THE DECWINDOWS COOKBOOK

OR

I DON'T DO WINDOWS, I ONLY TUNE THEM...

By
John Wisniewski
Consultant
Digital Equipment Corporation

Introduction:

This guide is not designed as the definitive documentation to DECwindows functionality, performance and tuning. The ideas presented here should be considered a starting point for your tuning experimentations.

Many of the concepts presented will improve performance and functionality of the DECwindows software directly, others will have little value for your specific environment. The only true test of the success of any tuning action is if it improves the performance of your specific application.

To that end:

> Measure the performance.
>
> Adjust the System.
>
> Measure the performance again.
>
> If it works, keep the changes.
>
> If it doesn't, change it back.

Simple advice, but sometimes, the most experienced system tuner may become enamored with changing SYSGEN parameters and not really examining the application to see if it is truly performing better.

Between the KISS principle and Murphy's Law, tuning is more of an art than a Science. This document contains many brush strokes, no doubt many of them will help paint a better DECwindow picture for you and your users.

Many thanks to John Hildebrand, Brendan Eagan, Jack Smith, Ferne Hayashi, and Pat Ryan the many contributors to the DECwindows VAXnotes Files, and INTERNET NEWS feeds.. While more than a conduit, with this document, I'm much less than a source of all knowledge regarding DECwindows.

Many more windowing mysteries await, for MOTIF is soon to be with us...

Good luck, and Happy Windowing...

John Wisniewski
December 1990

Table of Contents

Chapter 1 DECWINDOWS TUNING

DECnet PIPELINE QUOTA

DECwindows default under VMS and from VT1300 Xwindow terminal is to use the DECnet protocol as transport mechanism for the X Window protocol.

(Note: the VT1000 and 1200 use LAT as an X transport!)

Using DECnet, the Xclient (computer) sends the graphic display information (using the X protocol) to the Xserver (Graphic display).

The Xclient to Xserver delivery needs adequate buffer space to support incoming and outgoing traffic or display/paint hesitation will be experianced.

In a standalone workstation, the traffic is more incoming with graphic displays forwarded to the local DECwindow display.

In a networked VMS Xclient, remote Xwindow access from other nodes may reverse the ratio to incoming to outgoing traffic.

DECnet uses the EXECUTOR PIPELINE QUOTA, adjusted under NCP to specific the number of bytes that may be used in nonpaged pool for transmission over all logical links.

This number of bytes may be used for multibuffering (incoming and outgoing DECnet traffic).

The command structure to change the DECnet pipeline quota is as follows

```
$MCR NCP
NCP> SET EXECUTIVE PIPELINE QUOTA x
NCP> DEFINE EXECUTIVE PIPELINE QUOTA x
NCP> EXIT
```

Be sure to perform both the SET and the DEFINE so the change is in both the volatile and permanent DECnet database.

DECnet PIPELINE QUOTA (cont)

Recommended DECnet Pipeline quota

VMS DEFAULTS FOR DECWINDOWS	= 10000 (bytes)
VMS STANDALONE WORKSTATIONS	= 25000 (bytes)
VMS REMOTE DECWINDOWS ACCESS WORKSTATIONS	= 25000 (bytes)
VMS MULTI SERVER CLIENT	= 32000 (bytes)

VMS multi server DECwindow clients (such as VAXes that have no DECwindow display) may service more than one DECwindow server. Enough DECnet Pipeline quota must be available to support the DECwindow sessions and the DECnet logical links that VMS establishes.

Don't be tempted to raise this number to the 65000 limit for PIPELINE quota. The Xwindow protocol will only use about 25000-32000 bytes and you'll be wasting valuable memory available in pool.

The net gain to your DECwindows is faster repaint times, less stuttering (partial paint - pause - finish) and less CPU overhead to transport DECwindow protocols to your DECwindows clients (both local and remote displays using DECnet as a transport).

SYSGEN PARAMETERS FOR DECWINDOWS

General rules for tuning DECwindow environments are all memory related.

An under-memory configured VAXstation will produce a sluggish, almost unusable user interface. If there is 8 to 16MB of memory on the VAX system, performance improves dramatically, but over time (if many processes are created and destroyed) memory may become allocated to processes that are dormant or worse not using all the memory that's been allocated to the process.

General memory reclamation can be encouraged by adjusting LONGWAIT and other related SYSGEN parameters, but under DECwindows V 1.0 (VMS 5.0) to V2.0 (VMS 5.4) there is no truly proactive memory reclamation activity for dormant processes.

By keeping the number of processes low, shutting down processes that are not being used, and by limiting the amount of memory consumed by a process, memory reclamation can be simulated at a useful level.

SYSGEN PARAMETERS FOR DECWINDOWS (cont)

The cost for this memory activity is trivial CPU overhead, moving pages from Window (process) working sets to the Memory Freelist.

The benefit is that having enough memory to accept new processes or grow existing ones, makes DECwindows run more responsive at a human level.

Future version of VMS may perform this function automatically, today we can simulated some functionality with a combination of SYSGEN and Authorize .

SYSGEN PARAMETERS FOR DECWINDOWS

The following is an example SYS$SYSTEM:MODPARAMS.DAT for a standalone VAXstation 3100.

You should take great care in tuning a system, and match the best tuning with the primary application. Only after the requirements of the primary application have been met, dole out the rest of the computer resources.

Please consider this as a STARTING POINT for your tuning activities, it is by no means to be considered as the ultimate DECwindows tuning mix.

General Rules:

Always use @SYS$UPDATE:AUTOGEN.COM to modify the system.

Tune for your primary application first.

Window activation may not be the best measure of performance.

The relevant SYSGEN parameters are noted and commented.

SYSGEN PARAMETERS FOR DECWINDOWS (cont)
SYS$SYSTEM:MODPARMS.DAT

```
!++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
! SYS$SYSDEVICE:[SYS0.SYSEXE]MODPARAMS.DAT
! EXAMPLE MODPARAMS.DAT VAXSTATION 3100 16MB of memory
!----------------------------------------------------------------------------
!
!-----------------MODPARAMS.DAT DECWINDOWS WORKSTATION
SCSSYSTEMID        =3213
SCSNODE            ="MYNODE"
VAXCLUSTER         =0    !is this a VAXcluster member 0=no 1=yes
!


!-----------------General VMS System parameters
DUMPSTYLE          =0    !Put system dump in the pagefile
SAVEDUMP           =1    !0= take a dump/1=don’t take a system dump
MIN_SYSMWCNT       =2400 !system workingset size
MIN_PAGEFILE       =60000!0= don’t change size/x=size of pagefile in blocks
SWAPFILE           =15000!0= don’t change size/x=size of swapfile in blocks
MIN_MAXPROCESSCNT  =50   !maximum process on the system
MIN_BALSETCNT      =52   !number of processes slots
MIN_WSMAX          =19000!Largest working set allowed
!QUANTUM=20              !Default VMS quantum for VAXstation
QUANTUM=14               !Best >5 window response
!QUANTUM=12              !Best 5-10 windows response
!QUANTUM=8               !Best 10+ window response (for each window)
WINDOW_SYSTEM=1       !VWS=0 DECwindows=1

!-----------------------MEMORY PARAMS
MIN_GBLPAGES=45000
MIN_GBLSECTIONS=400
MIN_GBLPAGFIL=20000
MIN_SRPCOUNT=1024     !Small request packets of memory
MIN_SRPCOUNTV=1000    !
MIN_IRPCOUNT=200      !Intermediate request packets of memory
MIN_IRPCOUNTV=250     !
MIN_LRPCOUNT=25       !Large request packets of memory
MIN_NPAGEDYN=800096   !nonpaged dynamic memory  note:this is for 16MB
MIN_PAGEDYN =804448   !paged dynamic memory     note:this is for 16MB

!-----------------------Make Working Set Adjustment more Agressive
!-----------------------This can have adverse effects on compute bound
!-----------------------Workstation jobs -- Measure the memory use.
PFRATH=80
PFRATL=1
LONGWAIT=10
WSINC=100
WSDEC=30
```

SYSGEN PARAMETERS FOR DECWINDOWS (cont)
SYS$SYSTEM:MODPARMS.DAT

```
!------------------------Process related values for DECwindows
MIN_VIRTUALPAGECNT=20000
MIN_MAXBUF=4096
MIN_PQL_DPGFLQUOTA=5000
!MIN_LONGWAIT=310
MIN_SWPOUTPGCNT=500
MIN_FREELIM=40
MIN_PQL_MASTLM=100
MIN_PQL_MBIOLM=40
MIN_PQL_MDIOLM=40
MIN_GBLPAGFIL=6024
MIN_PQL_MPRCLM=8
MIN_PQL_MFILLM=8
MIN_PQL_MBYTLM=32768
MIN_PQL_MWSQUOTA=60
MIN_PQL_MWSDEF=60
MIN_PQL_DWSEXTENT=500

!-----------------------RdB parameters
MIN_CHANNELCNT=256
MIN_LOCKIDTBL=325
MIN_RESHASHTBL=512
MIN_PQL_DBYTLM=22000
MIN_LOCKIDTBL_MAX = 2048

!-----------------------CDD/PLUS parameters
!MIN_NPAGEDYN=900096    !nonpaged dynamic memory
!MIN_PAGEDYN =904448    !paged dynamic memory
```

After SYS$SYSTEM:MODPARAMS.DAT has been created, you can invoke AUTOGEN to change the SYSGEN parameters and tune the workstation:

$@SYS$UPDATE:AUTOGEN GETDATA REBOOT NOFEEDBACK

This will tune and reboot the workstation.

USER ACCOUNT REQUIREMENTS

DECwindow user accounts should be adjusted to compliment the memory reclamation strategy that is created from the SYSGEN parameters set in the previous section.

WSDEFAULT should be set to a size large enough to create the window quickly, yet allow only enough memory to begin the process's image or task.

WSQUOTA should be large enough to contain the idle Process's image when it is idle and inactive. This will allow maximum restart performance for a window.

WSEXTENT should be set to allow for the total number of ACTIVE DECWINDOW PROCESSES to divide the rest of the memory up evenly among themselves.
If there is one main DECwindow application, make sure that it's process has all the memory required for it's best performance set here.

Note that this philosophy of memory management is different from the traditional time-share methods. We are not preallocating large slices of memory resources for each user to keep at all times, we are trying to give the maximum memory to latest application(s) a DECwindow's user activates and spreads the rest of memory resources among the remaining background processes and windows..

In a memory poor system, this method limits the overall performance, but heightens the active window process performance by reallocating memory only to active processes.

In a memory rich system, the memory scavenging may seem inappropriate, but idle jobs should have pages put back to the free list so that if other tasks are started, they too can have enough memory ready for quick activation.

The net gain is more memory for active or new jobs and the performance increase that only available memory can give.

In a workstation environment, we are only borrowing from ourselves, the resources should follow us where we are working and not remain trapped in idle processes or iconed windows.

USER ACCOUNT REQUIREMENTS (cont)

HOW TO DETERMINE IMAGE MEMORY REQUIREMENTS

The show process command will show peak and peak virtual size for a process:

```
$ show proc/acc
15-DEC-1990 19:53:02.72 User: SYSTEM                Process ID: 000003A3
                        Node: ROVER                 Process name: "_TWA48:"
Accounting information:
Buffered I/O count:   404       Peak working set size:       823
 Direct I/O count:      125     Peak virtual size:           5299
 Page faults:          1558     Mounted volumes:             0
 Images activated:     8
 Elapsed CPU time:     0 00:00:03.74
 Connect time:         0 06:38:50.50
```

After running the program within the current DECwindows. processes, the Peak working set size is a very good starting point for WSQUOTA and about 50-75% of peak virtual size is a good starting point for WSEXTENT.

Additional monitoring as the process performs real work will reveal the amount of memory actually required for WSEXTENT. The VAX Performance Advisor is an excellent tool for making this kind of determination as history is kept and correlated about an image’s average memory requirements automatically.

USER ACCOUNT REQUIREMENTS (cont)

EXAMPLE DECWINDOW USER ACCOUNT

An example user authorization file entry shows the memory, pagefile quota, and Bytlm that will make the DECwindow user with multiple task concurrently executing feel a much more responsive.

Make sure set Pgflquo: such as to leave about 5k-10k pages in the pagefile or your system may periodically hang.

```
$ RUN AUTHORIZE
UAF> SHOW DECWINDOW_USER1
Username: DECWINDOW_USER1              Owner: Win Dos
Account: WS122                           UIC: [124,4] ([BETTER,WINDOS])
CLI: DCL                               Tables: DCLTABLES
Default: NON_SYSTEM_DISK:[WINDOS]
LGICMD:
Flags: (Don't use Captive flag or DECwindows won't login!)
Primary days: Mon Tue Wed Thu Fri Secondary days:          Sat Sun
No access restrictions
Expiration:          (none)          Pwdminimum: 8   Login Fails:     0
Pwdlifetime:         (none)    Pwdchange:   22-SEP-1990 21:48
Last Login: 15-DEC-1990 11:02 (interactive), 15-DEC-1990 13:14 (non-interactive)
Maxjobs:       0   Fillm:          60    Bytlm:       100000
Maxacctjobs:  0 Shrfillm:          0    Pbytlm:           0
Maxdetach:   0   BIOlm:            36   JTquota:        1024
Prclm:        40   DIOlm:          36    WSdef:            512
Prio:          4   ASTlm:          96    WSquo:         2049
Queprio:       0   TQElm:          40  WSextent:       15000
CPU:   (none)    Enqlm:       4400    Pgflquo:        50000
Authorized Privileges:
TMPMBX NETMBX
Default Privileges:
TMPMBX NETMBX
   Identifier            Value              Attributes
 NICE_GUY            %X80010004
```

Note that these setting are for the user who is doing a variety of tasks, editing, compiling, and mail from their windows. Be sure to follow the tuning requirements for specific applications (like CAD/graphic intensive program).

Just because the mail window pops right up, that might not be the reason you or your company bought the machine. Tune the primary application first, then share the resources leftover with other DECwindow jobs!

Chapter 2 DECWINDOWS SYSTEM MANAGEMENT

DECwindows base applications are VMS processes, detached processes, and application programs, no different from traditional VMS managed entities.

The goal of DECwindows is to implement the Xwindow Client /Server graphic protocol. As a VMS application, DECwindows can be tuned and managed for better performance.

SYS$MANAGER:DECW$STARTUP

Starting DECwindows under VMS is accomplished by executing the command file

@SYS$MANAGER:DECW$STARTUP

This command is usually executed as part of system startup to set the DECwindows logicals, and startup the DECwindow server process.

Adding new fonts, or changing DECwindow logicals can be accomplished by using:

@SYS$MANAGER:DECW$STARTUP RESTART

This command shuts down the DECwindow processes and any window sessions that are logged in. Without rebooting VMS, DECwindows restarts and may then be used normally.   Hung DECwindow sessions may be freed with this command (if you have terminal access to the VAX) without rebooting the VMS system.

@SYS$MANAGER:DECW$STARTAPPS.COM

This command is executed from the DECW$STARTUP command invoked whenever SYS$MANAGER:DECW$STARTUP is started.

This file allows you to change logicals, create search lists for logicals and place new logicals in the (DECW$LOGICAL_NAMES) logical table. Changes to this file are unsupported because during upgrades, VMS replaces this file with a standard startup. Any special changes will have to be saved and re-edited into the file after any VMS upgrade.

DECWINDOWS SYSTEM MANAGEMENT (cont)

One of the limiting resource for the DECwindow processes is the workingset sizes and the process quotas for the DECW$SERVER_x and DECW$TE_x VMS processes.

The command file SYS$STARTUP:DECW$STARTAPPS.COM establishes the defaults for these process, and for most users these defaults are established very low. The defaults allow these processes to function on a very memory poor system, but for more memory available systems, increasing these values could greatly improve performance and eliminate DECwindow processes paging.

With background and new graphic intensive applications being charged against the workingset of the DECW$SERVER_x process (like the display postscript capablity of the CDA View in VMS 5.4) additional memory is one of the main improvements to performance we can give DECwindows.

Adjusting DEC$SERVER_x and DECW$TE_x processes are controlled from logical not documented in SYS$STARTUP:DECW$STARTUPAPPS.COM but can be added easily to this startup proceedure.

Logicals can be specified to define defaults for Workingset DEFAULT, QUOTA, and EXTENT can be as well as page file quota and enquelimits. These logicals must be specified before the body of DECW$STARTAPPS.COM or the processes created will not reflect their values.

DECW$BOOK LOGICAL

DECW$STARTAPPS.COM is the source of all DECW* default logical definitions. The DECW$BOOK logical is one of the more useful that is defined from this proceedure.

Creating a search list of DECW$BOOK enables the DECwindows Bookreader to read libraries from multiple locations.

Multiple bookreader Library files located in the search list appear appended to one another in order of the searchlist. This makes it appear as if all of the library files are one library to bookreader.

(The file DECW$BOOK:LIBRARY.DECW$BOOKSHELF controls what bookshelves and books are available from the Bookreader application)

DECWINDOWS SYSTEM MANAGEMENT (cont)

DECW$BOOK LOGICAL

This search list ability allows you to have several Libraries of books either on CDrom drives or Books created from DECwrite all available from one Bookreader session.

Using this technique, older versions of documentation could be kept on line and searched in order when looking for an appropriate book.

EXAMPLE SYS$STARTUP:DECW$STARTAPPS.COM

```
$ !
$ ! DECW$STARTAPPS.COM - Initialize the DECwindows user environment
$ !
$!***********************************************************************
$ !*
$ !*  COPYRIGHT (c) 1987, 1989 BY
$ !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
$ !*  ALL RIGHTS RESERVED.
$ !
$ ! This command procedure sets up the DECwindows user and application
$ ! development environment.  It should be run at system startup time.
$ ! It is not advisable to edit this procedure as it may be replaced in
$ ! future software updates.  Instead, you should perform any site-specific
$ ! commands in your SYSTARTUP file.
$ !
$ ! Define the appropriate logical names.
                         ...

$!---------------------------Logicals for addition to DW defaults
$! decw$define decw$term_wsdefault_ctrl_wsdef    2048
$! decw$term_wsdefault_ctrl_wsdef doesn't seem to work!
$ decw$define decw$term_wsdefault_ctrl_wsquota  4096
$ decw$define decw$term_wsdefault_ctrl_wsextent 'F$GETSYI("WSMAX")
$!
$ decw$define decw$server_wsdefault  2048
$ decw$define decw$server_wsquota    8182
$ decw$define decw$server_wsextent   'F$GETSYI("WSMAX")
$ decw$define decw$server_page_file  40000
$! decw$define decw$server_enqueue_limit
$ decw$define decw$book sys$common:[decw$book],dkb400:[decw$book]
$!DECW$BOOK search list looks in two locations!
$!---------------------------Logicals for addition to DW defaults
$ decw$define decw$server_main_kb_updown True
$ decw$define vue$library    decw$syscommon:[vue$library.user],-
decw$syscommon:[vue$library.system]
$ decw$define decw$winmgrexe sys$system:decw$winmgr.exe

                        ...
```

DECWINDOWS SYSTEM MANAGEMENT (cont)

SYS$MANAGER:DECW$SYLOGIN.COM

This command file will execute startup commands when any user of a VAXstation logs in through DECwindows.

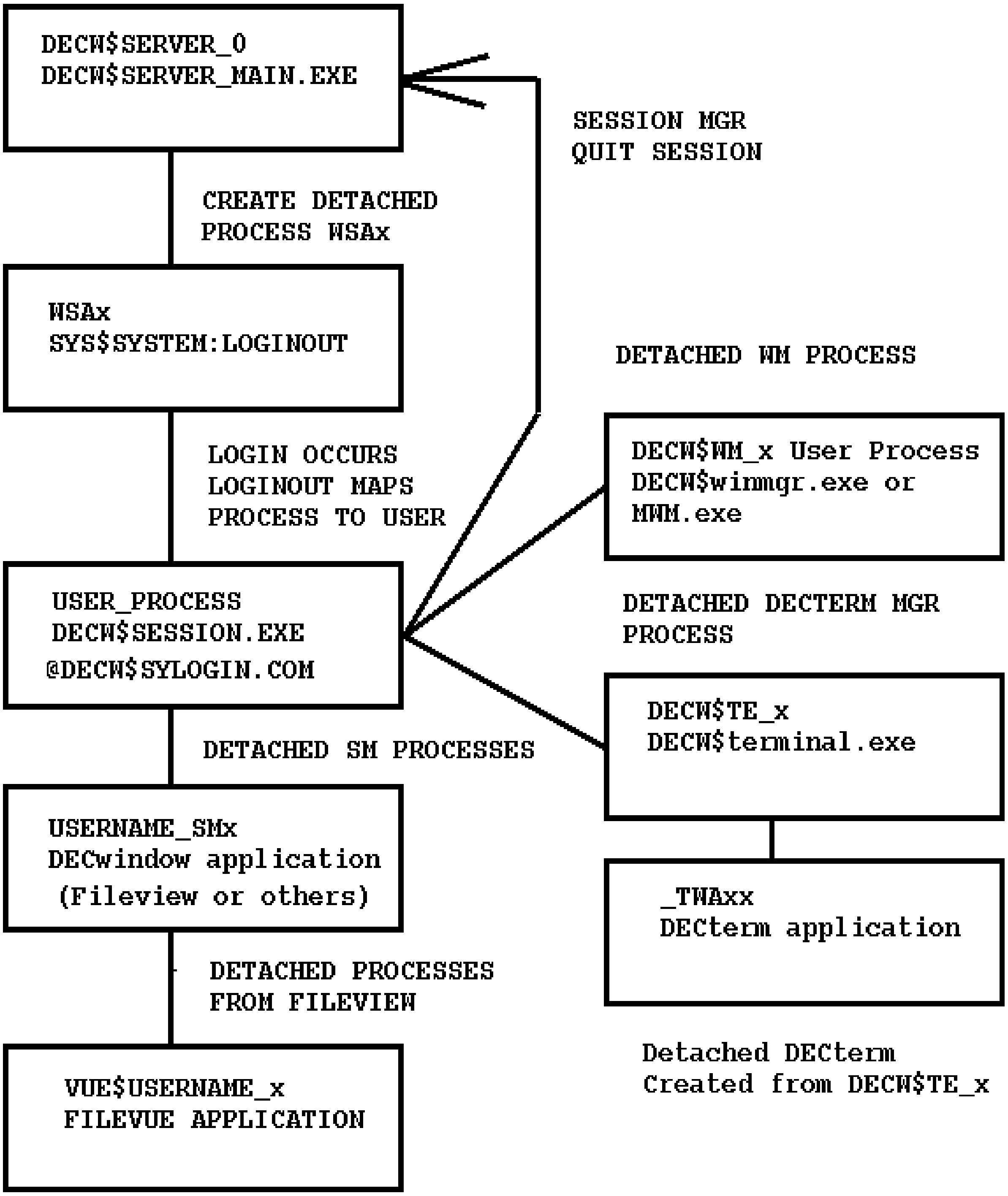
This is useful in starting DECwindow application programs that need to run as part of a general login. (The primary workstation applications may be started up though this file)

Do not put interactive requests ($INQUIRE, $READ, etc.) in this file as during login, DECwindow processes have SYS$INPUT, SYS$OUTPUT mapped to temporary mailboxes.

Places for interactive login information is still SYS$MANAGER:SYLOGIN.COM and the SYS$LOGIN:LOGIN.COM.

(Note except for a DECTERM or another Xterminal session, these files will not be executed. Extreme care must be observed in creating a restricted environment.)

DECWINDOWS PROCESS FLOWCHART



DECWINDOWS SYSTEM MANAGEMENT (cont)

PROCESS NAMES / DESCRIPTIONS

From a $SHOW SYSTEM display, identifying different DECwindow processes is somewhat confusing.

```
$SHOW SYSTEM
VAX/VMS V5.4  on node ROVER  15-DEC-1990 23:52:21.59   Uptime  4 01:16:43
  Pid    Process Name    State  Pri      I/O       CPU       Page flts Ph.Mem
...
00000382 USERNAME        LEF      7       47   0 00:00:27.50      1644   1567
00000383 DECW$WM_2       LEF      7       47   0 00:00:27.50      1644   1567
00000384 USERNAME_SM1    LEF      4      148   0 00:00:53.41      2862   2158
00000398 USERNAME_SM3    LEF      6      321   0 00:00:05.27      2555   1870
00000399 DECW$TE_2       LEF      7    58612   0 00:10:16.96      5118   3030
0000039A _TWA47:         LEF      4   192777   0 00:03:53.77     24567    406
000003AC VUE$SYSTEM_1    LEF      4    16681   0 00:19:22.04     10159   4965
000000C6 DECW$SERVER_0   HIB      8    19171   0 00:11:14.75     31424   3985
...
```

USERNAME -- This process running DECW$SESSION is the orgin of all other DECwindow processes.

USER_SM1 -- These names refer to DECwindow jobs spawned or created
2 from the Logged in user session manager window.

DECW$WM_x -- The Window Manager process that controls the movement and resizing of DECwindow Windows. If this process is terminated, windows while functional cannot be moved manipulated or changed in size or occlusion.

DECW$TE_x -- This process is a parent process for all Session Manager DECterms created. Terminating this process terminates all DECterms (_TWAx: processes) created from the Session manager.

VUE$SYSTEM_x -- This process is FILEVIEW CREATED PROCESS and other processes beginning with that name refer to detached processes created from Filevue.
The primary FILEVIEW process will be of the form USER_SMx PROCESSES on the system.

DECW$SERVER_x -- This process is the primary DECwindows server for all all windows and the Window Manager. Terminating this process stops DECwindows (not recommended) and logs out current DECwindow sessions.

DECWINDOWS SYSTEM MANAGEMENT (cont)

SYS$SYSROOT:[DECW$DEFAULTS]

This directory contains the DECwindow default files. The *.DAT files control various parameters for the DECwindow environments for many DECwindow applications.

Directory SYS$COMMON:[DECW$DEFAULTS.SYSTEM]
DECW$BOOKREADER.DAT;1
DECW$CALC.DAT;1
DECW$CALENDAR.DAT;1
DECW$CARDFILER.DAT;1
DECW$CLOCK.DAT;1
DECW$LOGIN.DAT;1
DECW$MAIL.DAT;1
DECW$NOTEPAD.DAT;1
DECW$PAINT.DAT;1
DECW$PUZZLE.DAT;1
DECW$SESSION.DAT;1
DECW$SM_BW.DAT;1
DECW$SM_COLOR.DAT;1
DECW$SM_GENERAL.DAT;1
DECW$SM_GRAY.DAT;1
DECW$TERMINAL.DAT;1
DECW$WINMGR.DAT;1

DECWINDOWS SYSTEM MANAGEMENT (cont)

SYS$SYSROOT:[DECW$DEFAULTS]

The important parameters are located in DECW$SM_GENERAL.DAT;1. This Session Manager general file controls such things as default pointer character, window focus, and many useful customizations. Any changes made to this file will become the default for all the VAXstation users.

Creating a DECW$*.DAT file in the user's login directory, overrides any defaults with the values specified within the file. Additions to the user file may only be the parameters that user wants. All other parameters will be derived from the system files.

FILEVIEW / SYS$SYSROOT:[VUE$LIBRARY]

FILEVIEW is DIGITAL's point and click application for running programs, manipulating files and navigating directory structures under DECwindows. FILEVIEW verbs are defined from a pulldown menu, but some programs need to have specific parameters, and values passed to them as the are activated.

The solution to this is the SYS$SYSROOT:[VUE$LIBRARY] in this directory are *.COM files that activate images with the appropriate parameters.

This location of files control STATUS WINDOWS, Pop up displays and many other FILEVIEW activation parameters. User procedures may be placed in this location as well for use by the entire system.

DISABLING OPCOM MESSAGES FROM DECWINDOWS

Background OPCOM messages may be disabled by the following

```
 $ Define/User Sys$Command OPA0:
 $ Reply/Disable
```

Or in the SYS$MANAGER:SYLOGICAL.COM file,

```
$ DEFINE/SYSTEM OPC$OPA0_ENABLE FALSE
```

Either method will prevent operator messages from going to OPA0: and eliminate the interruption of DECwindows graphics. All messages will still be logged to the operator file and active operators.

Chapter 3 DECWINDOW SYSTEM CONTROLS

DECW$LOGIN_BACKGROUND logical

The DIGITAL Logo that is on the login window is the default for all DECwindow systems. This default can be overridden by defining a System Executive logical name:

```
$DEFINE/SYSTEM/EXEC DECW$LOGIN_BACKGROUND SYS$MANAGER:DECW$TITLE.COM
```

After this logical is defined and the DECW$TITLE.COM file is CREATED, the DIGITAL logo will no longer appear.

Whatever commands are in DECW$TITLE.COM will execute and continue executing until the command procedure is exited.

The usefulness of this functionality may be only to put alternate information on the login DECWINDOWS screens, or it may be used to construct an entire CAPTIVE DECWINDOWS environment.

Any valid DECwindows program can be executed from DECW$TITLE.COM and information or even the window manager can be placed on the root window and created within child windows.

The main security risk to a captive DECWINDOW login window is that if any of the applications can spawn or execute DCL commands, they execute as the system account (The process which is running DECW$TITLE.COM in this example is DECW$SERVER_x which is always owned by the system.).!

Care must be taken to prevent this from occurring but it can be done either by a program or cleverly constructed DCL proceedure.

Know the application you are starting in the login window, spawn only captive protected DCL COMMAND PROCEDURES that change the UIC of the executing programs, or if security is of top priority:

Don’t create a CAPTIVE LOGIN environment for DECwindows.

This is a workaround for CAPTIVE LOGINS, and it makes you responsible for SECURITY!

DECW$LOGIN_BACKGROUND logical (cont)

```
$!++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
$!                    SYS$MANAGER:DECW$TITLE.COM
$!                EXAMPLE file for DECWINDOWS LOGIN SCREEN
$!     ONE DECTERM, ONE BOOKREADER, WINDOW MANAGER, CAPTIVE LOGIN READY
$!    LIMITATION:  Cycle not complete, no mechanism for terminating
$!    without a session manager.
$!                 DECW$WINMGR.EXE PROCESS.
$!                    EXAMPLE ONLY.....
$!----------------------------------------------------------------------------
$set cont
$set noon
$10:
$set proc/prio=4        !set proc/prio=interactive level
$!
$!Create DECTERM which will have to have a USER/PASSWORD entered.
$!this is nessisary because if you don't log in and just do the
$!create from here, you'll have a DECTERM logged in as SYSTEM!
$!
$create/terminal
/noprocess/display=rover::0
/WINDOW_ATTRIBUTES=(INITIAL_STATE=ICON,ICON_NAME="CAPTIVE LOGINWINDOW",-
 TITLE="CAPTIVE LOGINWINDOW")
$!
$!RUN any specific application as a spawned subprocess
$!nowait returns control to this proceedure.
$!
$SPAWN/NOWAIT/NOLOG run sys$system:decw$bookreader.exe
$!
$!Running the DECW$WINMGR creates a session box and allows the placement
$!windows located on the LOGIN (ROOT) WINDOW!
$!
$run sys$system:decw$winmgr.exe
$!
$!IF WE EXIT HERE, ALL THE RUNNING PROGRAMS START.  IF DECW$WINMGR IS
$!TERMINATED, CYCLE THROUGH AND START AGAIN...
$goto 10:
$exit
```

Note that if we cycle out of this proceedure, the DECW$SERVER_x process will not attempt to run the command proceedure again. It will only be run by logging out of a DECwindows Session or restarting the server process.

DECWINDOW FONT CONTROLS

SYS$MANAGER:DECW$PRIVATE_SERVER_SETUP.COM .TEMPLATE

This template is an example of how the 75dpi and the 100 dpi fonts are selected for your DECwindow session. You can modify the template file and save it as a *.COM file. Then you have to reboot or restart DECwindows to install the new fonts.

The font name can be wildcarded. The server will pick the first font name in its list that satisfies the wildcard match. Normally you should specify enough fields to keep the search small. For example:

-*-Terminal-Medium-R-Normal-*-140-*-*-*-*-ISO8859-1

Tools like xlsfonts.exe (from the MIT Xwindow distribution) can quickly display all the fonts available on your system. Another way uses to view the different fonts is to use the standard DECwindow application, NOTEPAD.

USING THE NOTEPAD APPLICATION TO CHANGE FONTS:

* Select a font with NOTEPAD customize menu
* Change NOTEPAD FONT
* Save NOTEPAD CUSTOMIZATIONS
* Type SYS$LOGIN: DECW$NOTEPAD.DAT
* Cut font type into command
  or create a logical FONT-NAME and use it:

* $CREATE/TERM/WINDOW=(FONT=" font-name")/LITTLE_FONT
  (There is black magic in the format of this command,
  the /LITTLE_FONT qualifier is require after the /WINDOW)

* DECterm window is created with the chosen font.

If you like the DECterm that results, do a "save current settings" from the DECterm customize menu and it will be permanently saved in your SYS$LOGIN:DECW$TERMINAL.DAT.

FONT DENSITY ON DIFFERENT MONITORS

If you use the 100 dpi fonts on a 75 dpi monitor, everything looks bigger. That's the easiest way to increase the size of everything

DECWINDOW POINTER CONTROL

DECwindow pointers may be controlled from the SESSION MANAGER'S CUSTOMIZE menu or may be controlled by the file:

SYS$LOGIN:DECW$SM_GENERAL.DAT

This file may be edited/created and with the addition of the line:

sm.pointer_shape: number

When any number other than 0 will index into the standard MIT X CURSOR FONTS.

These 76 pointer shapes are programatically documented in the file

SYS$LIBRARY:DECW$CURSOR.H

but several of the more useful number are documented here:

```
finger_up                      2
waiting_watch                 -4
no_entry_sign                 -6
open_square                    8
boat                          -8
star                          +20
square_labyrinth              -20
white_donut                   -24
XC_coffee_mug                 -28
junction                      -34
circle_white                  -38
arrow_up_down                 -42
hand_back_north               +42
hand_back_north_east          +44
finger_pointing_west          +45
circle_arrows                 -50
a_bird                        -54
hand_pointing_left_1          -55
hand_pointing_left_2          -60
little_man                    -80
skull_&_crossbones            -88
rectangle_mouse               -100
yet_another_square            -102
little_ship                   -104
various_arrows                -105 up to -110
XC_spraycan                   -124
starship_enterprise_top       -142
umbrella_4_white_backgr       -146
chimney_4_black_backgr        -147
wrist_watch                   -150
```

DECWINDOW POINTER CONTROL (cont)

In order to install the new pointer shapes, after editing the DECW$SM_GENERAL.DAT file, log out and log back in.

A slightly more complex way (but preferable to logging out) is to edit the DECW$SM_GENERAL.DAT file and use the SESSION MANAGER'S CUSTOMIZE WINDOW to USE LAST SAVED SETTINGS.
This will reset the cursor to the new value.

DECWINDOW TERMINAL/DEVICE TWAx: (MESSAGE BOX)

The Message box is the lowest number TWA device on the system. Information can be REPLYed, TYPED, or /OUTPUT= to this device as if it were a terminal (assuming you had write access to that terminal!)

Very useful location if status messages are needed from jobs running in background.

The device number is an incrementing number, adding to itself every login (similar to the LTA device incrementation). The only easy way to identify the message window is to do a $SHOW DEV TW and identify the lowest device number.

DECWINDOW WINDOW FOCUS ON ACTIVATION

In DECW$SM_GENERAL.DAT the resource control line

```
wm*default.startupfocus:  true           !sets focus to new/latest window
wm*default.deiconifyfocus:  true         !sets focus to deiconified window
```

This is clearly a violation of the XWINDOWs Style guide, but it keeps focus not on the last window a DECwindow user was working on, but on the latest window created or deiconified.

This also saves an extra mouse click to focus specifically on a newly created window.

Chapter 4 DECTERM CONTROLS

DECTERMS DETACHED PROCESSES

SYS$INPUT/SYS$OUTPUT ARE MAILBOXES UNTIL LOGIN

DECwindow applications and DECterms are created first as detached processes with SYS$INPUT and SYS$OUTPUT defined as mailboxes.

When they are created, they read and execute SYSLOGIN.COM and LOGIN.COM After process creation, the process redefines SYS$INPUT and SYS$OUTPUT to be the terminal/device it's attached to.

After the redefinition of the logicals, the process executes both SYSLOGIN.COM AND LOGIN.COM again!

This make any interactive use of SYS$LOGIN.COM and/or LOGIN.COM fail the procedures because $INQUIRE or $READ commands will not work against the mailbox definitions.

Executing the SYS$LOGIN.COM and the LOGIN.COM twice is also quite expensive and time consuming when waiting for a window to pop up.

One way to correct the double execution (if desired) is to check to see that the device is a terminal or a mailbox. A single IF statement can handle the check in both files:

```
$ IF F$GETDVI("SYS$OUTPUT", "TRM") THEN GOTO START_TERMINAL
$ GOTO EXIT
$START_TERMINAL:

...        BODY OF SYS$LOGIN.COM OR LOGIN.COM

$EXIT:
$EXIT
```

This check will save time in getting your DECterms created and logged in by only executing the login file when they are connected to real terminals.

This also allows for your interactive connections to work as well if they are partitioned by the check in the DCL proceedure.

DECterm ESC/OSC control sequences

DECterm windows utilize a DIGITAL supplied terminal emulator that emulates many of the Character Cell Terminals built by Digital.

Some escape sequences can control various characteristics of the terminal window.

For Example:

```
$ OSC[0,8] = 157
$ ST[0,8] = 156
$ write sys$output OSC,"2L;New Icon Name",ST
$ write sys$output OSC,"21;New Window Bar Name",ST
```

This procedure quickly changes the Window Banner and Icon Name. Changes can come from a program or from a DCL command proceedure. This approach allows for status information to be quickly displayed in an Icon without having to deiconify the window and examine it fully.