One-on-One with
The Control Editor (TCE)

An Image Control Environment (ICE) Application

This booklet prepared exclusively for
NewEra Software

BACKUP—TAKE A BACKUP BEFORE MAKING CHANGES **TEST**—TEST CHANGES TO PARMLIB, PROCLIB AND/OR JCLLIB BEFORE COMMITTING THEM
HISTORY—RESEARCH THE HISTORY OF PRIOR CHANGES BEFORE ATTEMPTING NEW ONES **DOCUMENT**—DOCUMENT ACTUAL CHANGES AT THE POINT WHERE THE CHANGE TAKES PLACE **NOTIFY**—NOTIFY THOSE WHO NEED TO KNOW THAT A CHANGE HAS BEEN MADE

The Control Editor (TCE) – Premiering at IBM System z – Caesars Palace – October 1-3

[1]Expert z/OS TSO/ISPF Workflow Assistant; experienced with all releases. References Available on Request.

Foreword

I cannot tell you how many times I have had to get involved in a system security dispute between my Security Officer and System Programming Manager. Both are well intended, one wants more security of z/OS system changes; the other says no way. The Control Editor resolved this for me, for us. It was easy to set up and since it enhances ISPF it was a snap to learn. The Security manager has a Compensating Control with lots of event detail and reports while the System Programmers have a continuous backup of critical datasets and ISPF enhancements that give them direct access to a timeline of change history and validation of syntax and member construction.

– zEnterprise CIO

Executive Summary

Sound z/OS System Programming *Best Practices* are straightforward and simple enough, but we're all human, all busy, we all forget and our best intentions to conform to these practices will sometimes go unfulfilled. Do you:

- Take a **Backup** before making changes to z/OS Configuration components?
- **Test** changes to PARMLIB, PROCLIB, JCLLIB before committing them to production?
- Research the **History** of prior changes before attempting new ones?
- **Document** Actual changes at the point where the change takes place?
- Finally, **Notify** those with a need to know that a change has been made?

No *Backup*, no *Test*, no *Review*, no *Documentation*, no Notification. Any of these can lead to a loss of z/OS integrity or compliance or worse -- to a loss of z/OS availability.

We will examine the workflow patterns commonly found in many System Programming Configuration Management tasks and how these tasks can be automatically conformed to *Best Practices* using The Control Editor (TCE), an expert TSO/ISPF workflow assistant we call the z/OS Nanny.

The Control Editor is an Image Control Environment (ICE) application. ICE enables users to enhance individual and team productivity by enabling them to quickly adapt to new business system requirements and progress toward reducing Total Cost of Ownership (TCO).

Table of Contents

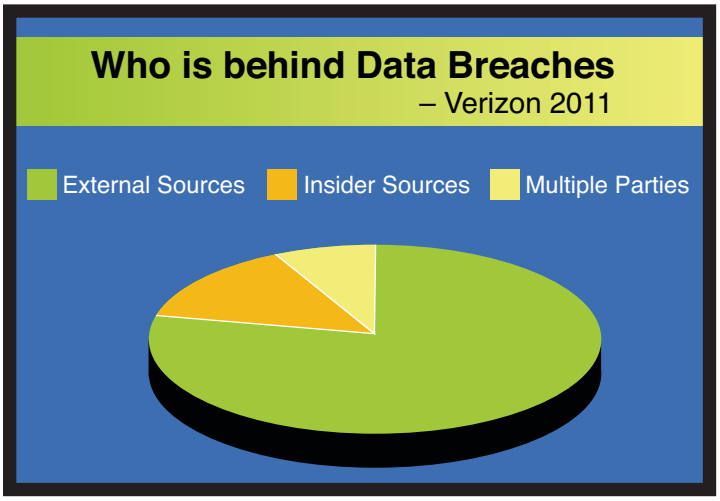
1 About This Book

This book is designed to help regular (non-technical) users of TSO/ISPF to understand the implications of working in an environment where The Control Editor (TCE) is implemented. What you will find are de-jargon-ified explanations of concepts and specific parameters. It is written in "Clear English".

It will take you through the decision making process that saw our organization pick TCE as a "Compensating Control" and give a high level explanation of the reasons why this type of control is becoming more necessary.

The screens that you will see and how to fill them in are also detailed here along with an explanation of what this all means to YOU in your day-to-day job.

2 The Case for Better Internal Controls



The 2011, Verizon Business RISK Team's Data Breach Investigations (carried out in cooperation with the U.S. Secret Service and the Dutch High Tech Crime Unit) revealed that while data breaches caused by external sources were increasingly more prevalent last year (due to a massive increase in small external attacks rather than a decrease in insider activity), internal source breaches were larger and had higher value impact to the business.

PCI-DSS Definition of a Compensating Control:

Compensating controls may be considered when an entity cannot meet a requirement explicitly as stated, due to legitimate technical or documented business constraints, but has sufficiently mitigated the risk associated with the requirement through implementation of other controls. Compensating controls must:

(1) Meet the intent and rigor of the original stated PCI DSS requirement;

(2) Provide a similar level of defense as the original PCI DSS requirement;

(3) Be "above and beyond" other PCI DSS requirements (not simply in compliance with other PCI DSS requirements); and

(4) Be commensurate with the additional risk imposed by not adhering to the PCI DSS requirement.

Other security standards also exist...

This emphasizes why the model, where data within the mainframe's sphere of influence was considered to be safe, must be replaced to allow for the new security needs.

Most data breaches assessed in the Verizon report result from a combination of events rather than a single trigger. Some sort of error is still a significant factor although this also leads to an increase in the number of hacker attacks where configuration errors are exploited.

In the banking and insurance industries, the most damaging form of data breach can be the exposure of customer details. This can lead to everything from distrust of an organization to the complete downfall of a business. But a simple "Denial of Service" attack could be just as damaging to a brand.

With over 150 pieces of legislation (such as SOX, SAS70, NAIC, PCI-DSS etc) in place across many Countries there has been no better time to solidify our Organization's Best Security Practices. And implementing The Control Editor as a "Compensating Control" allows us to fully address the three top concerns of the majority of the regulations that we are all trying to comply with what changed; who changed it; and on what authority?

3 Why We Selected TCE

TCE was selected for three specific reasons: lowest total cost of ownership, seamless integration into the change process and its full set of compliance reporting tools.

Actually, it would be best to have you read about why some other organizations selected TCE:

> "…our system audit reviews are done as part of our financial audit process. We have been written up several times for not having adequate documentation of actual changes. We do a good job of documenting what we are going to do but not what we actually did. The Control Editor filled this hole in our change management process by requiring users to provide descriptive information, documenting each change at the point of the change using standard TSO/ISPF. No more negative audit findings."

> "…everyone knows that submitting JCL during a TSO/ISPF Edit session can open a big hole in z/OS system security. JCL can be edited and submitted, even by those without UPDATE authority, and then the Edit session cancelled. No one is the wiser and generally RACF, ACF2 and Top Secret are totally bypassed. This has been an open audit finding in our environment for some time. The Control Editor closed this hole for us. No more undocumented changes to or submission of our JCL."

> "…we have contractors coming in and out of here all the time. We give them pretty much the same access we give to our own system programming staff. Until we began using The Control Editor we had no idea what they were actually doing, what changes they were making. The TCE reporting and query functions resolved this completely. I now know who did what, to what and when."

Whatever the reason you now find yourself in a position to use TCE, you will learn to value it as a tool. It couldn't be easier to get started and with just a little practice, you can get TCE to provide you with all sorts of useful functionality and insight into configuration changes.

Most users particularly like the automatic backup of members that TCE takes before saving any changes to controlled datasets – just like you get with Microsoft Word or Excel. In fact, just that facility can be enough to encourage an organization to roll the product's sphere of influence out to a much broader target than anticipated.

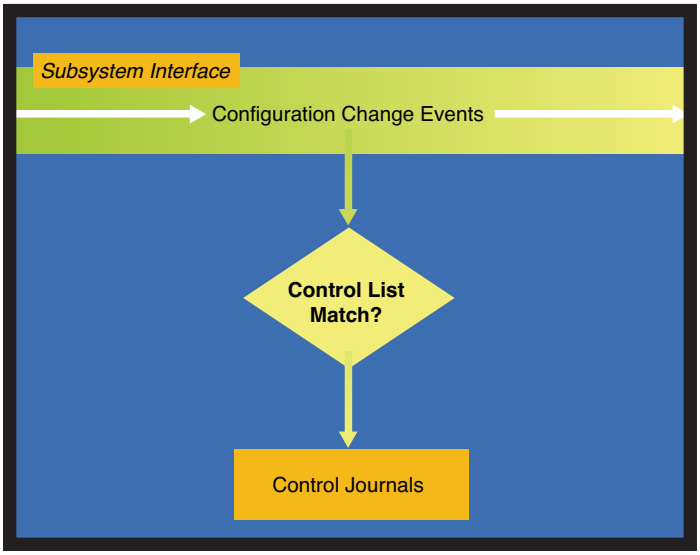
4 What is TCE?

Change Management Control products provide a balance between the needs of Users who must have access to critical z/OS resources and the desire of security officers, compliance managers and auditors to establish effective management controls. TCE is a z/OS Software Utility designed by NewEra Software to provide change management control **at the point when and where the actual change is made.**

TCE allows us to acknowledge that there is risk associated with keeping any z/OS system running but demonstrate that you are doing everything that you can to mitigate the risk – the very definition of a "Compensating Control". It does this by addressing the most basic issues associated with internal control over the z/OS configuration, namely who actually changed what, when, where and how.

4.1 What is it, Generally?

TCE is a "Compensating Control" that provides a layer of non-invasive security over z/OS configuration components housed in sets of partitioned and sequential datasets which have been identified to it as important to the organization. It significantly enhances the level of security provided by the organization's External Security Manager (ESM), i.e. CA ACF2, CA Top Secret or RACF.



In addition, TCE can be configured to identify and take action on the occurrence of specific System, Health Checker and External Security Manager Messages and record Change Events detected by the NewEra Family of Supplemental Change Detectors.

All of which means that TCE represents a clever box of tricks which can help you immediately by:

- automatically backing-up members before changes are saved;

- helping make sure that changes are syntactically correct (a function not available in base z/OS);

- allowing a level of automation of responses to specific problems; and
- giving insight into what has been changed and by whom in the case of a major malfunction with the system.

4.1.1 Dataset Control List

Datasets managed by TCE are called “Controlled” or “Boundary” datasets. To be either, our TCE Administrator must define the dataset to The Control Editor’s “Control List” by specifying the details of the list in the NSECTLxx configuration member. Once correctly defined, members found within Controlled Datasets, called “Control Members” or “Control Points”, will fall under the control and management of TCE.

You will either encounter an already “Controlled” dataset or you may wish to ask our TCE Administrator to include a new one.

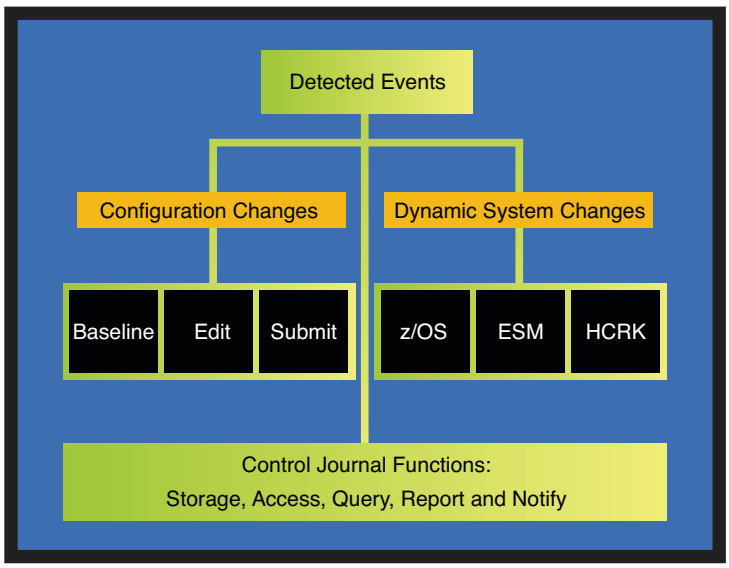
4.1.2 Control Journals

Controlled Events defined to TCE are captured and stored in Control Journals as they occur. If any change is made or detected to one of our Controlled Datasets, a new backup is made and the event is recorded in the Control Journal. These Control Journals are what you run your query against when finding out what has happened, for example, when you use the “History” command described later.

4.2 What is it, Technically?

Geek speak alert - TCE can be thought of as a “Listener” on a TSO/ISPF subsystem interface that allows it to “Hear” all “Events”, recording only those that match a predetermined event profile (Control List) and optionally logging all defined events (Control Journals) when forensic system analysis is required. These processes require no z/OS modifications, “Hooks” or “Exits” and are totally within the TCE Administrator’s control.

Any edit activity initiated against any partitioned dataset in the control list will activate TCE and begin the recording process. It really is as simple as that!



4.3 What are its Functions?

TCE is an ISPF editing platform from which we can both control and manage the access to critical system configuration datasets (Control Datasets) and productively identify changes to them and other system components known as Controlled Resources.

At its most basic level, TCE detects, records and reports on events that change z/OS LPAR configurations. This is important because such changes, which can fundamentally change the shape of the z/OS environment, could lead to inter-

ruptions to services or even a complete Denial of Service. Accidental or deliberate, the end result is the same – potential damage to OUR business.

By detecting any change that could otherwise go unreported or unrecorded, TCE provides a comprehensive centralized solution, without extensive in-house development. It should increase levels of confidence that you are able to avoid disruption to service levels, and in the event of a recovery situation that you are able to identify the root cause of the problem.

5 How the Control Editor Works

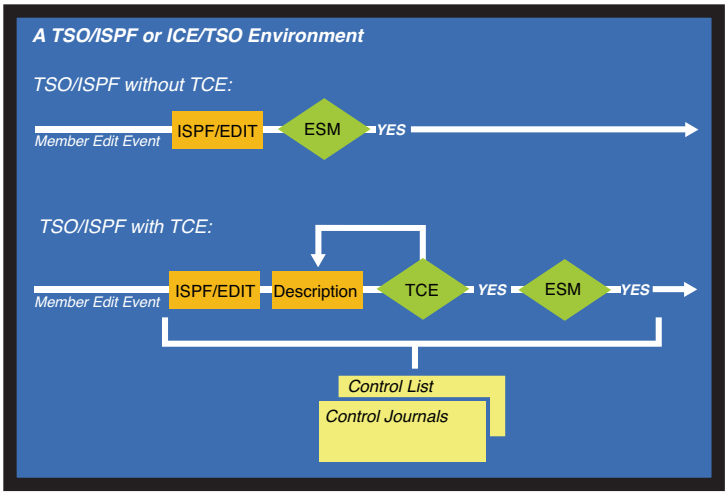
Keeping track of changes to important system datasets and configuration settings is a difficult job, and TCE is here to help. It does this by ensuring that any changes to these datasets have to be fully documented, and taking a copy of the relevant member before the change is made (great for those little mistakes we all make at some point or another).

This helps to make sure that any change is linked to a specific Request For Change (RFC), and allows you, by going back through the RFC history, to check that the change has been tested and that it is less likely to cause an issue when the change is made that impacts a production system.

TCE can be enabled and configured to step into the member save routine in one of two ways. The appendices explain how the running of a simple script supplied with the product allows TCE to intervene when we attempt to save the changes to a member that they have edited. TCE in no way impacts the way that our External Security Manager (ESM) protects datasets. This makes sure that only users ALREADY authorized to view or make changes to datasets can do so.

Dataset changes can be attempted in a number of different ways and TCE is ready to step in for Member delete, rename, copy, restore, and move actions as well as simple edit and save operations.

TCE can also keep track of and document configuration changes that have been applied dynamically. This allows a full audit trail should a change be made that did not require a product to be recycled, or an IPL to take place.



In the diagram to the left the first example shows a normal ISPF edit session save, the second shows how TCE steps into the process.

Because TCE saves copies and a history of changes made, you can track back through entries should an issue be detected at a later date and even roll back changes.

6 What You Can Expect

Installing and using TCE will just add a single step to the user's normal process for saving a dataset. This process will all happen automatically when the product is installed and activated using the CEINIT script. NewEra has made this process as simple as possible so that we can see the benefits that TCE brings almost immediately.

6.1 First Steps

Installing and customizing TCE is the responsibility of the Systems Programmer. From this point forward, we assume that process has been completed and all the necessary steps have been taken to allow you to use TCE.

However, if you're interested, it is achieved by running a supplied executable as your TSO/ISPF session starts up, i.e. when you Logon to TSO.

This is of course all fully documented in the installation process in the manual and makes it easy to ensure that TCE is helping us to protect our system configuration.

6.2 Descriptors

When an attempt is made to edit and save a member in a controlled dataset, TCE steps in and the Panel Descriptor is displayed (see supplied sample below). This panel can be customized to meet our site requirements for the documentation of changes so what you see may differ slightly.

You can see in the example below that one of the fields has been completed with incorrect characters. This is the project number, where we have specified it must be numeric, but the details have been entered as alpha characters.

When you have completed the necessary fields with the specified parameters, you will get the small pop up question box, shown on the next page, asking you to confirm that all of the necessary details have been completed.

```
          TCE 7.5 SAMPLE CUSTOM PANEL DES          Must be numeric

             Your Company Name Here:   Descriptor Data Entry DDE@PNL9
Option  ===>

  Change request #    : 0000000001          Implementor: MAU

  Project #           : abcdefghijk

  Implementation date: 2012/07/11
                      (yyyy/mm/dd)


 Change details:

                                                                More:     +
 Part of the conversion process from HFS to zFS required a change to BPX
 Parameters
```

These descriptor panels are fully functional ISPF panels, and as such, different panels can be defined for different tasks. This means that different criteria may have to be entered dependent on what task is actually being done, and the data that is entered into the fields can be checked to ensure correctly formatted data has been entered into each of them. Once you have entered the required data it is saved as a Control Journal in TCE.

```
------------------------------------------------
          Descriptor Panel Data Entry

     Descriptor data entry complete (Y/N):

               Press END to exit

------------------------------------------------
```

Another major advantage of using set fields for entry of data is that these fields are completely searchable, and available for search as soon as they have been entered. This makes the auditing and checking of any changes a much easier task and allows you to track back to exactly when a change was made.

6.3 History

The History function that exists in TCE allows you to view changes made. You can view changes chronologically for the entire system, or for an individual member. You can also sort this information by the fields that you have specified in your panel descriptors.

Accessible by typing 'History' on the command line this function is very helpful when auditing a system, or when tracking back through changes to see when a system parameter was altered. You drill further down through the history by placing your cursor on the values in the right-hand section of the screen and pressing enter. It is particularly useful when trouble-shooting in an attempt to find when

```
        TCE 7.5 - Control Journal - Event Selection   Row 1 to 6 of 6

----ICE 10.0----                                          --Member Events--
------ Journals IFODSN - Control DSN(MBR) CEDIT.VENDOR.PARMLIB(BPXPRMVN) ------

Option ===>                                                  Scroll ===> CSR

  Selection: Cursor under Value then press enter to display History Worksheet

- Row ------------Event Targets---------- Your ---------Period to Date---------

S Num -Class- -----------Name------------ News Days Week Mths Qtrs Years Totals

_ 001 Updates Member_Edits_and_Changes       9    1    3    9    9     9      9

_ 002 Detects Auto_Detect_Member_Changes     3    0    1    3    3     3      3

_ 003 Submits Member_JCL/JOB_Submissions     0    0    0    0    0     0      0

_ 004 Excepts Exceptional_Member_Events      0    0    0    0    0     0      0

_ 005 ------- -------------------------- ---- ---- ---- ---- ---- ----- ------

_ 006 Totals  All_Journaled_Member_Events   12    1    4   12   12    12     12

******************************** Bottom of data ********************************
```

certain functions were introduced or altered should they have had an impact on certain parts of your z/OS platform.

The subsequent screen will list the selectable Event log entries for the changes that were made.

6.4 Inspect

The Inspect option allows the configuration parameters entered in a Controlled member to be checked for integrity. To perform an inspection you should just type 'INSPECT' or "I" on the command line when editing a member. TCE will then check the member for any possible problems. In the example below you can see that I have chosen to inspect the CLOCK00 member that affects the system time.

```
EDIT       CEDIT.VENDOR.PARMLIB(CLOCK00)                  Columns 00001 00072
Command ===>                                                Scroll ===> CSR
****** ***************************** Top of Data ******************************
000100 OPERATOR NOPROMPT
000200 TIMEZONE W.00.00.00
000300 ETRMODE  YES
000400 ETRZONE  YSE
000500 STPMODE  YES
000600 STPZONE  YES
000700 TIMEDELTA 10
****** **************************** Bottom of Data ****************************
```

The resulting display below shows that an issue has been identified and even tells you which parameter is in error, and even which level of z/OS is required for certain parameters to be used.

```
BROWSE    SINGLE_MEMBER_INSPECTION                 Line 00000000 Col 001 080
Command ===>                                                Scroll ===> CSR
******************************** Top of Data ********************************
....IFO1818E ETRZONE - INVALID SUBPARM "YSE" FOUND.
....IFO1813W STPMODE - REQUIRES Z/OS V1R7 OR ABOVE.
....IFO1813W STPZONE - REQUIRES Z/OS V1R7 OR ABOVE.
....IFO1813W TIMEDELTA - REQUIRES Z/OS V1R7 OR ABOVE.
....IFO0676N CLOCK00 MEMBER INSPECTION COMPLETED WITH ERRORS.
******************************** Bottom of Data ********************************
```

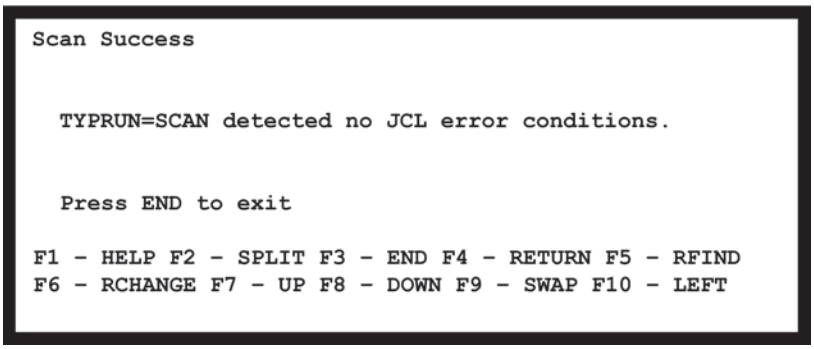
The INSPECT command can be run before the member is saved. This allows you to check your changes prior to committing them to the dataset, and the change having to be documented in TCE. Running the inspection will minimize the potential for any issues when a member is recycled or an IPL is performed.

The Inspect option forms part of what is called the NewEra Inspection Server and is one of the standard tools available with TCE, and is an extension of the product that helps you to verify that any changes you have made are syntactically correct.

An Inspection can only be performed against members that TCE is expecting to be used as part of the IPL process, such as system parmlib datasets and their members.

6.5 Scan

The Scan command allows TCE to check your JCL for you before you save it to a protected library. You can now check for JCL errors prior to saving and submitting you JCL. The command works by effectively submitting your JCL with TYPRUN=SCAN on your job card, but it is no longer submitted to a queue before the syntax can be checked.



The SCAN command is available when you are editing a member and should be entered on the command line. It reports back with any JCL errors that are found, or no errors are found if you have everything syntactically correct.

You can significantly reduce the number of changes that have to be recorded in TCE by scanning your JCL before you submit or save the member in a protected library.

7 Your Continuing Responsibility

While TCE introduces an internal compensating control into our change management process, it is still up to you to report any concerns that are brought to your attention by the product. TCE can be used as an additional tool when highlighting any potential issues, and the history function gives you the possibility to roll back the change that has been identified as a possible risk.

Should you identify anything when you are editing a parameter file, or see change information in TCE that concerns you, ask questions of either the project owner or the person that has made the change. Escalate until you have received a satisfactory explanation, or your concerns have been addressed.

You are a valued internal control point and it is up to you to help ensure that any potential risk is avoided. There may of course sometimes be circumstances where this risk has been assessed and is seen as acceptable (such as an emer-

gency change), but you should never assume that an issue has been considered. Teamwork and change management have been trying to achieve this for a number of years, and TCE is an extra factor that can help you ensure that our systems are more stable.

Audit points can now be addressed with documented evidence to show when and why a change was made. The date that the change was made can be identified as TCE ensures that it is documented at the time that the parameter is altered. You are no longer reliant solely on the information from change management. Should a change be delayed, TCE records the actual time together with the userid of the person that made the change, and can roll back the change if needed.

All of this helps to build a more stable, auditable z/OS environment, helping us to ensure that system Service Level Agreements (SLA's) and compliance are achieved. As z/OS professionals this is our goal, and TCE is a useful tool to help us reach it.

Appendices — Tips and Tricks

A Manually Activating The Control Editor

You can start (and, in an emergency, stop) TCE functions for your own current TSO/ISPF session, if you are authorized, using the supplied scripts found in hlq.CETSO.SISPCLIB.

These scripts can be run from the ISPF Command Shell by entering the following commands.

EX 'hlq.CETSO.SISPCLIB(CEINIT)' - to start The Control Editor functions

EX 'hlq.CETSO.SISPCLIB(CETERM)' - to stop The Control Editor functions

However, it should not be necessary for you to do so.

B Altering the Panel Descriptor Requirements

Descriptors are where the settings for TCE are stored. We can think of them as the parameters for the product. They are also used to define what information needs to be provided when documenting any changes to protected files.

The panel descriptor member name is DDE@PNL9, and this is the member that needs to be altered should you have different documentation requirements. It can be found in hlq.CETSO.SISPPENU. There is an associated help panel in this library that is displayed when you press F1. To alter the help panel you should make the changes to the @DDEHLP9 member.

You can also customize your environment so that the Edit Descriptor panel can be set to only be active during certain times of the day. If a change is made outside of the set times, the panel will not be displayed. (Alternatively, mandatory use of the panel could reinforce existing procedures, e.g. for emergency changes made outside normal hours.)

Making changes to these panels requires some coding experience and should only be performed by someone who is knowledgeable about ISPF dialogs.

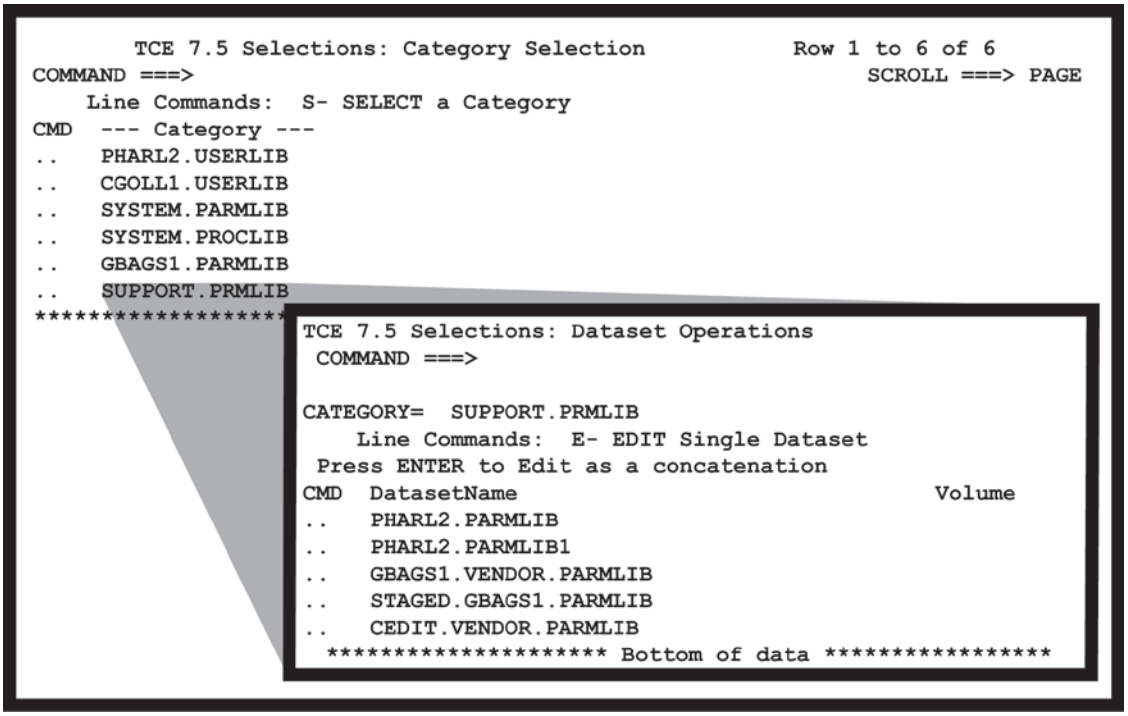
C Adding to your Protected Dataset List

How do we make sure that all of our important system datasets are being protected? TCE allows us to add site-specific datasets to its protected datasets list. This means that TCE can protect our system parameters no matter what our datasets naming standards are.

The Categories Selection screen (above) can be accessed from option D on the main TCE Admin Panel. Each of these categories contains a list of datasets that are to be protected by TCE. You can view the datasets protected within each category by selecting that category ('S') and pressing enter. The insert in the white box shows the datasets protected by the category SUPPORT.PRMLIB, and offers the chance to edit members of those datasets.

The NSECTL00 member in the hlq.PARMLIB library controls these lists of datasets. Full instructions on the column layout required by NSECTL00 member are

included at the top of the member, but the basic layout once into the definitions section is that columns 2 – 17 contain the Category name, and 18-61 the dataset name.



NewEra Software, Inc.

Contact Information
800-421-5035 (Toll-free in North America)
408-520-7100
support@newera.com
www.newera.com

User Guides and Product Data Sheets:
www.newera-info.com/Links.html