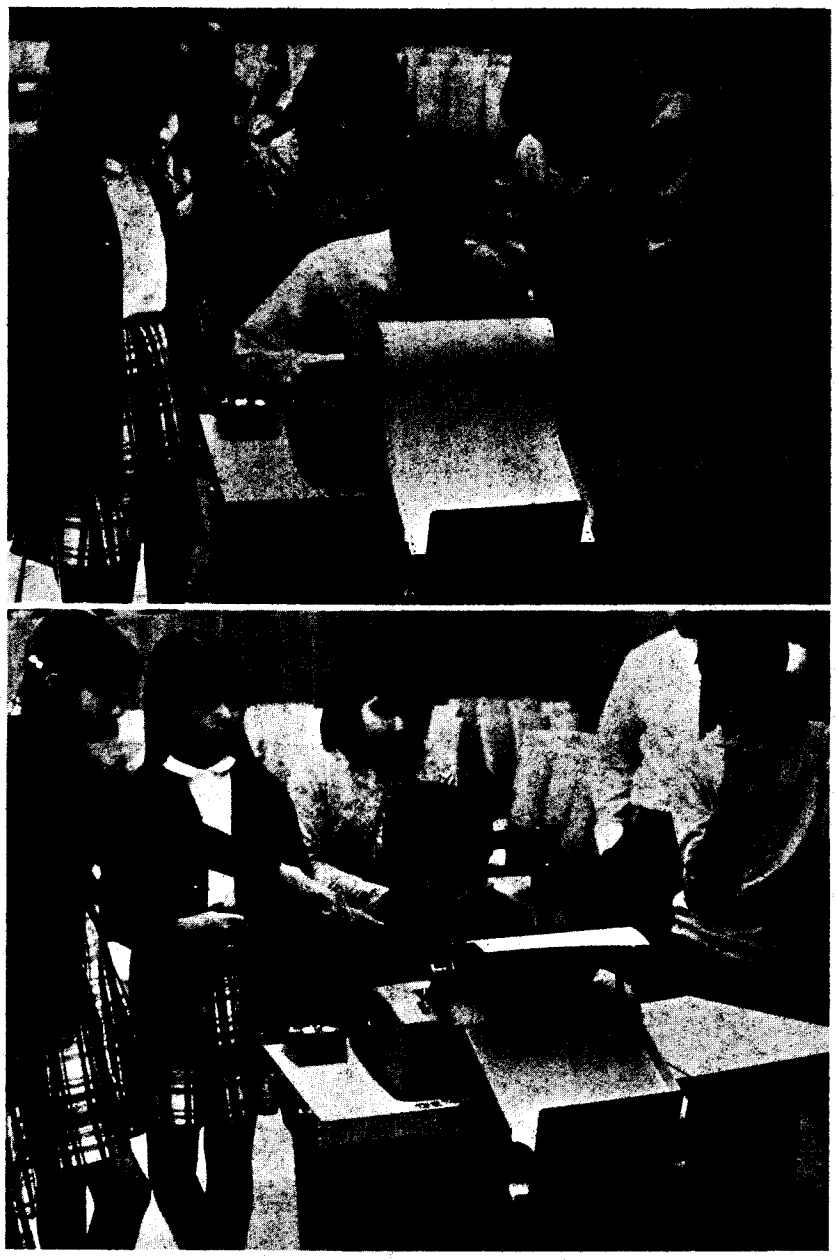
Appendix B

A GALLERY OF KIDS

Among the most responsive of the delightful procession of young visitors to JOSS were sixth-grade students enrolled in a class in remedial arithmetic. These youngsters had been prepared by their teacher in class for their coming conversation with a machine. They had talked about flowcharts, those pictures of a problem that generally precede the computer description. They had even drawn a logical portrait of such familiar activities as getting ready for school in the morning.

Once they overcame their initial shyness in the unfamiliar Rand setting, the children easily ignored the camera in favor of this new friend who said funny things like *Eh?* when he didn't understand. From *Type 2 + 2,* they soon progressed to a simple two-step stored program, and thence to the game of hangman. The following photographs indicate something of their pleasant morning with JOSS.





REFERENCES

1. Marks, S. L., *The JOSS Newsletter, November 1967-June 1971,* The Rand Corporation, P-3940/7, August 1971.
2. Shaw, J. C., *JOSS: A Designer's View of an Experimental On-Line Computing System,* The Rand Corporation, P-2922, August 1964.
3. Shaw, J. C., *JOSS: Examples of the Use of an Experimental On-Line Computing Service,* The Rand Corporation, P-3131, April 1965.
4. Shaw, J. C., *JOSS: Conversations with the JOHNNIAC Open-Shop System,* The Rand Corporation, P-3146, May 1965.
5. Shaw, J. C., *JOSS: Experience with an Experimental Computing Service for Users at Remote Typewriter Consoles,* The Rand Corporation, P-3149, May 1965.
6. Greenwald, I. D., *JOSS: Arithmetic and Function Evaluation Routines,* The Rand Corporation, RM-5028-PR, September 1966.
7. Greenwald, I. D., *JOSS: Console Service Routines (The Distributor),* The Rand Corporation, RM-5044-PR, September 1966.
8. Bryan, G. E., *JOSS: User Scheduling and Resource Allocation,* The Rand Corporation, RM-5216-PR, January 1967.
9. Greenwald, I. D., *JOSS: Disc File System,* The Rand Corporation, RM-5257-PR, February 1967.
10. Baker, C. L., *JOSS: Console Design,* The Rand Corporation, RM-5218-PR, February 1967.
11. Baker, C. L., *JOSS: Rubrics,* The Rand Corporation, P-3560, March 1967.
12. Bryan, G. E., *JOSS: Accounting and Performance Measurement,* The Rand Corporation, RM-5217-PR, June 1967.
13. Smith, J. W., *JOSS: Central Processing Routines,* The Rand Corporation, RM-5270-PR, August 1967.
14. Bryan, G. E., *JOSS: 20,000 Hours at the Console—A Statistical Summary,* The Rand Corporation, RM-5359-PR, August 1967.
15. Bryan, G. E., *JOSS: Assembly Listing of the Supervisor,* The Rand Corporation, RM-5437-PR, August 1967.
16. Baker, C. L., *JOSS: Introduction to a Helpful Assistant,* The Rand Corporation, RM-5058-PR, July 1966.

17. Bryan, G. E., *JOSS: Introduction to the System Implementation,* The Rand Corporation, P-3486, November 1966.
18. Gimble, E. P., *JOSS: Problem Solving for Engineers,* The Rand Corporation, RM-5322-PR, May 1967.
19. Marks, S. L., and G. W. Armerding, *The JOSS Primer,* The Rand Corporation, RM-5220-PR, August 1967.
20. Bryan, G. E., and E. W. Paxson, *The JOSS Notebook,* The Rand Corporation, RM-5367-PR, August 1967.
21. Bryan, G. E., and J. W. Smith, *JOSS Language,* The Rand Corporation, RM-5377-PR, August 1967.