

# TSO Times

Spring 2004

## In This Issue

### Region Size: What You See is Not What You Get

by Mark Zelden

### Random ISPF Tricks

by Lionel B. Dyck

### Chicago-Soft Joins Cornerstone Systems TechPartner

by Steve Caplan

### History of TSO: Part Three

by Jim Moore

### Dyanmic Allocation - The MVS Way

by Steve Myers

### REXX Comparison Tool

by Yash Pal Samnani

### Windows Keyboard Shortcuts

TSO Assistance for the MVS Professional

## What You See is Not What You Get

by Mark Zelden

Most people who have worked on OS/390 or z/OS for any length of time have probably run into various out of storage abends (x0A, x78, etc.) at one time or another. The fix may be as simple as increasing the REGION size in the JCL for a batch job, or logging on to TSO with a larger region size.

If there is no IEALIMIT or IEFUSI exit in place on the system, then the defaults for the REGION parameter documented in the MVS JCL Reference apply. Even the documented behavior for actual "region below" and "region above" can be confusing for some. For example, if you specify REGION=20M the default extended region size will be 32M, not 20M. Please see the MVS JCL Reference for more detail on the defaults. Then there is Unix System Services, also referred to as OS/390 Unix or z/OS Unix, which takes its default region size from the MAXASSIZE setting specified in the BPXPRMxx member of the system PARMLIB concatenation.

The problem is that most shops use an IEFUSI exit to modify the defaults and unless the rules of the exit are clearly documented, you may not know what to expect. It is also up to the system programmer to decide whether to invoke the IEFUSI exit for batch jobs, TSO users, started tasks and Unix System Services.

As a consultant who moves from site to site frequently, I found a simple way to determine what region size I was actually getting after IEALIMIT/IEFUSI exits (possibly) made their changes. The answer was in a control block called the Local Data Area (LDA) which is documented in the MVS Data Areas manual and is mapped by the IHALDA macro. I wrote a small REXX routine called REXXSTOR to display the requested and actual region size values from the LDA for both below the line and above the line virtual storage. It also shows the amount of storage used, but this may only be beneficial if invoked from an interactive TSO or Unix System Services (OMVS) session.

By running the exec from TSO, Unix System Services (or even a OS/390 web server) and in batch with various region sizes specified in the JCL, I could see what effect IEFUSI had on the region size.

The REXXSTOR source code and sample JCL to run it in batch are available for download at [www.tsotimes.com/currentarticles.html](http://www.tsotimes.com/currentarticles.html).

*Mark Zelden is an independent contractor with many years of development experience. His work has involved all manner of systems work, databases and troubleshooting.*

[www.tsotimes.com](http://www.tsotimes.com)

# Random ISPF Tricks

by Lionel Dyck

For this months article I am going to document some little known features of ISPF (based on ISPF under z/OS 1.3).

## ISRDTLCV

This handy REXX program is provided in the ISPF REXX library (ISP.SISPEXEC) as a tool for converting an ISPF panel that has been created using DTL so that you can make updates to the panel without knowing DTL. A prime example of when to use this tool is when you want to make an update to ISR@PRIM to add some local installation menu options. To use ISRDTLCV, first get into ISPF Edit on a copy of ISR@PRIM, and then execute this tool which is an ISPF Edit macro. The result will be that all the hexadecimal codes in the panel will be converted to something that you can actually edit and see, and which are valid in the panel.

## ISPLIBD

On rare occasions, more so if you develop ISPF dialogs, you may have a need to see what the current set of libraries are that have been allocated via LIBDEF. This command will do that by simply entering ISPLIBD on any ISPF command line. If you are interested in only a specific library you can enter ISPLIBD library (e.g. ISPLIBD ISPLIB).

## ISRONLY

Provided in the ISPF Samples library (ISP.SISPSAMP) this handy REXX program should be copied into a library in your SYSPROC or SYSEXEC concatenation, typically under the name ONLY. Then whenever you are in ISPF Edit just type in the ISPF command line ONLY xxxx to display only those records with the string xxxx. The parameters for ISRONLY and the format of the string are the same as for the FIND command with the exception that FIRST, LAST, NEXT, and PREV are not supported.

One side note that most users of this command are unaware of is that while processing sequence numbers are turned off and then turned back on when the command completes. The sequence numbers are not removed when numbers are off, they just become part of the data for this short period of time.

## ISRBPDF and ISREPDF

These two samples are provided in the ISPF REXX library (ISP.SISPEXEC) and are very helpful if you want to provide a fast path command to ISPF Browse and ISPF Edit. To use these you must first Alias, Copy, or Rename these to drop the ISR and then update your ISPF Site command table (or ISPCMDS if you don't have a Site command table, which you should) with these entries:

```
BPDF  2  SELECT CMD(%BPDF &ZPARM)
EPDF  2  SELECT CMD(%EPDF &ZPARM)
VPDF  2  SELECT CMD(%EPDF &ZPARM VIEW)
```

You can use whatever name you like but these work.

Then tell your users that they can, from any ISPF command line enter BPDF data.set.name, EPDF data.set.name or VPDF data.set.name to directly enter ISPF Browse, Edit or View on the specified data set. The ISREPDF program has other options which you might find useful and for those you'll need to browse the supplied source code.

## 3.4 and DSLIST

When in ISPF 3.4 or when using the DSLIST command the result is the same, program ISRDSLST is executed. This results in a list of data sets that match your selection criteria. By this point in ISPF life nearly every ISPF user knows one or both of these options. What they may not know is that when in the data set list there are several primary commands which can be used to make life easier, among them are:

CONFIRM which can be used to turn on or turn off confirmation when a data set is deleted. To enable confirmations enter CONFIRM ON and to disable enter CONFIRM OFF on the ISPF DSLIST command line.

DSLSET can be used to bring up a popup panel where you can change the default behavior of the DSLIST.

EXCLUDE can be used to hide data sets with a specified character string. Use the ALL option if you want more than just the first occurrence to be hidden.

*...continued on next page*

## MVS/Quick-Ref™

**The #1 Programmer's Choice**

MVS/Quick-Ref is an on-line quick reference tool for users of the MVS operating system. It has the ability to "pop-up" over an active ISPF application and give you a quick answer to an MVS-related question and then go away without affecting the active application.

**Instant access to over 27 million lines of MVS/OS/390 reference information.**

**Why do you need MVS/Quick-Ref?  
It saves you time and money!**

For more information or to order a FREE trial, give us a call at 603-643-4002 or visit us on-line at [www.chicago-soft.com](http://www.chicago-soft.com)



## Random ISPF Tricks

---

*...continued from previous page*

MEMBER followed by a member name or member name pattern can be used to find all data sets in the list (X for only excluded, NX for only those not excluded). Data sets which are migrated will not be searched unless you use options RECALL1 for data sets migrated to DISK or RECALL2 for all migrated data sets. This is really handy but don't be fooled into thinking you can enter a B (to browse) the member of the data set as a B will invoke ISPF Browse on the entire member list (this might be a good requirement to submit to IBM to allow the B (or whatever command) to operate on the member or members found by the MEMBER command).

SAVE will save the current list of data sets to a data set which can then be processed or printed.

RESET will undo the EXCLUDE.

SRCHFOR will invoke the ISPF SuperC utility to search all of the non-migrated data sets. After entering SRCHFOR on the ISPF DSLIST command line you will be presented with a panel where you can enter the search arguments and select various search criteria such as only search online data sets.

There are a lot of 'hidden' capabilities within ISPF which you can find by reading the tutorials or (gasp) reading the documentation.

This column in TSO Times is for your education and enjoyment. If you have questions about anything in TSO or ISPF send them to [information@tsotimes.com](mailto:information@tsotimes.com) and we will address as many as we can each month (and if there are no questions then we will be creative and write something that you will find worthwhile).

---

*Lionel B. Dyck has been in data processing since 1972. Over the years, he has developed many TSO/ISPF tools to simplify repetitive or complex tasks. Many of Lionel's tools can be found on the old SHARE tools tape, the CBT Tape, and also at his web site: <http://www.lbdsoftware.com>*

## Chicago-Soft Joins Cornerstone Systems TechPartner

---

*by Steve Caplan*

Chicago-Soft, Ltd, has entered into an agreement with Cornerstone Systems, Inc., of Irvine, California, to participate in their newly formed TechPartner program. The TechPartner program provides select "best of breed" software, preloaded for easy evaluation, on zFrame™ S/390 servers from Cornerstone Systems. It's the S/390 version of "bundled" software.

Cornerstone customers purchasing a zFrame™ S/390 server, running OS390/zOS, will find Chicago-Soft's flagship product, MVS/Quick-Ref, already installed on their system. MVS/Quick-Ref is the premier online documentation program for MVS, OS/390 and z/OS. All that will be required to begin a free trial evaluation will be an evaluation code zap, available from Chicago-Soft. Cornerstone customers who purchase MVS/Quick-Ref through this program will receive special TechPartner discount pricing from Chicago-Soft.

Cornerstone Systems, founded in 1990, is a "Tier 1" IBM Premier Business Partner. They are authorized and trained in the sale and support of the IBM eServer Systems products, including IBM's zSeries (s/390), pSeries (RS/6000) and xSeries (Netfinity) processor lines, Enterprise Storage Server ("Shark") disk sub-systems, Storage Area Networks (SAN) and Virtual Tape Servers (VTS). In addition to sales, implementation and support of these systems, Cornerstone provides a variety of services in the OS/390, VM, z/OS, Linux on 390, AIX, HP-UX, Linux and Solaris operating systems and environments. Their services include everything from complete systems outsourcing and back up, enterprise network security, to application development.

More information about the Cornerstone Systems and the TechPartner program is available on their website: <http://www.csihome.com>

---

*Steve Caplan is Vice President of Strategic Partnering for Chicago-Soft, Ltd.*

TSO Times is published by Chicago-Soft, LTD., which develops and markets corporate productivity and management software for main-frame, Windows and web-based systems. Intended as a forum for the exchange of technical information of interest to TSO and ISPF users, it's content will be mostly technical, with some promotional materials.

All inquiries concerning subscriptions, requests and change of address should be sent to Chicago-Soft, LTD, ATTN: TSO Times, One Maple Street, Hanover, NH 03755, TEL: 603-643-4002, FAX: 603-643-4571, email: [information@tsotimes.com](mailto:information@tsotimes.com)

Additional information can be found at [www.tsotimes.com](http://www.tsotimes.com).  
Chicago-Soft, LTD information can be found at [www.chicago-soft.com](http://www.chicago-soft.com)

*Copyright © 2003 Chicago-Soft, LTD All rights reserved.  
All trademarks are property of their respective holders.*

# The History of TSO ~ Part Three

by Jim Moore

I received my first computerized “instant message” sometime in the summer of 1976. I was busily writing code one morning on a pin-feed, paper-based, 300 baud DEC Writer (with a bi-directional print head!) when suddenly, after I pressed ENTER, a message printed out.

I turned to my co-worker (Eric McConney, an early mentor of mine) who was working next to me in the bullpen and asked him:

“Eric, what the heck is this? Where did this message come from?”

He leaned over, looked at the printed message and said:

“Someone up on Chicago Avenue is asking if you want to go to the ball game tonight. Just reply back and ask who they are.”

“How?” I asked Eric.

After a quick lesson from Eric in the use of the TSO SEND command, I instant messaged the sender back in reply.

It turned out that the person sending me the message had mixed up my TSO-id with a similar one. Alas, no White Sox game for me that night but that didn’t matter much to me. I was far more intrigued with the message sending concept. Ball games come and ball games go on a daily basis. But this instant, textual communication technique was something brand-new to me back then and I was spellbound by it. What a great new toy!

## TSO as Networking Software

In Part Two of this series, I mentioned that an important feature of mainframe TSO is its networking capability. Considering the humble TSO SEND command is one way to put TSO’s networking capability into perspective. The only conceptual difference between something like the TSO SEND command and Internet instant messaging is the number of people that can send and receive messages.

With mainframe TSO, the network consists of currently logged on TSU address spaces plus any operator consoles. With Internet IM, the network is vast, world-wide and far-flung. But in concept, both work quite similarly—instant textual communication across some kind of network of logged-on users.

It is the polling done by the Terminal Controlling Address Space (TCAS) on an MVS LPAR that allows for instant messaging via the SEND command. This polling is the same mechanism that recognizes a press of an AID key (See Sidebar One) by a TSU. Once a key press is detected, the TSU address space is swapped-in, made active and the requested processing is done.

When the requested processing has finished, the entire TSU address space is swapped-out. Think of all the times that you have been in ISPF edit, looking at program code, JCL or other data. As long as an AID key has not been pressed, your underlying TSO session is in a dormant state, or as it is more commonly known, swapped-out.

## What’s an AID key?

This is an IBM term that combines the word “attention” and the acronym “ID”, or *attention identifier*. Examples of AID keys would be ENTER, CLEAR, PA1, PA2 and all of the function keys. These keys were part of the hardware of the 3270 series terminals. An AID key requests the attention of whatever underlying operating system is being communicated with on a 3270 system. When you press an AID key, it becomes the first component in the 3270 data stream. It describes what action caused the data stream to be sent. The most frequently pressed AID key is ENTER, especially in TSO/ISPF.

## Conversational, Again

Getting a TSO session to remain active on the operating system’s chain of jobs, waking up and performing work only upon the press of an AID key, was the real breakthrough for those long-ago IBM developers of TSO. The swap-in, swap-out (or *wake up, go to sleep*) nature of a TSU is what allowed a true “sharing of the time” between multiple, signed-on users.

TSO retains this exact characteristic to this day. What must always be remembered is that even in a swapped-out state, any resources held by a TSO session (such as allocated files, memory and even the occupied slot in the Address Space Vector Table, or ASVT) remain held even when the TSU address space is in a swapped-out state.

I know I’ve said this before, but it certainly bears repeating: This is what is meant by the term *conversational*. As long as the TSU address space is logged-on, it will hold on to something.

## SEND Command Revisited

Consider the earlier SEND command scenario.

I was logged on to TSO, typing code in an editor but not pressing an AID key. Some other TSO user on the network typed in a SEND command with a message routed to my TSO session.

...continued on page 7

## Why ISPF uses the semi-colon as the default command stacking character

A unique key on older IBM terminals was known as the *Field Mark* key. It was purely a terminal type key and had no equivalent EBCDIC (hex) representation. When pressed, the screen image of it looked like a semi-colon with a short, horizontal bar above it. It was not an AID key.

In line-mode (Ready-mode) TSO, the Field Mark key could be used to “stack up” a string of commands and invoke them all with one press of the ENTER key (one swap-in). When ISPF added command stacking capability, it must have seemed natural to the IBM developers to allude to the Field Mark and use the semi-colon as the default dividing character for an ISPF command stack.

# Dynamic Allocation ~ The MVS Way

Steve Myers

The first TSO release in 1971 provided a limited form of dynamic allocation. It is still there and it still works. A separate article, available on the TSO Times web site discusses TSO dynamic allocation. It is easier to use, particularly for reentrant programs, than MVS dynamic allocation, but it can only be used in a TSO environment.

One of the big changes in the first MVS release MVS was the introduction of dynamic allocation. However, MVS dynamic allocation works differently than TSO dynamic allocation. MVS is much more powerful and much more extendable than TSO dynamic allocation, but power comes with a cost in complexity. It is also more difficult to use, particularly for reentrant programs.

Table 1 shows a complete dynamic allocation request.

	LA	R1, ARBPTR
	DYNALLOC	,
	.	.
	.	.
ARBPTR	DC	A(X'80000000'+ARB)
	SPACE	1
ARB	DC	0A(0), AL1(S99RBEND-S99RB, S99VRBAL, 0, 0)
	DC	2AL2(0)
	DC	A(ATXTPP)
	DC	2A(0)
	SPACE	1
ATXTPP	DC	A(ATXT01, ATXT02, ATXT03, X'80000000'+ATXT04)
	SPACE	1
ATXT01	DC	AL2(DALDSNAM, 1, L'ADSN)
ADSN	DC	CL44'ZYS1.MACLIB'
ATXT02	DC	AL2(DALRTDDN, 1, L'ALDDN)
ALDDN	DC	CL8' '
ATXT03	DC	AL2(DALRTORG, 1, 2), AL2(0)
ATXT04	DC	AL2(DALSTATS, 1, 1), AL1(X'08')

Everyone using dynamic allocation must be familiar with two IBM macros in SYS1.MACLIB. The IEFZB4D0 - believe me, you'll memorize this crazy name - macro defines the symbols for the definition of the request block pointer, the request block, the text unit pointer list, and the text units. IEFZB4D0 also defines the symbols for a data area called the request block extension, which we will not discuss. The IEFZB4D2 macro defines the symbol names used in the text units.

## The Request Block Pointer

The request block pointer - ARBPTR in our example - is real simple. It contains the address of the request block, and the high order bit must be set on.

## The Request Block

The request block is a 20-byte data area that tells about the request. The first 4 bytes are flag bytes.

- S99BLEN - The length of the request block. The example shows the correct way to specify the field.

- S99VERB - The type of allocation request
  - o S99VRBAL - A "regular" allocation request
  - o S99VRBUN - An unallocate (free) request
  - o S99VRBCC - A concatenation request
  - o S99VRBDC - A de-concatenation request
  - o S99VRBRI - A remove-in-use request
  - o S99VRBDN - A DD name allocation request. This is really a request to take an existing allocation and increase the in-use count
  - o S99VRBIN - An information request from an existing allocation
- S99FLAG1
  - o S99FLG11 - Miscellaneous flags. They only need to be present for full allocation requests. Some of them are:
    - S99NOCNV - Make sure a new allocation is used, rather than an existing allocation.
    - S99NOMNT - Do not allow a volume mount. This is more useful for batch than TSO, since most TSO profiles forbid a volume mount anyway.
    - S99NOMIG - Do not recall a data set from archival storage to complete an allocation.
  - o S99FLG12 - No bits for us peons are present.
- S99RSC
  - o S99ERROR - An error code
  - o S99INFO - Additional information to document the error code.
- S99TXTPP - The address of the text unit pointer list.
- S99S99X - The address of the request block extension.
- S99FLAG2 - Flags us peons can't use.

## The Text Unit Pointer List

The text unit pointer list just points to each text unit. The last entry has the high order bit set. It is a list of addresses.

...continued on page 7

## FREE SUBSCRIPTION

Was this issue of the TSO Times mailed directly to you? If not, would you like to receive your own copy? If so, please visit [www.tsotimes.com/subscribe.html](http://www.tsotimes.com/subscribe.html) and fill out our on-line subscription form to start receiving your own copy. It's that easy!

# REXX Comparison Tool (CA)

by Yash Pal Samnani

Comparison of old and new source code is essential to ascertain whether the changes done to programs (JCL/ Control Proc/ Copybooks) are in sync or not. When using the IBM provided COMPARE command, this often requires long comparison commands such as COMP < pds (member name) >. This problem of typing long comparison commands is compounded by the use of different production library names. All of this extra typing consumes time throughout the process of coding and testing the programs.

My focus was to eliminate the wasted typing time. If this waste could be avoided, programmers would be more efficient. I propose a solution that will save users a lot of time and increase their productivity. The solution is the use of a REXX edit macro (named CA) that contains a list of all the libraries that could ever be used to compare changed components against. That is, the list consists of the baseline component libraries divided by type. Determining which library to compare against is handled by logic in the CA REXX edit macro. The tool automatically recognizes the component being compared. There is no need to specify whether the source is a program, proc, control proc or a copybook.

Assume that the following production libraries are installed at a site. Each and every time the CA macro is executed, these libraries (or additional ones, if desired) will be compared at the member level:

```
BOXA.PROD.TECH.SRCE
BOXA.PROD.TECH.PROC
BOXA.PROD.TECH.CTLPROC
BOXA.PROD.TECH.COPYLIB
```

When you type CA, the member name (that needs comparison) is automatically picked up from the dataset being edited and is searched sequentially in production (or any other pre-defined libraries). These libraries could be source, proc, control proc or copybook library. Wherever the member is found, it is compared and the comparison is displayed.

The comparison results will be fine even if the production library has a member in proc and source with the same name. This is achieved by additional logic that uses the presence of keywords like IDENTIFICATION, ENVIRONMENT in the source to determine what type of member is being edited.

Figure 1 - Command ==> CA

Entering the CA edit macro without a member name.

This is also helpful to mainframe shops that use the version control software named Changeman. Type CA on the command line and press enter. Even though Changeman provides this feature as a part of its software, CA will still be effective. One exception: In the case of Changeman, if the component is a control proc or a copybook then a member name is required. There is no need to specify the member name in case the component being compared is a program or a proc.

Figure 2 - Command ==> CA member name

Entering the CA macro with a trailing member name

If the member exists in the production library, then the comparison will be done. Otherwise, the message **‘Attempt failed! Either the component is not available in the Production Library OR the component being compared is other than Proc, Source, Control Proc or Copy Book’** will be displayed in a pop-up message window.

## List of possible message for different components displayed on top left hand side of the panel

Mismatch found -	Program:	Source compared!
	Proc:	PROC compared!
	Control Proc:	compared!
	Copy Book:	Copy Book compared!
No mismatch found	Program:	Files are same!
	Proc:	Files are same!
	Control Proc:	Files are same!
	Copy Book:	Files are same!

Yash Pal Samnani is a computer programmer who develops ISPF productivity aids for himself and co-workers.

## Windows Keyboard Shortcuts

Here is a table of some of the most useful Windows keyboard shortcuts. With minimal practice, you can use these shortcuts to work faster and relieve stress on your mouse hand. In the table, “WK” indicates the “Windows Key”, the key with the small Microsoft Windows logo on it.

Keyboard	What it does	Where Used
F5	Same as Refresh	Any Internet browser
Ctrl-C	Copy text to clipboard	Anywhere in Windows
Ctrl-V	Paste text from clipboard	Anywhere in Windows
Ctrl-X	Cut text to clipboard	Anywhere in Windows
WK-M	Minimize all Windows	Anywhere in Windows
WK-R	Invoke Windows Run	Anywhere in Windows
WK-E	Invoke Windows Explorer	Anywhere in Windows
WK-D	Minimize/Undo Minimize	Anywhere in Windows
WK-F	Invoke Find Files Dialog	Anywhere in Windows
WK	Pop-up the Start Menu	Anywhere in Windows
WK-Tab	Move between open apps	Anywhere in Windows
Alt-F4	Close an open window	Anywhere in Windows

For more Windows keyboard shortcuts, read the Windows Help text, accessible by pressing WK-F1!

## Dynamic Allocation ~ The MVS Way

continued from page 5

### The Text Units

The text units, taken together, describe the allocation in detail. Some text units provide information for dynamic allocation. Most text units are an exact equivalent of a JCL parameter. Other text units request information from dynamic allocation. As such, they do not have a JCL equivalent.

The IEFZB4D2 defines symbols for all the key names used by dynamic allocation. **USE THEM!** The first byte of each key name is D. The second and third bytes are a copy of the last two bytes of the symbolic dynamic allocation request verb used in the request block. All the key names for full allocation, S99VRBAL, start with DAL. The remaining 5 bytes are suggestive of the function. For example, DALDSNAM means you are specifying a data set name. All text units start as 2 half-word data areas. The first two bytes contain the IBM-defined text unit key. The second two bytes are the number of parameters that follow.

Each parameter contains a two-byte length field followed by the parameter.

Here are four text units taken from the online example program:

- AL2(DALDSNAM,1,44),CL44' ' - This text unit specifies a data set name. It has one parameter that contains 44 bytes, followed by the 44-byte parameter area.
- AL2(DALSTATS,1,1),AL1(X'08') - This text unit defines the allocation status. X'08' is DISP=SHR. Unfortunately there are no formally defined symbolic equates for this data area. However, IKJDAPB08 - one of the macros used with TSO dynamic allocation defines an equivalent field and it provides equates that are identical to DALSTATS.
- AL2(DALRTDDN,1,8),CL8' ' - This key requests dynamic allocation to return the length and text of the DD name assigned to the allocation. The DD name does not have to be 8 bytes. Dynamic allocation will update the length field with the actual length.
- AL2(DALRTORG,1,2),AL2(0) - This key requests dynamic allocation to return the DSORG of the data set it allocated. The data it returns is the same as DCBDSORG or DS1DSORG. The real reason to use this key is because dynamic allocation has to obtain this key from the VTOC entry for the data set. If the data set is cataloged, but it does not exist on the volume, dynamic allocation will fail the request. This is better than getting an S213 ABEND when you OPEN the data set!

The sample TSO command for this article is DYNALLOC.ASM. As written, DYNALLOC.ASM does not apply the normal convention of inserting the current prefix into the data set name if the data set name in the command line is not enclosed in quotes. What change would you make to the command so that this could be done?

---

*Steve Myers has been programming since 1965 and writing TSO command processors since the very beginning of TSO.*

## History of TSO ~ Part 3

continued from page 4

When the message sender pressed ENTER, his TSO address space was swapped in, the SEND command invoked and then swapped right back out again.

The message, routed to me, was placed in a temporary holding area of the broadcast dataset. The message *did not* immediately appear on my terminal.

Only when I pressed the ENTER key and in so doing, swap my TSO address space in, did I see the message. Additionally, any pending changes that I had made to the code I was working on were also updated in memory.

By the time I began reading the “take me out to the ballgame” message that suddenly appeared on my DEC Writer, my TSO session had already reverted to a dormant, swapped-out condition.

### Conclusion

Even back in the old days of TSO, there were methods for getting more bang for the buck when a TSO session was swapped in. Refer to Sidebar Two for an interesting historical connection between ISPF and TSO.

The final part of this four-part series is online at [www.tsotimes.com/currentarticles.html](http://www.tsotimes.com/currentarticles.html)

---

*Jim Moore is the Editor of the TSO Times.*

## TSO Times Advertising

### Rates

Full Page (7 1/2" x 10")	\$500
1/2 Vertical (3 1/2" x 10")	\$400
1/2 Horizontal (7 1/2" x 5")	\$400
1/4 Vertical (3 1/2" x 4 3/4")	\$150
Business Card (3 1/2" x 2")	\$75

Please submit your ad in one of the following PC-compatible digital formats: Illustrator, Photoshop, PageMaker or Freehand. Images/graphics should be saved in one of the following formats: .eps, .tif, .jpg, .bmp, or .wmf and have a minimum resolution of 300 dpi/ppi when scanned or scaled. You may send your ad on a cd-rom, floppy or Iomega ZIP disc to Chicago-Soft, LTD, ATTN: TSO Times, One Maple Street, Hanover, NH 03755 or by email to [speck@tsotimes.com](mailto:speck@tsotimes.com).

The TSO Times has a circulation of 25,000 people.

### Publication Schedule

Spring 2004 / Fall 2004 / Spring 2005 / Fall 2005



One Maple Street  
Hanover, NH 03755

Presorted Standard  
US Postage Paid  
New London, NH  
Permit #11

# TSO Times

***For the Power ISPF User***

Have you visited [www.tsotimes.com](http://www.tsotimes.com) yet? Several of the articles in the paper edition are continued online. The web site is also the place to go for code downloads, examples and other interesting articles that appear ONLY on the web site. Have a look. You'll be glad you did.

**NEW PRODUCT**

*From the Developers of  
MVS/Quick-Ref™*

## **CADII™**

**Cursor Assisted Direct Interface to ISPF**

- Easy cursor invocation with rapid execution
- Ability to remember data set names
- Ability to assign up to 100 different 5-character data set IDs for use instead of long data set names
- Simple and easy to remember command set

***Optimize ISPF Usage 50+%***

For more information, to download the CADII demo, or to order a FREE trial call 603-643-4002 or visit us online at [www.chicago-soft.com/cad战略](http://www.chicago-soft.com/cad战略)

## **FAXBACK - 603-643-4571**

To receive more information or product trials, simply fill out this form and fax it to Chicago-Soft, LTD at the number above. You can also go to our website at [www.chicago-soft.com](http://www.chicago-soft.com).

**Please send me:**

- More Information on:
- ISPF-Plus™
  - CADII™
  - MVS/Quick-Ref™
  - Quick-Ref for Windows™
  - Web/Quick-Ref™
- FREE Product Trial for:
- ISPF-Plus™
  - CADII™
  - MVS/Quick-Ref™
  - Quick-Ref for Windows™
  - Web/Quick-Ref™

Name: \_\_\_\_\_

Title: \_\_\_\_\_

Company: \_\_\_\_\_

Address: \_\_\_\_\_

City/State/ZIP: \_\_\_\_\_

E-mail: \_\_\_\_\_

Phone: \_\_\_\_\_

Fax: \_\_\_\_\_