7 November 2013

# Messaging in the Early SDC Time-Sharing System

David Hemmendinger
Department of Computer Science
Union College
Schenectady, NY 12308
hemmendd@union.edu

## Introduction

Electronic communication, as email, file exchange, and various forms of messaging, is central to contemporary computer use. Nearly all of it now uses the Internet, and email rapidly became popular when it was introduced on the ARPANET around 1971. Communication programs emerged before networking, however, in time-sharing systems in the mid-1960s, as described in histories such as Van Vleck's [10].

Today we are likely to think of electronic communication as a means of information exchange that enables people to collaborate remotely. Such collaboration was one motivation for its early development, but another, equally important in early time-sharing systems, was to provide access to system information for users who were logged-in remotely and to let them communicate with its operators.

One of the early time-sharing systems was at the System Development Corporation [SDC] in Santa Monica, CA. Most of the publications that emerged from this project were about technical aspects of time-sharing, such as scheduling and memory-management, or about programs that were run on it. Its use of messaging has not received much attention. This paper describes its early messaging commands and how they were used as part of system operation and for user instruction and collaboration among programmers. It concludes by reflecting on the significance of this work.

SDC, a spinoff of the Rand Corporation, developed its time-sharing system in 1963; it ran on the AN/FSQ-32 (Q-32) computer built by IBM for the military. According to Baum [1], SDC planned to get the computer in 1960 for a SAGE Super Combat Center project, but when the U.S. Air Force canceled that project along with other air-defense programs, SDC proposed to use it for an ARPA-supported command-and-control research project. The Defense Department agreed, and SDC acquired the Q-32 in mid-1961. In late 1962, J. C. R. Licklider, head of the Information Processing Techniques Office (IPTO) of the US Defense Department Advanced Research Processing Agency (ARPA), in keeping with his interest in interactive computing, proposed building a time-sharing system for it. SDC asked Jules Schwartz, author of the JOVIAL real-time programming language, to head the project. The SDC time-sharing system, sometimes called TSS, began operating in mid-1963, at about the same time as the better-known Compatible Time-Sharing System (CTSS) system at MIT.

Although like other early time-sharing systems, the TSS was a testbed for scheduling and other operating system policies, it was generally less a research system than, for example, the CTSS, and more a tool for ARPA projects [9]. These included simulations of a command post, computer-assisted instruction, online database and text-retrieval systems, and military gaming, as well as providing remote access to multiple users across the country. The system was also intended to provide remote access, eventually to users across the country and even in Europe. As a result, its designers had to deal very early with these users' need for information about the operation of the system and about its current status. Such needs were common among users of time-sharing; as Robert Fano, who worked on CTSS, said in an interview, "The point is, how can you end the technical isolation of the individual who works at the terminal remote from other people?" [2]

**Messaging in the SDC system**

One of the commands implemented in 1963 and described in the first paper on TSS in 1964 was DIAL, which let a user communicate with others on line, either users or operators [8]. It was thus an early text-messaging capability; according to a preliminary user manual for the TSS, DIAL messages were limited to a single line of text (about 88 characters) and could be sent to up to three terminals [4]. In addition to that command, another 1964 paper on the system [7] also describes a LINK command:

> Initiates linkage of any two teletypes so that they act as one. Both teletypes can input and output to and from the same program or the system. Also, both teletypes can type all the information being input or output on each other.

A 1966 paper [3] gives an example of the use of DIAL. Unlike other early time-sharing systems, it says, Q-32 system management did not limit the number of concurrent users or their resource acquisition. A user could schedule a job, but if a requested resource such as drum space were not available at the time to run, the user would notify the operator, who would ask some low-priority users to terminate their jobs. Figure 1 is an abbreviated portion of dialogue between users and operator, from [3, Fig. 3]. It shows a user's request for more storage and the operator's request to another user to free it up.

```
LOGIN 1033S JG025
$OK LOG ON 16
DRUMS
$4324 DRUM WDS.
DIAL 9  THIS IS JOHN JONES, I NEED 20K IN ORDER TO LOAD MY PROG,
FROM 9  WE CAN GET YOU ON IN 5 MINUTES.
FROM 9  GO AHEAD AND LOAD,
LOAD ALPHA1

(CHANNEL 9'S TELETYPE COPY - THE OPERATOR'S TELETYPE)
FROM 16  THIS IS JOHN JONES, I NEED 20K IN ORDER TO LOAD MY PROG.
26  (REQUESTS MANAGER PROG. TO GIVE A DETAILED DISPLAY OF CH. 26)
!DIAL 26 CAN YOU FINISH IN 5 MIN. - WE HAVE A HIGH PRIORITY REQUEST,
FROM 26  OK - I'LL FINISH UP IN 5 MINUTES.
DIAL 16  WE CAN GET YOU ON IN 5 MINUTES,
FROM 26  I'M THROUGH
DIAL 16  GO AHEAD AND LOAD.
```

Figure 1: An example of DIALing between user and operator.

I have found no published examples of messaging between Q-32 users, and communication with the operator may have been a major use of DIAL. Larry Press, who used the Q-32 system in the mid-1960s, recently wrote of the command, "I don't recall using it much except to fool around. The Research Directorate was small and we all worked at the same place and used the same terminal room so online collaboration was not so important."[1] When the Q-32 time-sharing system later became a computing resource to universities and military sites across the country, however, its messaging commands became important to remote users.

Art Rosenberg, who worked on the TSS in the 1960s, provided some information about LINK in a recent blog [5]. In early 1964, shortly before the April conference at which [8] was to be presented, Rosenberg asked the programmer who was writing the remote-user interface to set

[1] Email, Jan. 21, 2013. Ed Coffman, who worked on the TSS, has similar recollections (email, Feb. 7, 2013).

it up to allow two connections to the same program. That LINK program was finished a week before the conference, and allowed the two connected users to type messages to one another as well as to share program input and output. Rosenberg set up the TSS demonstration at the conference, having arranged with researchers who were using the system to be available remotely (Figure 2). During the meeting, he linked to the online researchers in several parts of the country, as he described in his blog:

> Needless to say, computer show attendees who were used to batch-processing, premise-based main frames, could not believe what they saw from the Model 33 ASR terminals [running at 110 bits/second] connected to standard phone lines that I was using. The computer system itself was three thousand miles away and they could interact in real-time with different applications and concurrently exchange text messages with the people who were also three thousand miles away.

In his blog and in telephone conversations[2] Rosenberg pointed out that such messaging capabilities, along with later electronic mail, contributed to a shift in emphasis from time-sharing as a way to give multiple users access to interactive programs, perhaps remotely, to time-sharing as a means of communication and collaboration. The SDC system was used for a small number of research projects and by relatively few people, which may help to account for its early text-messaging not becoming more widely known. According to Rosenberg, however, several of these projects did involve collaborative work, and in addition to the operating-system messaging functions, the application programs themselves provided communication capabilities.

Figure 2: Demonstrating remote access to the SDC TSS at the 1964 Spring Joint Computer Conference, including the use of the LINK command. Clayton Fox is at the teletype. — from *SDC Magazine*, **7** (May, 1964), courtesy of Art Rosenberg.

Remote terminals for a time-sharing system posed new difficulties for computer users, who could not easily consult with an operator in case of trouble. Clark Weissman, who was in charge of a LISP project on the TSS, describes dealing with some of them in a 1965 paper given at an IEEE human-factors meeting, under the heading, "party lines" [12]. DIAL could send a brief

[2] August 6 and October 1, 2012.

message to let someone find and talk telegraphically with an expert, to schedule meetings or send announcements. Its most important use, he says, was as illustrated in Figure 1 — to send instructions to the computer operators and to get messages from them. (Interestingly, while the DIAL command was part of the TSS executive, the actual message transfer did not have to use the Q-32, but only a PDP-1 that was the interface between the Q-32 and both local and remote terminals.) As Rosenberg described, LINK could either connect two terminals to one another to send messages of any length or connect them both to the Q-32 to interact with a program. In addition to remote demonstrations such as Rosenberg ran at the 1964 SJCC, Weissman writes that LINK was used for consultations about remote terminal problems, for remote instruction, and for group work such as joint debugging of a program written by a group of programmers — capabilities that we generally associate with what the Internet and World Wide Web have made possible.

The paper describes a third command, JOIN. It was invoked by a program rather than by a user, and could couple multiple terminals to the program, which had to be designed to handle these multiple connections. According to the paper, it was used extensively by SDC programs for command-post simulation, war games, and experiments in decision-making, though the paper doesn't further document such uses.

In a paper on group communications [6], Rosenberg elaborates on uses of JOIN and LINK. The former was designed so that only idle terminals could be joined to a program, so as not to interfere with a terminal already engaged in running another program. While a joined terminal could quit the connection, it was not allowed to issue system commands that would interfere with the joining program. On the other hand, in case a terminal didn't disconnect when it should have, there was also an UNJOIN command that only the originating terminal could issue.

While JOIN was asymmetric, with one terminal initiating a program that then connected other terminals to itself but restricted what those terminals might do, LINK was symmetric once two terminals were connected. As a result, linking could interfere with output on a linked-to terminal (which were generally teletypes with printed output that could not simply be refreshed to eliminate clutter). Either of two linked terminals could issue any system commands, including one to terminate the shared program in the midst of a run. According to this paper, both a command to block linking and one that would make the linked-to terminal passive and unable to send program input were available but when issued, they would go to the system operator for approval rather than take effect directly.

LINK also provided an unexpected capability: a user could link to an unused terminal, start a program, and then unlink, leaving that previously unused terminal connected to the running program. The user could relink to that terminal periodically to check on the program, and could repeat this operation with other terminals. According to Weissman, this let a programmer develop a JOVIAL program interactively with TINT, a JOVIAL interpreter, then use another terminal to compile and run the program while continuing to work interactively. In effect, LINK provided the equivalent of running multiple jobs in the background, or of the ability today to run multiple programs in different windows — yet another demonstration of finding unintended uses for system commands.

Finally, Weissman and Rosenberg describe some of the mistakes that we still encounter with such remote messaging. One remote user asked an operator to "decipher the above error message", not realizing that it did not appear on the operator's terminal. Since DIAL messages were addressed only by easily-mistyped terminal numbers and couldn't be blocked, during one demonstration to high-ranking officials, someone sent a report of World Series game scores to its terminal. A blocking command was added later to prevent such blunders.

**Conclusion**

The papers by Rosenberg and Weissman are about collaborative work and about the needs of remote users. SDC had several projects on group decision-making and on computer-aided group work, and JOIN and LINK let people share terminal I/O and interact with one another. Weissman discusses remote users' need to have access to system information, both about how to use the computer and about its current status, needs partially met by using DIAL to communicate with the system operator. He adds that an online user guide would help to solve the former problem, though it had not yet been set up. The SDC messaging functions thus represent early experiments with time-sharing, exploring how the system could facilitate interactions among its users and what it needed to become more easily used remotely. As systems and programs matured, some of the former capabilities were provided by the programs written for group interaction rather than by an operating system, and some of the latter capabilities were provided by online manuals and 'help' utilities. Exchange of information among users was done by file-transfer and electronic mail, neither of which interfered with the flow of information on a terminal. Although it may be hard to discern direct descendants of the TSS messaging commands, we might note that some of its programmers and users continued to work on computer tools for effective interaction.[3]

Evidently text-messaging on the SDC system was well-established two years before the CTSS at MIT acquired a similar command along with electronic mail, which Van Vleck described recently [11]. According to that article, when Noel Morris and he implemented their WRITE command along with MAIL in 1965, they were not aware of other systems with such commands. Given that the SDC system DIAL and LINK commands were mentioned only briefly in two 1964 publications, it is not surprising that they would not have been aware of them then. Van Vleck's similar online history of email [10] says that he has been told that the SDC system had email and messaging in 1965. There is no indication that it did have email even by 1967. His online history page also says that he has not found any such commands documented before the 1965 MIT work. As this article shows, though, there were several published accounts of the SDC messaging commands in the 1960s, including [3, 7, 8, 9].

We may ask why the TSS didn't have electronic mail. In its initial design, it had no disk drive and hence no long-term user file-system, though according to [8], it did acquire a disk in 1964, not long after it started operation. The 1967 review of the TSS mentioned that disk space was a problem, and the operating system deleted the oldest files when the disk became more than 90% full. Unless email files were exempted from such scavenging, the practice would make it difficult to maintain an email system, and is probably sufficient reason for the TSS not to have had email.

As several writers have remarked, the invention of email and of messaging didn't require any major new steps; they emerged naturally in time-sharing systems, and the emergence of email put messaging in the background. The papers on the TSS emphasized other aspects of the system, and little appears to have been published on experiences of system users, apart from Weissman's technical report and conference talk [12]. Unlike the CTSS, the SDC TSS was used primarily as a tool for developing applications for the military, which may have limited the extent to which they could be published. Published work about the system was largely limited to research on details of time-sharing, such as job scheduling. It is surprising nonetheless that its early examples of messaging, as explorations in the effective use of time-sharing, did not get documented in histories of the topic, or as far as I have been able to determine, in histories of time-sharing.

---

[3] See, for example, Rosenberg's *Unified Communications Strategies* web site, `www.ucstrategies.com`.

**Acknowledgments**

I thank Ed Coffman, Larry Press, Art Rosenberg, and Clark Weissman for discussions of messaging on the TSS, and for SDC documents. Beth Hoppe of the Union College Library, and the staff of the Charles Babbage Institute helped to acquire two fifty-year old SDC technical reports.

**References**

1. Claude Baum, *The System Builders: The Story of SDC,* System Development Corporation, Santa Monica, CA (1981).
2. Robert M. Fano, "Oral history interview with Robert M. Fano," CBI OH 165, Charles Babbage Institute (April, 1989). `http://purl.umn.edu/107281`. Conducted by Arthur L. Norberg.
3. Richard L. Linde and Paul E. Chaney, "Operational management of time-sharing systems" in *ACM '66: Proceedings of the 1966 21st National Conference,* pp. 149-159, ACM, New York, NY (1966). DOI 10.1145/800256.810691.
4. Arthur. M. Rosenberg, "Interim operational design and user's guide for model 1 time-sharing system (TSS Model 1)," TM-1126, System Development Corporation (March 25, 1963). In the Charles Babbage Institute's Burroughs Collection, box 15, folder 23.
5. Arthur M. Rosenberg, "Back-to-the-future — real-time collaboration through applications" in *Unified Communications Strategies Blog* (October, 2011). `http://blog.ucstrategies.com/index.php/tag/collaboration`.
6. Arthur. M. Rosenberg, "Group communication in on-line systems" in *Proceedings of the Symposium on On-Line Computing Systems,* ed. Eric Burgess, pp. 69-82, American Data Processing, Inc, Detroit, MI (1965).
7. Jules I. Schwartz, "The SDC time-sharing system, part 1," *Datamation* **10**(11), pp. 28-31 (November, 1964).
8. Jules I. Schwartz, Edward G. Coffman, and Clark Weissman, "A general-purpose time-sharing system" in *Proceedings of the April 21-23, 1964, Spring Joint Computer Conference,* pp. 397-411, Association of Computing Machinery (1964). DOI 10.1145/1464122.1464163.
9. Jules I. Schwartz and Clark Weissman, "The SDC time-sharing system revisited" in *Proceedings of the 1967 22nd National Conference,* pp. 263-271, Association of Computing Machinery (1967). DOI 10.1145/800196.805996.
10. Tom Van Vleck, *The history of electronic mail* (2012). `http://www.multicians.org/thvv/mail-history.html`.
11. Tom Van Vleck, "Electronic mail and text messaging in CTSS, 1965-1973," *Annals of the History of Computing* **34**(1), pp. 4-6, IEEE (2012). DOI 10.1109/MAHC.2012.6.
12. Clark Weissman, "Communication techniques for an international time-sharing users group," SP-2029, System Development Corporation (April, 1965). Presented at the IEEE Sixth Annual Symposium on Human Factors in Electronics, Boston, MA, May 6-8, 1965.