WIS-CS-187-73
The University of Wisconsin
Computer Sciences Department
1210 West Dayton Street
Madison, Wisconsin 53706

Received July 9, 1973

LOOSELY COUPLED SYSTEMS

by

E. J. Desautels, V. Chow, M. Schneider

Computer Sciences Technical Report #187

July 1973

LOOSELY COUPLED SYSTEMS

by

E. J. Desautels, V. Chow, M. Schneider

ABSTRACT

The National Science Foundation is supporting an investigation into the costs and benefits of coupling a small stand-alone time-sharing system (such as TSS-8 or RSTS-11) to a large multiprogrammed system (such as OS/370 or 1108 EXEC8). A system is being designed which will support local limited resource time-sharing, remote batch operations, on-line remote job entry, and line concentration, as well as allow the small system to act as an "intelligent" terminal. The system will allow the migration of programs and data between the two systems as well as allow the user of the small system to access all physical resources of the larger one. The system will be initally implemented on a Datacraft 6024/1108 pairing but a primary goal of the design is to generalize to other machine pairs. These loosely coupled systems will increase the practical modes of access to computing services and hardware resources.

This work was supported by the National Science Foundation under Grant GJ-36078, and by the University of Wisconsin Graduate School.

INTRODUCTION

A proposal entitled "Loosely-Coupled Time-Sharing Computer Systems" was submitted to the National Science Foundation in December, 1971 (Desautels). The proposal was accepted, and this present report is an elaboration of the original proposal, and will be the first of a set of working papers describing the goals and the progress being made.

The phrase "loosely-coupled time-sharing systems" is used to refer to a pair of independent computing systems, which are able to communicate with each other, while at the same time retaining their capability for independent operation. The coupling of the two systems is intended to enhance the utility of each system, particularly if they are very dissimilar. The dissimilarities range from sheer physical size (mini vs. maxi) to diversity of languages processors and other services.

HISTORICAL CONTEXT

In the period 1965-1969, the senior author designed, implemented and ran in production mode a time-sharing system based on a machine (7094) ill-suited for the purpose. This research was conducted under NSF auspices, for regional computing networks (Desautels, Ph.D. Thesis). Since that time it has been noted that many large-scale computing facilities provide on-line capabilities only as an after thought; the time-sharing has been retrofitted onto a multiprogrammed batch system. This shows up dramatically in the command language structure and scope of some large systems. There have been a few exceptions, either because of large-scale time-sharing research emphasis such as MTS and Multics, or due to a dedication to providing good on-line access within a multiprogrammed batch environment, as exemplified by the PROCSY system at Purdue [PROCSY].

In the meantime, with the advent of mini-computers, several hundred small-scale, limited-resource, time-sharing systems have been deployed.* The most widely used are 8 or 16 user TSS-8 DEC systems, and 16 or 32 user Hewlett-Packard 2000 systems. DEC's more recent RSTS system based on the PDP-11 is coming into widespread use. These small systems (usually between $75,000 and $150,000 purchased) provide stable, convenient computing for many classes of applications. To date, such systems have not been exploited by coupling them to larger systems.

It is our contention that much of the work currently done on large systems (measured in number of jobs) could be done on limited-resource systems, at lower cost. Furthermore, a

*At such diverse locations as Ripon College, Wisconsin, Stanford Graduate School of Business, Physics Dept. - Cambridge University, etc.

limited-resource system, once coupled to a larger system, becomes attractive to the large-system user, for convenience in data acquisition, program and data preparation and remote job entry.

There are a number of advantages limited-resource systems have over large-scale time-sharing systems. Among these are:

1) availability - the small system is likely to be available 24 hours/day for weeks on end. Almost every large system is shut down daily for one or more hours of preventive maintenance and systems test.

2) stability - the small system is likely to run for a week or more without crashing, whereas most large systems have a MTBF of 12 to 24 hours, even after several years of development work.

3) maintainability - the software maintenance of the small system is orders of magnitude simpler than that of the large systems.

4) flexibility - the small system can more easily be adapted to handle new devices, respecting the full capabilities of the new devices (as opposed to forcing all devices to behave like a KSR 33).

5) growth - smaller systems can grow in much smaller increments; however they can reach their maximum size rapidly, and expansion takes place through replication.

This last factor - the ceiling on growth can effectively be alleviated, we contend, by coupling the limited-resource system to a larger one, having an almost unlimited capacity or growth potential vis-a-vis its smaller brethren.

The factors which contribute to the stability of the smaller systems, due to their simplicity, also limit their applicability. However, we contend that coupling them to larger systems makes all of the large system software available to the small-system user, if only via remote job entry on-line. Such a coupling can be supported by relatively low speed lines (under 10K baud). We would also wish to explore the problems associated with higher speed lines, in terms of providing highly interactive time-sharing on a large system via a transparent small system. That is, how effectively can a mini-computer simultaneously function as both a time-sharing system and a data concentrator?

Cost may or may not be an advantage of a small system versus a larger one, when viewed on a per-run basis. In terms of purchasing a facility, the startup cost and incremental costs for small systems may be an order of magnitude lower than that of larger systems.

CURRENT SYSTEMS

We will classify computer systems in three categories:

1) small (mini) computers
2) medium-large multi-programmed systems
3) computer networks

The boundary between category 1 and category 2 keeps shifting; some overlap between them is also evident. On the whole, it seems that most small computers are used in a single-user environment, often for a dedicated application. However, as indicated earlier, mini-computer based time-sharing systems have come to number in the hundreds in recent years. The highlights of a typical limited-resource time-sharing system (RSTS) are given in an appendix.

The larger systems are typically used in one or several of the following modes, in a multi-user environment:

1) batch processing, using on-line peripherals
2) remote batch, using remotely located peripherals
3) on-line (interactive, demand, time-sharing, conversational, etc.) with low-medium speed peripherals located remotely
4) on-line data entry

The remote batch (RJE, Remote Job Entry in IBM parlance) should be distinguished from a limited form of on-line access sometimes called on-line data entry (or CRJE for Conversational RJE). On-line data entry allows a user to prepare a batch job with the aid of an interactive editor, submit it, and retrieve its output at his own terminal; such a system is in all other respects a batch processor.

Computer networks can provide the widest range of services. The up-to-date survey by Schneider (S1) should be consulted for information on current networks.

Present use of minis connected to large systems.

Even though the stand-alone use of minis continues, the use of communication equipment for connecting minis to large systems is increasing.

The functions they perform on behalf of the large systems are:

1) control of high-speed peripherals, as in a remote batch station with a card reader and a line printer (e.g. PDP-8's in the OCTOPUS Network, Fletcher)
2) data concentration: handling of several low speed lines, acting as a front-end to the large system
3) communications control: store-forward and communications protocol handling.

PROPOSAL

It is suggested that many of the above services can be accommodated via a single small computer, coupled to a large system. We hope to demonstrate that many of these functions can be effectively and economically combined on one system. The system design should be sufficiently modular to enable a director of computing services to begin with any one of these services, and evolve into combinations of these services.

The proposed system must provide:

1) remote batch access to a larger system
2) on-line data entry (CRJE)
3) line concentration (access to remote on-line service)
4) local limited-resource time-sharing

In a system which can provide all of the above services, it becomes attractive to introduce new facilities which exploit the interaction of these services. These enhancements will be described.

It is of course understood that should any one use of the system reach a point where it dominates and excessively interferes with other uses, then it may be cost effective to separate that use and serve it on a separate single-use dedicated mini.

Thus in the limit, starting with multiple services provided on a single mini, we could evolve into a collection of minis, each providing one of the services, should the service loads require it.

TECHNICAL AND HUMAN ENGINEERING ISSUES

We have observed that most systems for large computer have been designed for programmers rather than for the ultimate end-user or consumer of information processing services. We have also observed that most systems for large computer are effectively cast in concrete; that is, either they can only be changed with extreme difficulty, or if they can be changed relatively easily, one runs into a significant amount of red tape in getting any change performed rapidly.

On the other hand, a system on a small computer is usually smaller, more quickly changed, and small-machine systems thus can economically be tailored for the end-user.

Assuming one has access to a large multiprogrammed system with a remote batch capability and with an interractive capability, it should be possible to develop a system on a small computer which provides access to both classes of service, with no changes in the large-system software. If the small system supported n interractive terminals and a card reader, line printer for remote batch, then a high speed synchronous line and as many as n lower-speed asynchronous lines would be required. Of course, these n asynchronous lines could be replaced by a faster line and a communication line multiplexor, but this would require a change in the large system-namely the introduction of a matching multiplexor, and political implications in the inability to contend for access to the large system, relative to other users with dial-up access.

If the large system has line concentration/deconcentration software, then the small system need only have one high speed synchronous connection, with all inter-system traffic sharing this path.

Much of the human engineering concerns endowing the small system with properties that allow the end-user to mold it as he sees fit. Specifically, he must be allowed to specify his own command language; carried to the extreme, the user-molded system could appear to have no command language (e.g., merely by turning on his terminal, the user immediately has the use of a simulated desk calculator).

The mechanisms which can facilitate such a molding involve language definition capabilities. Our intent is not to develop new definitional capabilities, but to apply known techniques at the command language level, as in the PROCSY system.

Among other things, the end-user must have control over the behavior of his own terminal. Thus he must be allowed to specify how he wishes to erase characters or lines, how the end of a line or message is to be indicated, etc. For users who do not wish to worry about such considerations, system provided defaults will apply.

PROBLEM AREAS

There are several problem areas which can be identified in this proposed investigation. Among these are:

a) coherent working environment - how does one facilitate, for the user, the transition from one operating system's command language to the other's. The implementation of a common command language macro facility would greatly alleviate this problem.

b) graceful degradation - how does one simultaneously provide a sufficiently loose coupling between the large and small systems while allowing for intimate interaction without the larger one's crash bringing down the smaller one.

c) file migration - what are workable procedures for automatically managing the files of one system by using another system's mass storage file system, even though file structures, code sets and other conventions are different.

d) system availability - what techniques are appropriate for backing up the small system library without shutting down the system, keeping in mind that unattended operation is the goal.

DESIGN MODULARITY AND GROWTH CAPABILITIES

Depending upon the immediate needs and budget of an installation acquiring its first "in-house" capability, the design approach we will be investigating should allow an installation management to select any one of the following service classes:

1) remote batch
2) local limited-resource time-sharing;
3) conversational remote job entry and interactive access to a large system

Incremental growth as needs indicate and the budget allows can then permit any of the other above services to be included.

We hope to provide experimental results on the cost/ performance aspects of these services in various combinations, showing the extent to which they interfere or complement each other in terms of system performance. We wish to measure the requirements of each in terms of memory space for code and buffers, and indicate which important parameters an installation manager could manipulate.

<u>SCENARIOS</u>

It may be helpful to describe some of the cases we foresee, in terms of user scenarios.

S1: remote batch. A user walks in with a card deck for submission to the large system, via the small system. His output will be produced on the small system printer.

S2: calculator. A user calls up the small system and uses it as a desk calculator.

S3: BASIC. A user calls up the small system, enters and runs a BASIC program.

S4: Canned. A user calls up the small system, and invokes a system routine (or library routine, or pre-stored BASIC program) and interacts with it.

S5: Remote Job Entry. A user calls up the small system, uses its editor to prepare a job which he then submits for batch execution on the large system. He then disconnects. Some time later, he retrieves the output from the small system, at his terminal. He could of course have had the output go directly to the small system high speed printer.

S6: Interactive. The user calls up the small system, and has it establish an interactive connection with the large system. He should be able to direct the large system to access his files on the small system. Here the user can adjust the "transparency" of the small system at will.

S7: Foreign RJE. It should be possible for the small-system users to prepare jobs which can only be run on some other large system, other than the most frequently used large system. At some scheduled time, the installation management can establish a remote-batch connection with alternate (and different) large systems.

Many other scenarios are possible. S6 presents the greatest challenge in that we would want the small system to interpret the messages issued by the large system originally destined for human interpretation, unless of course the user wishes the small system to be completely transparent.

MEASUREMENT AND EVALUATION

The success of the proposed system can only be established in relative terms. The main tools at hand are:

1) synthetic jobs
2) scripts
3) questionnaires
4) hardware monitors

One objective measure can be obtained. The degree of transparency of the small-system when used to access the large-system in interactive mode can be evaluated using a modification of Turing's test for machine intelligence. That is, a user need not be told whether he is directly connected to the large system, or connected via the small one. If in fact he cannot note any difference, or the difference favours the connection via the small system, we will have succeeded.

The questionnaires can be easily automated, facilitating their analysis. The synthetic jobs, scripts and hardware monitors can provide insight into system bottlenecks. They probably will tell us more about the shortcomings of the implementation and the configuration than about the value of the system.

One of the purposes of the small system is to facilitate access to the large system. As was pointed out by one of the proposal's referees, "the coupled system might be better because you can get to it".

DESIGN GENERALITY

We would wish to design a system which does not owe its success to the existence of any particular computer. A successful implementation will be the result of having appropriate configurations for the small and large systems, and having a competent implementation group, sensitive to the needs of the end-users.

A good design might be more portable if it were implemented in a "systems implementation language". We feel at this time that an appropriate use of macros and conventional documentation techniques is sufficient as a design guide for a new implementation.

The use of layering techniques, whereby the innermost "circle" of code is concerned with I/O and interrupts, and subsequent layers are concerned with logical entities, rather than the necessarily machine dependent physical entities, allows most of the machine dependent characteristics of the design to be clearly delimited.

IMPLEMENTATION CONSIDERATIONS

At this time, it is clear that we should use the university's 1108 as the large-system. It supports both remote-batch and interactive access under EXEC8. At this time it has no line concentration/deconcentration capability, but the 1108 administration is committed to developing one.

We have in our Computer Systems Laboratory, a PDP 11/20 with a 16KWX16 memory and a 64KW disc, and a Datacraft 6024/3 with an 8KWX24 memory and a 5MB disc. Both are equipped with synchronous interfaces. The maximum memory size on the PDP 11/20 is 28KW x 16 (56KB), while on the Datacraft the maximum memory size is 64KW x 24 (192KB). Given the large disc capacity of the Datacraft in the presently available systems and the potential for a larger memory, and the availability of options such as memory protection and privileged instruction traps, we have chosen the Datacraft as the base for the small-system. The appendix shows the current list prices for equivalent Datacraft and PDP-11/20 configurations (if one were to begin with no hardware now, the PDP-11/40 and Datacraft 6024/4, each with virtual memories, would be very promising).

We have noted that connecting a number of terminals to a computer can be an expensive proposition, in addition to the cost of the terminals. Thus we have specified our requirements for a terminal multiplexor with a control sophistication equal to any commercially available multiplexor we have seen, and it has been built at a reasonable cost (less than $200 per line, for a 16 line system). This multiplexor will be described in a subsequent report.

We have also noted that users need more convenient or economical storage media than are presently available on most systems. That is, users must be able to save information on a machine readable medium more convenient than paper tape, and we are investigating the advantages of floppy discs versus magnetic tape cassettes and cartridges.

Appendices show a proposed intial configuration for the system, and a tentative memory map.

OBSERVATIONS

Since our project was initiated, we have noted some manufacturers moving in the direction we had proposed. For instance, Digital Equipment Corporation, in its EDU bulletin no 6 (late 1972) describes on page 6 an RJE capability for the PDP-11, which can be run when the system is not being used in its usual interactive mode. It appears that the RJE capability is completely independent, and not able to access existing files.

It is appropriate to conclude with a quotation from the ACM SIGUCC Newsletter of November, 1972 (p. 61):

> "1. The totality of problems of providing computing services is not yet solved for either large or small universities. Probably the problem is that we are providing good batch, RJE service, and a modicum of interactive services but this does not mean that the problems of providing good/excellent computer services are understood completely, let alone solved. It is not an exact science, computing services are not quantifiable, and even the methodology of evaluating alternatives is not yet worked out (developed, defined) in this continuing dynamic environment."

We would hope to shed some light on some of these problems, not necessarily restricted to a university setting.

APPENDIX A

DEC's PDP-11/20 RSTS

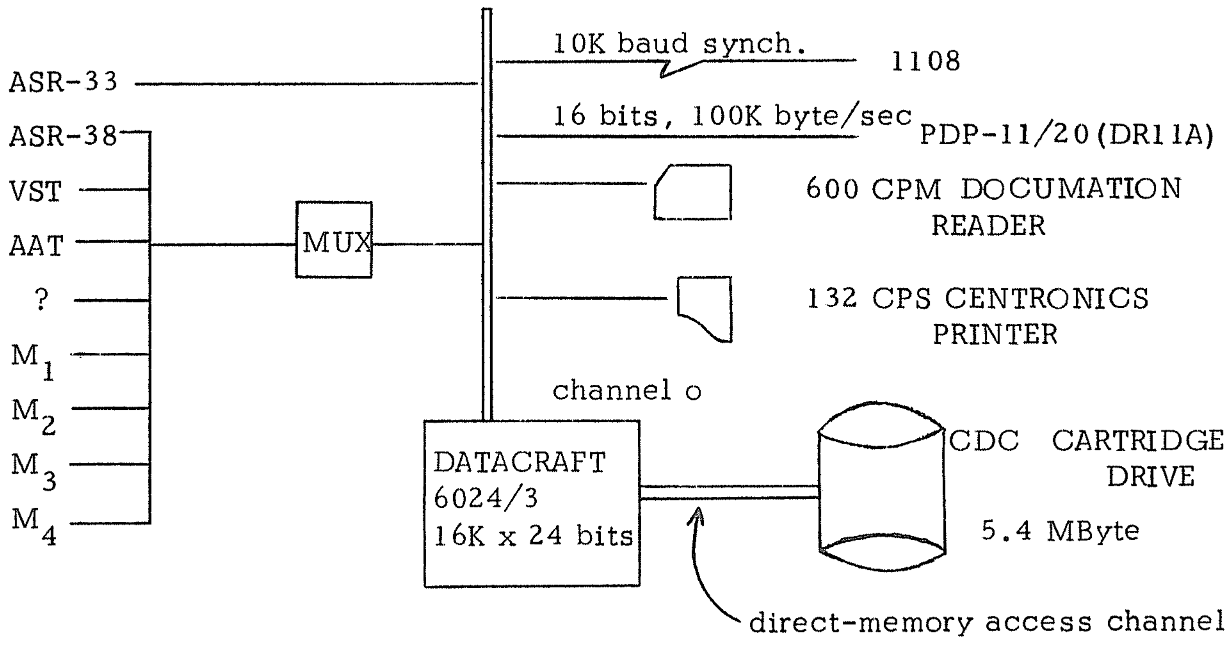
Digital Equipment Corporation has several versions of the Resource Time Sharing System RSTS-11. One of them, MINI-RSTS, uses a 4.8 MB moving head disc, and a PDP 11/20 with 48,000 bytes. An 8 user system costs $45,000 plus terminals. It supports an extended version of BASIC, which can manipulate files, allows program chaining, and automatic paging of large matrices.

A larger configuration for the standard RSTS will support 16 users, allowing a 16K byte core segment for user jobs and use of DEC tape for private libraries.

An extended RSTS, RSTS/E, is supported on the PDP11/40 or PDP11/45, starting at $85,000. It will support a maximum of 32 users, expand to 248K bytes of memory, and support 32K byte programs. Hardware supported virtual memory allow for data arrays of indefinite size.

APPENDIX B

Initial Configuration



VST: Video Systems CRT, baud rates 110, 300, 1200

AAT: Ann Arbor CRT, baud rates 110, 2400

M_1, M_2: answer-only modems (110-300 baud), Universal Data Systems

M_3, M_4: manual originate, auto-answer modems (110-300 baud), UDS

The MUX, designed and built by the UW Physical Sciences Laboratory, can handle 32 lines, with speed selection from 110 to 9600 baud.

A third channel will be used to distribute devices shown above as attached to channel o. A 120 Hz clock is attached.

APPENDIX C

Pricing for the Datacraft Configuration

The prices used were in effect in January, 1973. A Datacraft 6024/5 CPU is priced in place of our 6024/3. They are compatible, run at the same speed, have the same growth potential, but the more recent 6024/5 sells for $13,400 , as opposed to $32,800 for a 6024/3 (both with 8K words)

CPU, 8K	13,400	13,400
8K memory increment	5,500	18,900
DMA channel	1,500	20,400
600 CPM reader	6,000	26,400
Centronics 165 CPS printer	4,000*	30,400
Synchronous interface	2,000	32,400
Disc drive (5.4 Mbytes)	8,100	40,500
Disc controller	5,000	45,500
MUX, with 8 lines	3,000	48,500
Modems, 4 at $200	800	49,300
Processor options	1,500	50,800

*The printer sells for approximately $3,200; the interface which we designed and assembled could be replicated for less than $800.

APPENDIX

Tentative Memory Map

All estimates are in 24 bit Datacraft words; note that the disc sector size is 112 words.

Needs	Size	Running Total
MUX handler	500	500
Handler for disc, reader, printer	1,000	1,500
Remote batch package	2,000	3,500
Command language interpreter	500	4,000
Editor	1,000	5,000
Librarian	1,000	6,000
Overlay area	500	6,500
Buffer handler	500	7,000
BASIC processor	3,000	10,000
BASIC workspace	2,000	12,000
Support for 8 terminals	2,500	14,500
Slack	1,500	16,000

Support for 1 terminal	Size	Running Total
Control block	50	50
Line buffer	40	90
Disc buffer	224	314

Thus for 8 terminals, we need 2512 words, or approximately 2,500.

REFERENCES

E. J. Desautels, A Proposal to the National Science Foundation for a research grant in support of Loosely-Coupled Time-Sharing Computer Systems, Principal investigator, E. J. Desautels, Computer Sciences Dept., UW-Madison, Dec. 71.

E. J. Desautels, The PUFFT Time-Sharing System-Design, Implementation and Performance, Ph.D. Thesis, Purdue University, June 1969.

S. Rosen et al, The Purdue Remote On-Line Computing System PROCSY. Proc. ACM Conference, August 1971.

M. Schneider, A Survey Report on Computer Networks, UW Computer Sciences Dept. Technical Report #177, March, 1973.

J. G. Fletcher, The Octopus Computer Network, Datamation, April 1973.

BIBLIOGRAPHIC DATA SHEET	1. Report No. WIS-CS-187-73	2.	3. Recipient's Accession No.

4. Title and Subtitle Loosely Coupled Systems	5. Report Date July 1973
	6.
7. Author(s) E. J. Desautels, V. Chow, M. Schneider	8. Performing Organization Rept. No. 187
9. Performing Organization Name and Address The University of Wisconsin Computer Sciences Department, 1210 W. Dayton St, Madison, Wisconsin 53706	10. Project/Task/Work Unit No.
	11. Contract/Grant No. NSF GJ-36078
12. Sponsoring Organization Name and Address National Science Foundation Washington, D. C.	13. Type of Report & Period Covered
	14.

15. Supplementary Notes

16. Abstracts The National Science Foundation is supporting an investigation into the costs and benefits of investigation into the costs and benefits of coupling a small stand-alone time-sharing system (such as TSS-8 or RSTS-11) to a large multi-programmed system (such as OS/370 or 1108EXEC8). A system is being designed which will support local limited resource time-sharing, remote batch operations, on-line remote job entry, and line concentration, as well as allow the small system to act as an "intelligent" terminal. The system will allow the migration of programs and data between the two systems as well as allow the user of the small system to access all physical resources of the larger one. The system will be initially implemented on a Datacraft 6024/1108 pairing but a primary goal of the design is to generalize to other machine pairs. These loosely coupled systems will increase the practical modes of access to computing services and hardware resources.

17. Key Words and Document Analysis. 17a. Descriptors

time-sharing
networks
mini-computer
remote batch
data concentrator
multiplexor
file migration
command languages

17b. Identifiers/Open-Ended Terms

17c. COSATI Field/Group

18. Availability Statement	19. Security Class (This Report) UNCLASSIFIED	21. No. of Pages 24
	20. Security Class (This Page UNCLASSIFIED	22. Price

FORM NTIS-35 (10-70) USCOMM-DC 40329-P71