Scripts shell d'administration Unix

1 1 Introduction et sommaire du document

Tous les scripts shell, dont on donne ici le texte dans ce document, sont des outils d'administration Unix.
Ce sont des outils utiles pour les développement et la mise au point d'applications informatiques et pour l'administration UNIX.

Sommaire :

n°	nom script	Rôle du programme (script shell)
1.	**cnvdosux**	conversion de fichiers DOS -> UNIX
2.	**cnvuxdos**	conversion de fichiers UNIX UNIX
3.	**compc**	compilation d'un source "c"
4.	**contxsch**	recherche d'une chaine et les lignes qui l'entourent dans un groupe de fichiers (très utile)
5.	**dir**	liste tous les sous-répertoires d'un répertoire (très utile)
6.	**dtree**	liste tous les sous-répertoires d'un répertoire sous forme d'une liste arborescente. (affiche l'arborescence des répertoires situés sous un répertoire donné) (très utile)
7.	**tree**	liste tous les sous-répertoires et les fichiers associées d'un répertoire de façon "identée"

8.	**env_reg**	envoi par uucp d'un fichier à une liste de site destinataires
9.	**head**	affiche les premières lignes d'un fichier ASCII (si **head** n'existe pas sous cet Unix)
10.	**kill_uucp**	suppression d'envoi(s) "uucp" par un utilisateur
11.	**ll**	liste de toutes les caractéristiques de tous les fichiers (y compris les fichiers cachés) d'un répertoire (très utile)
12.	**pattsch**	recherche d'une chaine de caractère dans une arborecence de répertoire (très utile)
13.	**where**	recherche simple d'un fichier a partir d'un répertoire. permet de rechercher un fichier quelconque dans tous les répertoire de la machine (selon autorisations d'accès). Commande très utile, elle permet d'éviter de retaper toute la commande "find ..." dont la syntaxe n'est pas très conviviale.
14	**init_ksh**	initialisation de l'environnement de ksh
15.	**pr2**	impression formattée avec date nom fichier imprimé etc ...
Etc	**Etc**	etc

Voici le texte des scripts d'administration listés dans le sommaire ci-avant.

2 2 Programme de conversion de fichiers DOS -> UNIX : cnvdosux

```
# ------------------------------------------------------------------------
   ---
# Nom du script : cnvdosux   Auteur : B. LISAN  Date : 10/3/95
# But : Convertir
#     . un fichier ascii sequentiel au format DOS (fichier dont les
#       lignes se terminent par les caracteres \r\n
#       (retour chariot, nouvelle ligne) ou \015\012 ou ^M^J), en
#     . un fichier au format UNIX (fichier dont les lignes se terminent par
   le
#       caractere \n (nouvelle ligne ou le caratere qui correspond en ascii
   a
#       octal \012 ou ^J)
# parametre d'entree : $1 = nom du fichier DOS a convertir
# parametre de sortie: $2 = nom du fichier UNIX resultat de la conversion,
   si
#                       $2 vide alors $2=$1, on convertit le fichier en
   entree.
# ------------------------------------------------------------------------
   ---
CR=`echo "\015"` # le retour-chariot ou carriage-return = code octal ascii
   015
if [ "($1)" = "()" ] && [ "($2)" = "()" ]
then
   echo "aucun nom de fichier a convertir precise"
   exit
fi

if test -z "$2"
then
   fic=$1
else
   fic=$2
fi

sed -e "1,\$s/$CR\$//" $1 > $fic

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

3 3 Programme de conversion de fichiers UNIX -> DOS : cnvuxdos

```
# ------------------------------------------------------------------------
   ---
# Nom du script : cnvuxdos   Auteur : B. LISAN  Date : 10/3/95
# But : Convertir
#     . un fichier au format UNIX (fichier dont les lignes se terminent par
   le
#       caractere \n (nouvelle ligne ou le caratere qui correspond en ascii
   a
#       octal \012 ou ^J)
#     . en un fichier ascii sequentiel au format DOS (fichier dont les
#       lignes se terminent par les caracteres \r\n
#       (retour chariot, nouvelle ligne) ou \015\012 ou ^M^J),
# parametre d'entree : $1 = nom du fichier UNIX a convertir
# parametre de sortie: $2 = nom du fichier DOS resultat de la conversion,
   si
```

```
#                       $2 vide alors $2=$1, on convertit le fichier en
   entree.
# ------------------------------------------------------------------------
   ---

CR=`echo "\015"` # le retour-chariot ou carriage-return = code octal ascii
   015
TAB=`echo "\t"`
if [ "($1)" = "()" ] && [ "($2)" = "()" ]
then
   echo "aucun nom de fichier a convertir indique"
   exit
fi

if test -z "$2"
then
   fic=$1
else
   fic=$2
   cp $1 $fic
fi

cat $1 | sed -e "1,\$s/.*\$/$CR/" > $$
paste $1 $$ | sed -e "1,\$s/$TAB$CR\$/$CR/" > $fic
rm $$

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

4 4 Programme de compilation d'un source "c" : compc

```
#------------------------------------------------------------------------
   -----
# procedure shell-UNIX : compc
# procedure de compilation simple d'un programme source c
#------------------------------------------------------------------------
   -----
#
if [ $# = 1 ]
  then
   cc -g -I/usr/hsource $1.c  -o $1
fi
if [ $# = 2 ]
  then
   if [ "$2" = "c" ]
      then
        cc -g -I/usr/hsource $1.c  -o $1 -lcurses
      fi
fi

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

5 5 Programme de recherche d'une chaine et des lignes qui l'entourent dans un groupe de fichiers : contxsch

```
#------------------------------------------------------------------------
   ---
# @(#) contxsch  : pattern context search in files (belonging of a same
#                  directory)
# Nom / Name     : contxsch
# Type / Type    : job shell (UNIX)
```

```
# But / Aim      : recherche d'une chaine dans tous les fichiers precises
   en
#                  parametre de contxsch en affichant le la ligne contenant
   la
#                  chaine ainsi que la ligne situee avant et la ligne situe
#                  apres (ces groupes de 3 lignes etant separees par une
   ligne
#                  comportant des points).
# Auteur / Author: Benjamin LISAN  ->    date creation:  01-AUG-90
# Usage / Usage  : "contxsch pattern filename [filename]"
# Exemple / Ex.  : "contxsch 3B2 *"
#-------------------------------------------------------------------------
   ---
USAGE="Usage= $0 pattern filename [filename]"
case $# in
    0|1) echo $USAGE # > &2
         exit 1 ;;
esac

pattern=$1 # stocke la chaine avant decalage
           # save pattern before shift
shift      # suppression argument contenant la chaine
           # suppression argument containing pattern

for i in $*                # boucle a travers tous les noms de fichier
do                         # loop threw all the files names

     case $# in
              1) name="";;       # pas de nom de fichier stocke
                                 # no file name saved
              *) name="$i:  ";;  # stockage du nom de fichier
                                 # save of the file name
     esac ;\
     sed -n -e "/$pattern/{
                           x
                           s!\(.*\)!$name\1!p
                           x
                           s!\(.*\)!$name\1!p
                           n
                           s!\(.*\)!$name\1!p
                           n
                           s!\(.*\)!$name\1!p
                           n
                           s!\(.*\)!$name\1!p
                           a\\
                           ...
                          }" -e h $i
done

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

6 6 Programme liste tous les sous-répertoires d'un répertoire : dir

```
ls -p $1 $2 | grep /

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

7 7 Programme liste tous les sous-répertoires d'un répertoire sous forme d'une liste arborescente : dtree

```
#------------------------------------------------------------------------
----
# @(#) dtree        : Visual display of directory tree
# module name / nom :dtree
# module type / type: procedure script shell (UNIX)
# But / Aim         : Affiche a l'ecran l'arborescence des repertoires et
des
#                     sous-repertoires sous la forme d'un arbre
# Auteur/Author     :  ?                date creation / creation date: 07-
SEP-90
#
# Usage / Usage     : dtree [repertoire]
#                     dtree [starting-directory-name]
#------------------------------------------------------------------------
----
#
USAGE="Usage: $0 [starting-directory-name]"
case $# in
   0) startdir=".";; # Use current directory
   1) if [ ! -d $1 ]; then
         echo "\"$1\" not a directory." >&2
         echo $USAGE >&2
         exit 1
      fi
      startdir=$1 ;;
   *) echo $USAGE >&2
      exit 2 ;;
esac
 (cd $startdir; pwd)
find $startdir -type d -print 2> /dev/null | sort -f |
sed -e "s,^$startdir,," \
-e "/^$/d" \
-e "s,[^/]*/\([^/]*\)$,\`-----\1," \
-e "s,[^/]*/,|      ,g"


#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

8 8 Programme liste tous les sous-répertoires et les fichiers associées d'un répertoire de façon "identée" : tree

```
for i in `ls `
    do
        if test -d $i
           then
                echo "-$i"
           else
                echo "    $i"
        fi
    done


#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

9 9 Programme d'envoi par uucp d'un fichier à une liste de site destinataires : env_sites

```
# ------------------------------------------------------------------------ #
# Projet        : PROJ..             Sous-Projet  : ADMIN.                 #
# Auteur        : B. LISAN           Date creation: 29/03/95  17:..        #
# Nom programme : env_sites          Type Langage : Bourne SHELL           #
# Objet du trait: envoi d'un fichier Version      : ......                 #
#                 a une liste de sites                                     #
# Commentaire   : Il vaut mieux etre 'root' pour executer cette commande   #
#                                                                          #
# Fic.tmp.crees : $$.tmp env$$       type acces   : seq                    #
# ------------------------------------------------------------------------ #
# Modifications : Auteur : .............     Date : jj/mm/aa hh:mn         #
# Reference dem.:                                                          #
# But:                                                                     #
# desc:                                                                    #
# ------------------------------------------------------------------------ #
tput rev
echo "*--------------------------------------------------------------------*"
echo "| Debut du programme $0                                              |"
echo "| envoi d'un fichier a une liste de sites en regions (depots, DR) |"
echo "*--------------------------------------------------------------------*"
tput sgr0
#tput rmt0

# initialisations
# ---------------

TAB=`echo "\t"`
site_ori=`uname`
prg=`basename $0`
site_dev="G60DVP"
fic_adr="/usr/lib/uucp/Systems"
#fic_adr="/etc/x25hosts"

# verification du login 'root'
# ----------------------------

lognam=`id | cut -d'(' -f2 | cut -d')' -f1`
if [ "$lognam"  != "root" ]
then
   echo ""
   tput rev
   echo "$prg> vous n'est pas 'root': \c"
   tput smul
   echo "$lognam\c"
   tput sgr0
   tput rev
   echo " -> vos envois peuvent ne pas reussir"
   tput sgr0
   #exit
fi

# saisie de la liste des sites destinataires
# ------------------------------------------

echo ""
echo "$prg> 0) le fichier qui contient la liste des sites destinataires est
   :"
echo "$prg>    $fic_adr, si celui-ci vous convient taper Entree,"
```

```
echo "$prg>    sinon taper le nom du fichier les contenant: \c"
read repons
if [ ! -z "$repons" ]
then
    fic_adr="$repons"
fi

# saisie du nom du fichier a envoyer et du repertoire de destination
# ------------------------------------------------------------------

fic_env=""
while [ -z "$fic_env"  ]
do
   echo ""
   echo "$prg> 1) Donner le nom et le chemin du fichier que vous voulez
   envoyer,
"
   echo "$prg>    (exemple: /usr/lib/uucp/Permissions) "
   echo "$prg> : \c"
   read fic_env
done

if [ ! -f "$fic_env" ]
then
   echo "$prg> Ce fichier n'existe pas -> sortie du programme"
   exit
fi

rep_dest=""
while [ -z "$rep_dest"  ]
do
   echo ""
   echo "$prg> 2) Donner le nom et le chemin du repertoire de destination,"
   echo "$prg>    (exemple: /usr/lib/uucp/) "
   echo "$prg> : \c"
   read rep_dest
done
if [ ! -d "$rep_dest" ]
then
   echo "$prg> Ce repertoire de destination n'existe pas -> sortie du
   programme"
   exit
fi

# elimination des lignes de commentaires
# remplacement des tabulations par des blancs
# et on ne garde que la 1ere colonne des noms-alias.
# elimination du site emetteur du fichier dans la liste des sites
   destinataires
# ainsi que   le site de developpement.

cat $fic_adr                    \
| sed -e "1,\$s/#.*//g"         \
      -e "1,\$s/$TAB/ /g"       \
      -e "1,\$s/$site_ori//g"   \
      -e "1,\$s/$site_dev//g"   \
      -e "1,\$s/  / /g"         \
| cut -d" " -f1   >             /tmp/$$.tmp

# elimination des lignes blanches
```

```
> /tmp/env$$
for site in `cat /tmp/$$.tmp`
do
   echo "/usr/bin/uucp $fic_env $site!$rep_dest" >> /tmp/env$$
   echo "/usr/bin/uustat" >> /tmp/env$$
done

echo ""
echo "$prg> 3) Verification du contenu du script d'envoi du fichier "
echo "$prg>    $fic_env"
echo "$prg>    en region, taper une touche -> \c"
read repons

vi        /tmp/env$$
chmod 700 /tmp/env$$

# execution des envois

echo ""
echo "$prg> 4) voulez vous envoyer le fichier vers ses destinataires?"
echo "$prg>    si oui taper 'OUI', sinon taper une touche : \c"
read repons
if [ "$repons" = "OUI" ]
then
   # su -a uucp -c "sh -x /tmp/env$$"
   sh -x /tmp/env$$
   /usr/bin/uustat
else
   echo "$prg> Pas d'envoi du fichier vers ses destinataires."
fi

rm /tmp/$$.tmp /tmp/env$$
echo "$prg> Fin du programme $prg"
```

```
#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>
```

10 10 Programme de visualisations des premières lignes d'un fichier : head

Note : normalement toutes les interprêteurs de commandes shells des systèmes UNIX actuels possèdent cette commande **head**. Mais ce script peut simuler cette commande.

```
sed -n 1,$1p  $2  >$3

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>
```

11 11 Programme de suppression d'envoi(s) "uucp" par un utilisateur : kill_uucp

Note : l'envoi par le protocole X25 (sur par exemples des lignes Transfix), par **uucp**, devient rare.

```
# ---------------------------------------------------------------------- #
# Projet         : PROJET             Sous-Projet  : ADMIN.              #
# Auteur         : B. LISAN           Date creation: 29/03/95  17:..     #
# Nom programme  : kill_uucp          Type Langage : Bourne SHELL        #
# Objet du trait: tuer la liste de demande de transfert pour uucp faite  #
#                 par l'utilisateur qui a lance ces uucp et qui utilise  #
#                 cet outil 'kill_uucp'                                  #
#                                                                        #
# Fic.tmp.crees : /tmp/kiluu$$        type acces   : seq                 #
# ---------------------------------------------------------------------- #
# Modifications : Auteur : ............     Date : jj/mm/aa hh:mn        #
# Reference dem.:                                                        #
# But:                                                                   #
# desc:                                                                  #
# ---------------------------------------------------------------------- #
tput rev
echo "*--------------------------------*"
echo "| Debut du programme $0   |"
echo "| suppression de vos envois uucp |"
echo "*--------------------------------*"
tput sgr0
#tput rmt0

# initialisations
# ---------------

prg=`basename $0`
uustat | cut -d" " -f1 | sed -e "1,\$s/^/uustat -k/" > /tmp/kiluu$$
echo "$prg> Verification script de suppresion des envois"
read repons

vi /tmp/kiluu$$
sh /tmp/kiluu$$
rm /tmp/kiluu$$
echo "$prg> fin du programme"

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>
```

12 12 Programme listant toute les caractéristiques de tous les fichiers (y compris les fichiers cachés) d'un répertoire : ll

Note : ll : une commande simple et utile :

```
ls -al $* | pg
#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>
```

13 13 Programme recherchant une chaine de caractère dans une arborecence de répertoire : pattsch

```
#---------------------------------------------------------------------------
   -----
# @(#) pattsch  : pattern search in directories tree
# nom du module : pattsch                                       Version:
   V1.0.0
# type de module: procedure shell-UNIX
# But           : cherche une chaine donnee dans les fichiers situes sous
   un
#                 repertoire donnee y compris ceux de repertoires situes
   sous
#                 ce premier repertoire
# Auteur        : Bejamin LISAN
#
# Usage         : pattsch chaine    [nom_rep]
#
# Arg.INPUT:  chaine      -> Desc: chaine a rechercher
#                            Type: chaine de caracteres
#             num_impr    -> Desc: repertoire dans le quel est recherche le
#                                  contenant la chaine
#                            type: chaine de caracteres
#
# 2 utilisations possibles :
# usage n0 1) : "pattsch chaine"
# exemple     : "pattsch autorisation" (ici permet de retrouver tous les
#                                  fichiers contenant la chaine:
   "autorisation"
#                                  a partir du repertoire racine "/")
# usage n0 2) : "pattsch chaine nom_de_repertoire"
# exemple     : "pattsch autorisation INP" (ici permet de retrouver tous
   les
#                                 fichiers contenant la chaine:
   "autorisation"
#                                 et qui se trouvent sous le repertoire
   "INP" )
#
#---------------------------------------------------------------------------
   -----

USAGE="usage= $0 pattern [ directory ] "

if test $# -eq 0
then
    echo $USAGE
    exit 1
fi

if test $# -gt 2
then
    echo $USAGE
    exit 1
fi
```

```
echo "research of your chain '$1' can be long: to stop research type F5
   key"
if test $# -eq 1
then
    for fil in `find / -print`
    do
         fgrep -ln $1 $fil
    done
fi

if test $# -eq 2
then
    for fil in `find $2 -print`
    do
         fgrep -ln $1 $fil
    done
fi

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

14 14 Programme de recherche simple d'un fichier a partir d'un répertoire : where

Note : la commande **where** existe dans certains systèmes UNIX. Sinon, une commande équivalente : **which**

```
#---------------------------------------------------------------------------
   -----
# @(#) where  : file search in directories
# nom module  : where
# type module : procedure shell-UNIX
# but         : permet de retrouver facilement un fichier ou un repertoire
#              (en donnant ou non son chemin d'acces) a partir du
   repertoire "/"
#               si le nom du repertoire n'est pas donne
#               sinon a partir du repertoire qui est percise comme ci-
   dessous
# Auteur      : Benjamin LISAN        date creation : 01-FEB-90
#
# 2 utilisations possibles :
# usage n0 1) : "where nom_de_fichier"
# exemple     : "where *.rpt" (ici permet de retrouver tous les fichiers
#                             d'extension : ".rpt")
# usage n0 2) : "where chemin_d_acces nom_de_fichier"
# exemple     : "where INP *.frm" (ici permet de retrouver tous les
   fichiers
#                                 d'extension : ".frm" qui se trouvent
   sous le
#                                 repertoire "INP" )
#---------------------------------------------------------------------------
   -----
#
if test $# -lt 2
then
    echo "research of your file '$1' can be long: to stop research type F5
   key"
    find / -name $1 -print 2> /dev/null
else
   echo "research of your file '$2' can be long: to stop research type F5
   key"
```

```
   find $1 -name $2 -print 2> /dev/null
fi

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

15 15 Programme initialisation de l'environnement de ksh : init_ksh

Note: prévu au cas ou dans le fichier **.profile** , lancé au logging de l'utilisateur, n'existerait pas les variables de "setting" de l'environnement de "ksh" (***VISUAL***, surtout les variables : **HISTFILE**, **HISTSIZE**).

```
# ------------------------------------------------------------
# ksh_init
# a lancer sous le Korn-shell par la commande ". ksh_init"
# (le point est obligatoire)
# ------------------------------------------------------------
set -o vi
UNAME=`uname`; export UNAME
PS1='$LOGNAME:$UNAME!$PWD >'

# Si on n'a pas ksh au login dans son environnement, on peut le lancer par
la
# commande « ksh –o vi ». ksh n'est pas utilisé pour la crontab étant
« root ».
```

```
#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>
```

16 16 Script d'impression : pr2

Note: ne marche pas sur tous les sites UNIX.

```
#---------------------------------------------------------------------------
   -----
# nom du module : pr2                                          Version:
   V1.0.0
# type de module: procedure shell-UNIX
# But: imprime un fichier sur l'imprimante standard (ici qms0)
#      avec impression en en-tete de chaque page du num du fichier imprime
#      suivi de la date d'impression et avec une marge de 7 caracteres
# Syntaxe: pr2 nom_fichier [num_impr]
# Arg.INPUT:  nom_fichier -> Desc: nom du fichier a imprimer
#                            Type: chaine de caracteres
#             num_impr    -> Desc: numero d'imprimante
#                            Exemple: hp0
#                            type: int
#---------------------------------------------------------------------------
   -----
if [ $# = 2 ]
then
      if [ "$2" = "hp0" ]
               then         # cas de la hp0 (Imprimante Hewlett-Packard)
                 pr -o7 -f $1 | lp -op:l -d$2
               else
                 pr -o7 -f $1 | lp -d$2
               fi
else
  pr -o7 -f $1 | lp
fi

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

17 17 Exemple de script d’arrêt de toutes les bases Oracle

```
#!/sbin/sh
#---------------------------------------------------------------------------
---
# @(#) arret_toutes_bases
# Nom / Name     : arret_toutes_bases oracle
# Type / Type    : job shell (UNIX)
# But / Aim      : arret  de toutes les bases
# Auteur / Author: Benjamin LISAN  ->    date creation:  01-AUG-90
#
# Usage / Usage  : arret_toutes_bases
# Contexte / Co. : etre logue 'oracle'
# Commentaires   : version pour oracle 7.3.4
#---------------------------------------------------------------------------
---
cd /logiciel/oracle
for dirora in `ls -d oracle_[a-zA-Z]*|grep -v oracle_util`
do
   sid=`echo $dirora | sed "s:oracle_::"`
   # arret de la base $sid
   test -d $dirora && test -f /logiciel/oracle/$dirora/LOCK \
       && stop_bd $sid
```

```
   test $? -eq 0 && touch /logiciel/oracle/$dirora/indiclock # positionne
indic :base ayant ete lancee
done

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

18 18 Exemple de script pour se connecter successivement à une liste de machines : cnx_ux

```
# cnx_ux : script pour se connecter a toutes les machines UX suivantes :
liste="
bali
barbade
cervin
sumatra
hawaii
olga
olga2
olga160
pgi_mlb
mayotte
strmlbibm
tahiti
an2m
"

for mach in $liste
do
  echo "Voulez-vous vous connexer vers la machine $mach (O/N) ? : \c"
  read repons
  case $repons in
     [Oo][Uu][Ii]|[Oo]) rlogin $mach ;;
      *) : ;;
  esac
done

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

19 19 Script copiant la « crontab » sur une succession de machines

(dont la liste est précisées dans un fichier)

```
# nom du script : cop_cron
for mach in `cat ls_mach` # ls_mach : ce fichier contient la liste des
machines
do
echo "$mach"
rcp -p $mach:/usr/spool/cron/crontabs/root /usr/local/divers/ls_cron/$mach
done

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>
```

20 20 Script copiant un script d'exploitation sur une succession de machines

```
# nom du script : cop_script_exploit
nom_fic="$1"
if [ $# -eq 0 ]
then
```

```
   while [ -z "$nom_fic" ]
   do
      echo "donner le nom du fichier a envoyer : \c"
      read nom_fic
   done
fi
for machine in `cat << %%
hawaii
cervin
barbade
pgimlb
olga
olga2
olga160
sumatra
mayotte
an2m
bali
tahiti
%%`
do
    echo "copie du fichier '$nom_fic' vers la machine: $machine"
    rcp -p /usr/local/bin/$nom_fic $machine:/usr/local/bin/
done
#barbade

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
# variante du script ci-avant
# cop_script_expl

if [ "$#" -eq 0 ]
then
  echo "ce script copie le fichier, dont le nom est passe en"
  echo "argument sur toutes les machines du centre"
  echo "exemple: cop_script_hp script_exemple"
  exit 1
fi
echo "copie du script $1 vers les machines du centre informatique"
for machine in `cat << %%
barbade
cervin
olga
olga2
olga160
sumatra
mayotte
seychell
dpli1
%%`
do
    echo "copie du fichier '$1' vers la machine: $machine"
    rcp /usr/local/bin/$1 $machine:/usr/local/bin/
done

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
# variante du script ci-avant
# cop_script_selon_os
if [ "$#" -eq 0 ]
then
```

```
  echo "ce script copie le fichier, dont le nom est passe en"
  echo "argument sur toutes les machines HP et SUN du centre"
  echo "exemple: cop_script_selon_os vol_tot_vg"
  exit 1
fi
echo "1) copie des scripts pour OS HP"
for machine in `cat << %%
hawaii
cervin
pgimlb
olgaopti
olga
olga2
olga160
sumatra
mayotte
%%`
do
    echo "copie du fichier '$1' vers la machine: $machine"
    rcp /usr/local/bin/$1.hp $machine:/usr/local/bin/
done
echo "2) copie des scripts pour OS SUN"
for machine in `cat << %%
bali
tahiti
an2m
%%`
do
    echo "copie du fichier '$1' vers la machine: $machine"
    rcp /usr/local/bin/$1.sun $machine:/usr/local/bin/
done

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

21 21 Script calculant l'espace libre sur machine HP : esp_alloue_libre_hp

```
#!/sbin/sh
#-------------------------------------------------------------
# @(#) esp_alloue_libre_hp
# Nom / Name     : esp_alloue_libre pour HP
# Type / Type    : job shell (UNIX)
# But / Aim      : calcule espace libre alloue
# Auteur / Author: Benjamin LISAN  date creation: 10/01/2000
# Usage / Usage  : esp_alloue_libre_hp
# Exemple / Ex.  : esp_alloue_libre_hp
# Contexte utili.: cron
# Comment.       : version pour HPUX 10.20 et >
#-------------------------------------------------------------
#rm -f  /tmp/esp_alloue_libre[0-9]* 2> /dev/null
echo "--------------------------------------"
echo "Epace alloue libre machine HP `uname -n` le `date '+%d/%m/%Y'` \n"
echo "partition\tlibre (Mo)\tDisque/applica."
echo "---------\t----------\t---------------"

# raison du grep -v workdir : suppression liens NFS sur SUMATRA
df -b|sed "s:/dev::"|grep -v workdir|\
awk -v mac=`uname -n` 'BEGIN {sum=0}
{if($5/1024 > 1000)
        {
           {sum+=$5/1024}
```

```
            {printf("%s\t%s\t\t%s)\n",$1,$5/1024,$2)}

        }
}
END {printf("\nPlace totale allouee libre (Mo) sur %s : %s\n",mac, sum)}'
```

#>>
>>>>>>

22 22 Script calculant l'espace libre sur machine SUN : esp_alloue_libre_sun

```
#!/sbin/sh
#-------------------------------------------------------------
# @(#) esp_alloue_libre_sun
# Nom / Name     : esp_alloue_libre pour SUN
# Type / Type    : job shell (UNIX)
# But / Aim      : calcule espace libre alloue
# Auteur / Author: Benjamin LISAN  date creation: 10/01/2000
# Usage / Usage  : esp_alloue_libre_sun
# Exemple / Ex.  : esp_alloue_libre_sun
# Contexte utili.: cron
# Comment.       : version pour SUN 2.5 et >
#-------------------------------------------------------------
#rm -f  /tmp/esp_alloue_libre[0-9]* 2> /dev/null
echo "----------------------------------------------------------"
echo "Epace alloue libre machine SUN `uname -n` le `date '+%d/%m/%Y'` \n"
echo "Nom partition phys.\tlibre (Mo)\tPartition "
echo "-------------------\t----------\t----------"
df -k|grep -v Filesystem|grep -v proc|grep -v fd|grep -v swap|\
/usr/xpg4/bin/awk -v mac=`uname -n` 'BEGIN {sum=0}
{if($4/1024 > 1000)
        {
           {sum+=$4/1024}
           {printf("%s\t%s\t\t%s\n",$1,$4/1024,$6)}
        }
}
END {printf("\nPlace totale allouee libre (Mo) sur %s : %s\n",mac, sum)}'
```

#>>
>>>>>>

23 23 Script donnant espace disque libre sur toutes les machines du centre informatique

```
#!/sbin/sh
#--------------------------------------------------------------------
# @(#) esp_dsk_libre_centre
# Nom / Name     : esp_dsk_libre_centre
# Type / Type    : job shell (UNIX)
# But / Aim      : donne espace disque libre (alloue ou non alloue)
#                  sur toutes les machines du centre informatique
# Auteur / Author: Benjamin LISAN date creation: 01/11/99
# Usage / Usage  : esp_dsk_libre_centre
# Exemple / Ex.  : esp_dsk_libre_centre
# Contexte / Co. : cron
# Commentaires   : version pour HPUX 10.20
#--------------------------------------------------------------------
# script pour obtenir l espace non alloue de toutes les machines du centre
info
fic=/usr/local/divers/esp_non_alloue_centre.res
rm $fic 2> /dev/null
liste="
barbade
cervin
sumatra
hawaii
olga
```

```
olga160
pgi_mlb
mayotte
"

for mach in $liste
do
 remsh $mach -n 'sh /usr/local/bin/esp_non_alloue_hp'|tee -a $fic
 remsh $mach -n 'sh /usr/local/bin/esp_alloue_libre_hp'|tee -a $fic
done

liste="
bali
java
tahiti
"
for mach in $liste
do
 remsh $mach -n '/usr/local/bin/esp_non_alloue_sun'|tee -a $fic
 remsh $mach -n '/usr/local/bin/esp_alloue_libre_sun'|tee -a $fic
done

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

24 24 Script calculant l'espace non alloué sur machine HP : esp_non_alloue_hp

```
#!/sbin/sh
#---------------------------------------------------------------------------
---
# @(#) esp_non_alloue_hp
# Nom / Name     : esp_non_alloue_hp
# Type / Type    : job shell (UNIX)
# But / Aim      : statistique sur la place disque restante sur les disques
# Auteur / Author: Benjamin LISAN  ->    date creation:  01/12/99
#
# Usage / Usage  : esp_non_alloue_hp
# Exemple / Ex.  : "esp_non_alloue_hp"
# Contexte / Co. : cron
# Commentaires   : version pour HPUX 10.20 avec LVM
#---------------------------------------------------------------------------
---
tot_free=0
echo
"=====================================================================
echo "Epace disque non alloue machine HP `uname -n` le `date '+%d/%m/%Y'`
\n"
echo "Disque \tEspace libre (Mo)"
echo "-------\t-----------------"

for vg in `ls -d /dev/vg*`
do
  free_PE=`/sbin/vgdisplay $vg | grep "Free PE"| sed -e "s:Free PE::" -e
"s: ::g"`
  size_PE=`/sbin/vgdisplay $vg | grep "PE Size (Mbytes)" | sed -e "s:PE
Size (Mbytes)::" -e "s: ::g"`
 # echo "free_PE='$free_PE' size_PE='$size_PE'"
  vol_free=`expr $free_PE \* $size_PE`
  echo "`basename $vg`\t$vol_free"
```

```
  tot_free2=$tot_free
  tot_free=`expr $tot_free2 + $vol_free`
done
echo "\nTotal place libre disque non alloue sur `uname -n`\t : $tot_free
Mo"
echo "-------------------------------------------------------------------"
```

```
#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>

#!/bin/sh
# @esp_non_alloue_sun
# Determine l'espace disque non alloue sur les disques.
# Commentaire : marche sur machine SUN (Solaris 2.5 et > )

tot_free=0
echo
"========================================================================"
echo "Epace disque non alloue machine SUN `uname -n` le `date '+%d/%m/%Y'`
\n"
echo "disque\tEspace libre en Mo"
echo "------\t------------------"

# Pour chaque disque sur la machine
for disk in `format < /dev/null|awk '{print $2}'|grep c|grep d|grep t`
do
  blocks_free=`prtvtoc -f /dev/rdsk/${disk}s2 \
  |awk '{FS="="}{print $3}' | awk '{print $1}'`
  kb_free=`expr $blocks_free / 2`
  Mb_free=`expr $kb_free / 1024`
  echo "$disk\t$Mb_free"
  tot_free=`expr $Mb_free + $tot_free`
done
echo "\nTotal place libre disques non alloue sur `uname -n`\t : $tot_free
Mo"
echo "--------------------------------------------------------------------"

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

25 25 script de recherche d'une chaîne de caractères dans tous les fichiers ascii, situés dans une arborecence

```
#!/bin/sh
#-------------------------------------------------------------------------
---
# @(#) find_text : pattern search in files, belonging to a same root
directory
# Type / Type    : job shell (UNIX)
# But / Aim      : recherche d'une chaine de caracteres dans tous les
fichiers
#                  ascii, situes dans l'arborecence d'un repertoire
# Auteur / Author: BL                          date creation:  01-AUG-98
# Usage / Usage  : "find_text directory pattern
# Exemple / Ex.  : "find_text /etc 255.255.255.0"
#-------------------------------------------------------------------------
---
if [ $# -ne 2 ]
then
   echo "usage : `basename $0` search_dir search_text"
   exit 1
fi
dir_dep="$1"
text_rech="$2"
export dir_dep text_rech

for fic in `find $dir_dep`
do
  file $fic | grep text > /dev/null
  Ret="$?"
  if [ "$Ret" = "0" ]
  then
    grep -l  $text_rech $fic
   [ $? -eq 0 ] && echo " ================"
    grep $text_rech $fic
  fi
done


#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

26 26 script d'impression sur une seule imprimante des tailles des partitions de chacune des machines du centre

```
# imp_partition_size_of_each_mach :
# script pour se conncter a toutes les machines UX suivantes
# et imprimer les tailles des partitions de ces machines, sur une seule
imprimante :
liste="
cervin
sumatra
hawaii
olga
olga2
olga160
pgi_mlb
mayotte
"
```

```
for mach in $liste
do
  echo "impression sur KYOCERA1 des tailles des partitions de la machine
$mach"
  remsh $mach -n '(
  mach=`uname -n`;
 bdf | sed -e "/Filesystem/{
i\\
$mach
}" | lp -dKYOCERA1
)'
done

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

27 27 script listant, dans un fichier unique, tous les scripts developpés pour le centre

```
#!/sbin/sh
#-------------------------------------------------------------------------
---
# @(#) inv_local_bin
# Nom / Name     : inv_local_bin
# Type / Type    : job shell (UNIX)
# But / Aim      : liste dans un fichier unique de tous les scripts
developpes
#                  pour le centre. Puis impression (a la demande)
# Auteur / Author: Benjamin LISAN  ->    date creation:  15/5/00
#
# Usage / Usage  : inv_local_bin
# Exemple / Ex.  : "inv_local_bin"
# Contexte / Co. : cron
# Commentaires   : version pour HPUX 10.20
#-------------------------------------------------------------------------
---
ls /usr/local/bin/*|grep -v old|grep -v makedepend|grep -v ess \
       > /usr/local/divers/local_bin.inv  #|grep -v $0
# modif. eventuelle de la liste
vi /usr/local/divers/local_bin.inv

> /usr/local/divers/script_loc_bin.txt # raz de l inventaire
for fic in `cat /usr/local/divers/local_bin.inv`
do
   cat $fic >> /usr/local/divers/script_loc_bin.txt
   echo
"\f#=========================================================================" \
                            >> /usr/local/divers/script_loc_bin.txt
done
reponse=""
while [ -z "$reponse" ]
do
   echo "Voulez vous verifier le fichier contenant le texte des scripts
developpes au CENTRE ? :\c"
   read reponse
   case $reponse in
       O|o|OUI|Oui|oui) vi /usr/local/divers/script_loc_bin.txt ;;
       *) : ;;
   esac
done
reponse=""
```

```
while [ -z "$reponse" ]
do
   echo "Voulez vous imprimer le texte des scripts developpes au centre ?
:\c"
   read reponse
   case $reponse in
       O|o|OUI|Oui|oui) lp -d suzuk_gos
/usr/local/divers/script_loc_bin.txt ;;
       *) : ;;
   esac
done

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

28 28 script de lancement de fenêtres ouvertes HPView sur une succession de machines HP

```
# lancement de fenêtres ouvertes HPView sur toutes les machines HP
suivantes

hpterm -name sumatra &
hpterm -name bali &
hpterm -name mayotte &
hpterm -name an2m &
hpterm -name seychell &
# ….

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

29 29 script fusionnant tous les fichiers hosts des machines, pour obtenir la liste de toutes les adresses IP existantes

```
# liste des addresses IP de toutes les machines suivantes
rm /tmp/ls_adr_ip.txt
liste="
barbade
cervin
sumatra
hawaii
olga
olga2
olga160
pgi_mlb
mayotte
tahiti
bali
java
"

for mach in $liste
do
  remsh $mach -n 'cat /etc/hosts' |grep -v "^#" \
 | grep -v "^$"|grep -v "^impression" \
 >> /usr/local/divers/ls_adr_ip.txt
done
```

```
sort -u /usr/local/divers/ls_adr_ip.txt > /usr/local/divers/lst_adr_ip.txt

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

30 30 script de copie d’une liste de scripts d’administration, sur les machines du centre

```
# outils de copie d’une liste de scripts d’administration, sur les machines
   du centre

cd /usr/local/bin
ls_fic='SAUVE_JOUR_BANDE
SAUVE_SYS_TAPE
arret_toutes_bases
cnvdosux
cnvuxdos
cnx_mac
compc
contxsch
count_C.sql
dir
dtree
esp_alloue_libre_hp
esp_non_alloue_hp
find_text
head1
imp_bdf_mac
imp_df_mac
imp_host
kill_uucp
ls_adr_ip
ls_mach
lsfic_cache
makedepend
pattsch
purge_fic_old_tmp.hp
purge_gros_fic
reboot_auto.hp
relance_des_bases
surv_mem
surv_mem_swap.hp
surv_nb_usr_ora
tet
top_50_dsk.hp
tree
tstcnx
tstmail
verif_occup_cpu.hp
verif_parti_full.hp
vm2
vol_ora
vol_orasys
vol_restanthp
where'
for fic in `echo $ls_fic`
do
  echo "rcp $fic seychell:/usr/local/bin"
  rcp -p $fic seychell:/usr/local/bin
done
```

```
#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

31 31 script de purge des fichiers anciens et temporaires (version pour HP)

```
#!/bin/sh
#---------------------------------------------------------------------------
---
# @(#) purge_fic_old_tmp
# Nom / Name     : purge_fic_old_tmp
# Type / Type    : job shell (UNIX)
# But / Aim      : supprimer tous les fichiers temporaires anciens
#                  (afin de faire de la place sur la machine)
# Auteur / Author: Benjamin LISAN  ->    date creation:  01-AUG-90
# Usage / Usage  : purge_fic
# Exemple / Ex.  : "purge_fic"
# Contexte / Co. : cron
# Commentaires   : version pour HPUX 10.20
#---------------------------------------------------------------------------
---
find /usr/tmp -type f -mtime +7 -exec rm {} \;
find /tmp -type f -mtime +7 -exec rm {} \;
find /lost+found -type f -mtime +7 -exec rm {} \;
find /usr/lost+found -type f -mtime +7 -exec rm {} \;
find /var/lost+found -type f -mtime +7 -exec rm {} \;
find /var/adm/sw/queue/logfiles -type f -mtime +7 -exec rm {} \;
find /var/adm/syslog -type f -mtime +7 -exec rm {} \;
find /var/opt/hparray/log -type f -mtime +7 -exec rm {} \;
find /var/adm/log -type f -mtime +7 -exec rm {} \;
find /var/sam/log -type f -mtime +7 -exec rm {} \;
#find /usr/sam/log -type f -mtime +7 -exec rm {} \;

find /var/X11/Xserver/logs -type f -mtime +7 -exec rm {} \;
find /appli -name core -type f -mtime +1 -exec rm {} \;
# find /intg -name core -type f -mtime +1 -exec rm {} \;
find /appli -name toto -type f -mtime +7 -exec rm {} \;
# find /intg -name toto -type f -mtime +7 -exec rm {} \;
find /usr/spool/sockets -type f -mtime +7 -exec rm {} \;
find /usr/spool/lp/request -type f -mtime +7 -exec rm {} \;
find /logiciel/mqm/var/mqm/qmgrs/FRRUM000/@ipcc/ssem/ -name "socket*" -type
f -mtime +7 -exec rm {} \;
find /logiciel/cft/fillog -type f -mtime +7 -exec rm {} \;
find /logiciel/oracle/oracle_7162/rdbms/log -type f -mtime +7 -exec rm {}
\;
find /logiciel/oracle/oracle_7162/rdbms/log -type f -mtime +7 -exec rm {}
\;
find /logiciel/oracle/oracle_7162/tcp/log -type f -mtime +7 -exec rm {} \;
find /logiciel/oracle/oracle_7162/network/log -type f -mtime +7 -exec rm {}
\;
find /logiciel/oracle/oracle_734/network/log -type f -mtime +7 -exec rm {}
\;
find /logiciel/oracle/oracle_734/rdbms/log -type f -mtime +7 -exec rm {} \;
find /logiciel/tuxedo/tuxedo_decoupe/logs -type f -mtime +7 -exec rm {} \;
find /logiciel/dm/dba/log -type f -mtime +7 -exec rm {} \;

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

32 32 script de purge des fichiers anciens et temporaires (version pour SUN)

```
#!/bin/sh
```

```
#------------------------------------------------------------------------
---
# @(#) purge_fic_old_tmp
# Nom / Name     : purge_fic_old_tmp
# Type / Type    : job shell (UNIX)
# But / Aim      : supprimer tous les fichiers temperoires anciens
#                  faire de la place sur la machine
# Auteur / Author: Benjamin LISAN  ->    date creation:  01-AUG-90
# Usage / Usage  : purge_fic
# Exemple / Ex.  : "purge_fic"
# Contexte / Co. : cron
# Commentaires   : version pour Solaris 2.5 et 2.6
#------------------------------------------------------------------------
---
find /usr/tmp -type f -mtime +7 -exec rm {} \;
find /lost+found -type f -mtime +7 -exec rm {} \;
find /usr/lost+found -type f -mtime +7 -exec rm {} \;
find /var/lost+found -type f -mtime +7 -exec rm {} \;
find /intg -name core -type f -mtime +1 -exec rm {} \;
find /intg -name toto -type f -mtime +7 -exec rm {} \;
find /logiciel/oracle/oracle_734/network/log -type f -mtime +7 -exec rm {}
\;
find /logiciel/oracle/oracle_734/rdbms/log -type f -mtime +7 -exec rm {} \;
find /logiciel/tuxedo/tuxedo_carnets/tux_usi/logs -type f -mtime +7 -exec
rm {} \;
find /logiciel/tuxedo/tuxedo_carnets/tux_fch/logs -type f -mtime +7 -exec
rm {} \;
find /logiciel/tuxedo/tuxedo_carnets/tux_abe/logs -type f -mtime +7 -exec
rm {} \;
find /logiciel/tuxedo/tuxedo_carnets/tux_bug/logs -type f -mtime +7 -exec
rm {} \;
find /logiciel/tuxedo/tuxedo_carnets/tux_esz/logs -type f -mtime +7 -exec
rm {} \;
find /logiciel/tuxedo/tuxedo_carnets/tux_mer/logs -type f -mtime +7 -exec
rm {} \;
find /logiciel/tuxedo/tuxedo_carnets/tux_jan/logs -type f -mtime +7 -exec
rm {} \;
find /logiciel/tuxedo/tuxedo_carnets/tux_epo/logs -type f -mtime +7 -exec
rm {} \;
find /logiciel/tuxedo/tuxedo_carnets/tux_sok/logs -type f -mtime +7 -exec
rm {} \;
find /logiciel/tuxedo/tuxedo_decoupe/tux_wma/logs -type f -mtime +7 -exec
rm {} \;
find /logiciel/tuxedo/tuxedo_decoupe/tux_rec/logs -type f -mtime +7 -exec
rm {} \;

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

33 33 script de purge des fichiers log ou temporaires trop gros

```
#!/bin/sh
#------------------------------------------------------------------------
---
# @(#) purge_gros_fic
# Nom / Name     : purge_gros_fic
# Type / Type    : job shell (UNIX)
# But / Aim      : 1) raccourcir les fichiers temporaires ou log trop gros
#                  2) deplacer les fichiers .Z .z .gz .tar .cpio trop gros
#                     dans /export/depot
#                     afin de faire de la place dans /logiciel
```

```
# Auteur / Author: Benjamin LISAN  ->    date creation:  15/12/99
# Usage / Usage  : purge_gros_fic
# Exemple / Ex.  : purge_gros_fic
# Contexte / Co. : cron
# Commentaires   : version pour HPUX 10.20
#-------------------------------------------------------------------------
---
echo "1/ raccourcir les fichiers log ou tmp, trop long, au 500 derniers
enr"
for fic in `find /logiciel -size +1000000c -type f \( -name *.tmp -o -name
*.log \)`
do
   ls -l $fic
   head -3   $fic >  /tmp/fic_a_racourc$$
   tail -500 $fic >> /tmp/fic_a_racourc$$
   mv /tmp/fic_a_racourc$$ $fic
done

echo "2/ deplacer les fichiers .Z .z .gz .tar ou .cpio, trop gros, dans
/export/depot"
test ! -d /export       && mkdir /export       2> /dev/null
test ! -d /export/depot && mkdir /export/depot 2> /dev/null
for fic in `find /logiciel -size +1000000c -type f \
 \( -name *.Z -o -name *.z -o -name *.gz -o -name *.tar -o -name *.cpio \)`
do
   ls -l $fic
   mv $fic /export/depot
done

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

34 34 script de reboot et de rédémarrage automatique (version pour HP)

```
#!/bin/ksh
# -------------------------------------------------------------------------
- #
# Auteur  : B.LISAN
# Date    : 25 fevrier 1999
# Objet   : REBOOT et REDEMARRAGE AUTOMATIQUE
# Comment : version pour HP
# -------------------------------------------------------------------------
- #
log=/var/adm/log/reboot.log; export log
echo `date` "REBOOT DE LA MACHINE: `uname -n`"| tee -a $log
cd /

# on ne reboot pas s'il a encore des usagers.
#/usr/local/bin/verif_users
Retour="$?"
if [ "$Retour" = "0" ]
then
    echo "plus d'utilisateurs -> on peut lancer le reboot" | tee -a $log
    # pour eviter que le shutdown soit le fils d'un produit d'admin. ITO...
    # kill ITO par le shutdown, on kill aussi le fils.
    # (at lui cree un job lie au cron donc au noyau)
    rm -f /tmp/shut[0-9]* > /dev/null 2> /dev/null
    # cette solution avec at est en prevision ou les batchs seraient
    # plannifies au niveau d'ITO et plus au niveau du cron
    echo "nohup /etc/shutdown -r -y 0  & " >> /tmp/shut$$
    at -f /tmp/shut$$ now
```

```
else
    echo "encore des utilisateurs actifs -> on ne peut lancer le reboot" |
\
    tee -a $log
fi
exit $Retour

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

35 35 script de reboot et de rédémarrage automatique (version pour SUN)

```
#!/bin/ksh
# ---------------------------------------------------------------------------
- #
# Auteur  : B.LISAN
# Date    : 25 fevrier 1999
# Objet   : REBOOT et REDEMARRAGE AUTOMATIQUE
# Comment : version pour SUN
# ---------------------------------------------------------------------------
- #
log=/var/adm/log/reboot.log; export log
echo `date` "REBOOT DE LA MACHINE: `uname -n`"| tee -a $log
cd /

# on ne reboot pas s'il a encore des usagers.
#/usr/local/bin/verif_users

Retour="$?"
if [ "$Retour" = "0" ]
then
    echo "plus d'utilisateurs -> on peut lancer le reboot" | tee -a $log
    # pour eviter que le shutdown soit le fils d'un produit d'admin. ITO...
    # kill ITO par le shutdown, on kill aussi le fils.
    # (at lui cree un job lie au cron donc au noyau)
    rm -f /tmp/shut[0-9]* > /dev/null 2> /dev/null
    # cette solution avec at est en prevision ou les batchs seraient
    # plannifies au niveau d'ITO et plus au niveau du cron
    echo "nohup /usr/sbin/reboot & " >> /tmp/shut$$
    at -f /tmp/shut$$ now
else
    echo "encore des utilisateurs actifs -> on ne peut lancer le reboot" |
\
    tee -a $log
fi
exit $Retour

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

36 36 script de relance des bases Oracle

```
#!/sbin/sh
#------------------------------------------------------------------------
---
# @(#) relance_des_bases
# Nom / Name     : relance_des_bases oracle
# Type / Type    : job shell (UNIX)
# But / Aim      : relance des bases
# Auteur / Author: Benjamin LISAN  ->    date creation:  09-NOV-99
# Usage / Usage  : relance_des_bases
# Contexte / Co. : etre logue 'oracle'
# Commentaires   : version pour oracle 7.3.4
#                  Ne sont relances que les bases qui etait lance avant le
#                  le lancement du script arret_toutes_bases
#------------------------------------------------------------------------
---
cd /logiciel/oracle
for dirora in `ls -d oracle_[a-zA-Z]*|grep -v oracle_util`
do

   sid=`echo $dirora | sed "s:oracle_::"`
   test -d $dirora && \
   # relance de la base $sid
   test -f /logiciel/oracle/$dirora/indiclock \
    && start_bd $sid
   rm -f /logiciel/oracle/$dirora/indiclock > /dev/null 2> /dev/null
done


   #OK=`cat /var/opt/oracle/oratab | grep -v "#"|grep -v "^$"|grep $sid |
awk -F':' '{print $3}'`
   #test -d $dirora && test ! -f /logiciel/oracle/$dirora/LOCK && test $OK
= 'Y' && start_bd $sid

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

37 37 Exemple de script de restauration d’une base réseau et fichiers à partir d’une bande

```
# ------------------------------------------------------------------------
#
# Projet        : PROJET                Sous-Projet  : sauv/resto
#
# Auteur        : B. LISAN              Date creation: 03/08/95  ..:..
#
# Nom programme : RESTO_DSK_BASE_CENTRE  Type Langage : Bourne SHELL
#
# Objet du trait:                       Version      : 1.1
#
#           restauration fichier base et hors base + fichiers variables
#
#           de type .seq et .txt
#
# Commentaire   :
#
# ------------------------------------------------------------------------
#
#ident "@(#)RESTO_DSK_BASE_CENTRE  Vers.1.1  03 Aout 1995 SMTT SRC"
decompress()
```

```
{
if test -f $DIR_SUP_SAUV/svg_*.Z ; then
   echo "$nom_pg : decompression fichiers sauvegarde disque svg_*.Z ?"
   echo "          cela prend 5 mn (O: oui, N: non): \c"
   read repons
   case "$repons" in
   O|o|OUI|oui)
       echo "$nom_pg: decompression des fichiers sauvg.disque svg_*.Z";
       uncompress -f $DIR_SUP_SAUV/svg_projet.Z 2> /dev/null;
       uncompress -f $DIR_SUP_SAUV/svg_seq.Z    2> /dev/null;
       uncompress -f $DIR_SUP_SAUV/svg_fap.Z    2> /dev/null;
       uncompress -f $DIR_SUP_SAUV/svg_oscar.Z  2> /dev/null;
       uncompress -f $DIR_SUP_SAUV/svg_fic.Z    2> /dev/null;
    ;;
     *) :
        ;;
   esac
fi
}

liste_connectes()
{
   # 1) extraction par  wc du nombre de lignes de finger

   nb_enr_finger=`finger 2> /dev/null| grep -v "Login" |wc -l`

   num_enr=1
   while [ "$num_enr" -lt "$nb_enr_finger" ]
   do
      # 2) extraction du contenu de chaque enregistrement
      #    et calcul de la longueur de chaque enregistrement

      contenu_enr=\
`finger 2>/dev/null | sort | grep -v Login | head -n "$num_enr" | tail -n
1`
      contenu_enr2=`who -u | head -n "$num_enr" | tail -n 1`
      chp1=`echo "$contenu_enr" | cut -c10-30`
      chp2=`echo "$contenu_enr" | cut -c67-72`
      test -z "$chp2" && chp2="?????"
      chp3=`echo "$contenu_enr" | cut -c1-8`
      chp4=`echo "$contenu_enr2" | cut -c12-36`
      chp5=`echo "$contenu_enr2" | cut -c52-66`
      echo "$chp1 $chp2 $chp3 $chp4 $chp5"
      num_enr=`expr $num_enr + 1`
   done
}
test_KSBASE_histo()
{
KSto=/appli/projet/data/DSK/dico/KSBASE.histo
etat_base=`cat $KSto`
if [ "$etat_base" != "0" ]; then
    echo "fichiers KSBASE.histo contenant la valeur: $etat_base \c"
    echo "(il devrait etre a 0)"
    echo "-> vous ne pouvez pas relancer la base"
    echo "(KSBASE.histo a 1 veut dire: base lance, arretee brutalement ou"
    echo " dernier recouvrement mal termine)"
    echo "Voulez-vous remettre cet indicateur a 0? O: oui, N: non: \c"
    read repons
    case "$repons" in
    O|o|OUI|oui) echo "0" > $KSto;
       echo "reset a 0 de KSBASE.histo (valeur: `cat $KSto`)"
```

```
      ;;
    *) echo "Vous ne pouvez pas relancer la base";
       echo "\007"; # bip
       exit 1
       ;;
    esac
fi
}
# ------------------------------------------------------------------------
-
debug()
{
    if [ "$DBG" = "Y" ] ; then
       echo "$nom_pg: taper une touche pour continuer (Retour=$Retour): \c"
       read repons
    fi
}
# ------------------------------------------------------------------------
-
DBG="N" # mode debugging : Y : mode debug active, N : mode desactive
nom_pg=`basename $0`
debug()
{
    if [ "$DBG" = "Y" ]; then
       echo "$nom_pg: taper une touche pour continuer (Retour=$Retour): \c"
       read repons
    fi
}
# ------------------------------------------------------------------------
-
# Debut  du programme principal
# ------------------------------------------------------------------------
-
DBG="N" # mode debut : Y : mode active , N : mode deactive
nom_pg=`basename $0`
ESP_SUP_SAUV=/appli/svgbase/svg_projet
export ESP_SUP_SAUV
DIR_SUP_SAUV=`dirname $ESP_SUP_SAUV`
export DIR_SUP_SAUV
PATH_DICO=/appli/projet/data/DSK/dico/
export PATH_DICO
ESPX_SAUV_R=O
export ESPX_SAUV_R
cd /

echo "$nom_pg: Debut de la restauration disque des donnees de PROJET"
echo "$nom_pg: ------------------------------------------------------"
if [ "$LOGNAME" != "projet" ]; then
   echo "$nom_pg: il faut etre loggue 'projet' pour lancer ce programme"
   echo "$nom_pg: votre loggin n est pas 'projet' -> sortie du programme"
   echo "\007"; # bip
   exit 1
fi
echo "$nom_pg: date de debut sauveg. : `date`"
echo "$nom_pg: support de sauvegarde : $ESP_SUP_SAUV"
echo "$nom_pg: sauv.relative ?       : $ESPX_SAUV_R"
echo "$nom_pg: dictionnaire donnees  : $PATH_DICO"
echo "$nom_pg: repertoire de la base : $meteor/espace"
echo "$nom_pg:votre repertoire actuel: `pwd`"
echo "$nom_pg: rep.fichiers hors base: $opr"
echo "$nom_pg: Note : ce script doit etre lance avec le login : projet"
```

```
echo "$nom_pg: votre loggin          : $LOGNAME"
echo "\nCes parametres vous conviennent?"
echo "si Oui taper la touche Entree, sinon ensemble les touches Ctrl et C:
\c"
read repons
echo "$nom_pg: `finger 2>/dev/null | grep -v "Login" | wc -l` \c"
echo "utilisateurs encore actifs : "
echo "Nom Utilisateur       Tel   Login    Num.Terminal Date connect
Adresse IP
-------------------- ----- -------- ------------ ------------ ------------
---"
liste_connectes
echo "taper Entree : \c"
read repons
echo "$nom_pg: `etat | wc -l` programmes LEM encore actifs : "
etat | grep -v "A    "

decompress

echo "$nom_pg: verification presence fichiers de sauvegarde disque :"
echo "$nom_pg: (il doit y avoir au minimum le fichier suivant : svg_projet"
echo "$nom_pg: sinon il peut y avoir jusqu'a 5 fichiers:"
echo "$nom_pg: svg_projet svg_seq svg_fap svg_oscar svg_fic)"
if test ! -f $ESP_SUP_SAUV ; then
  echo "$nom_pg: pas de presence du fichier disque de sauvegarde des
fichiers"
  echo "$nom_pg: base et hors base (svg_projet) -> sortie du script !!!!!"
  echo "\007"; # bip
  exit 1
fi
ls -l $DIR_SUP_SAUV/svg_*
echo "Est-ce que cela vous convient? si Oui taper touche Entree, sinon ^C:
\c"
read repons


echo "$nom_pg: sauvegarde fichiers non traces, fich. Masques KV0099 et
POINTS"
for fic in `cat $PATH_DICO/HORSLOG`
do
   mkdir $meteor/REF 2> /dev/null
   test -f $opr/${fic}.dat && cp -p $opr/${fic}.dat $meteor/REF/.
   test -f $opr/${fic}.idx && cp -p $opr/${fic}.idx $meteor/REF/.
done

# test presence du programme de traitement des commandes Minitel : KP39
if ps -ef | grep "KP39" | grep -v "grep" ; then
   echo "$nom_pg:\007 minitel lancee"
   echo "demander l'arret du minitel, aux pilotes"
   exit 1
fi

# test presence du programme de reception des envois des depots Gold : SF0C
if ps -ef | grep "SF0C" | grep -v "grep" ; then
   echo "$nom_pg:\007 Reception gold active"
   echo "demander l'arret de gold, aux pilotes"
   exit 1
fi

# test presence de programme(s) LEM lance(s)
nb_pg=`etat | wc -l`
```

```
if [ "$nb_pg" != "0" ] ; then
   echo "$nom_pg: `date`: $nb_pg programmes LEM actifs \007 :"
   etat
   echo "demander aux utilisateurs, d'arreter leur programmes"
   exit 1
fi

# test presence de la base lancee
if ps -ef | grep "runesp" | grep -v "grep" ; then
   echo "$nom_pg: `date`: base lancee \007"
   echo "demander l'arret de la base PROJET aux pilotes\007"
   exit 1
fi

if test -f $ESP_SUP_SAUV ; then
  echo "$nom_pg: Nettoyages des fichiers SI de la base et SI hors base ?"
  echo "$nom_pg: \c"
  tput rev
  tput smul
  echo "Cette etape est OBLIGATOIRE!\c"
  tput sgr0
  echo " (O: oui, N: non): \c"
  read repons
  case "$repons" in
  O|o|OUI|oui|Oui)
     echo "$nom_pg: debut nettoyage fichiers de la base";
     find $meteor/espace \( -name "*.dat" -o -name "*.idx" \) -exec rm {}
\;
     echo "$nom_pg: et nettoyage fichiers SI (sequentiels indexes) hors
base";
     find $opr \( -name "*.dat" -o -name "*.idx" \) -exec rm {} \;
     #echo "$nom_pg: Fin nettoyage fichiers SI de la base et hors base";
     echo "$nom_pg: liste fichiers base non effaces s ils existent
encore:";
     ls $meteor/espace/*.dat $meteor/espace/*.idx 2> /dev/null;
     echo "$nom_pg: liste fichiers SI hors base non effaces, s ils
existent:";
     ls $opr/*.dat $opr/*.idx 2> /dev/null;
     echo " ";;
  *) : ;;
  esac
  # ---
  echo "$nom_pg: Nettoyages non obligatoire des fichiers .seq meteor,"
  echo "         et fichiers files d'attentes .fap ?(O: oui, N: non): \c"
  read repons
  case "$repons" in
  O|o|OUI|oui|Oui)
     echo "$nom_pg: debut nettoyage fichiers files d'attentes fap";
     find $opr  -name "*.fap" -exec rm {} \;
     echo "$nom_pg: debut nettoyage fichiers .seq";
     find $meteor/DSK -name "*.seq" -exec rm {} \;
     echo " ";;
  *) : ;;
  esac
  debug

  debug
  tput rev
  tput smul
  echo "L'etape suivante est OBLIGATOIRE!"
  tput sgr0
```

```
  echo "$nom_pg: Restauration disque fichiers c_isam base et hors base?"
  echo " (O/N): \c"
  read repons
  case "$repons" in
  O|o|OUI|oui|Oui)
    echo "$nom_pg: Debut restauration disque de la base & fichiers hors
base";
    echo "$nom_pg: ------------------------------------------------------
-";
    echo "$nom_pg: A La question 'mot de passe' taper le mot de passe
TOTAL"
    echo "$nom_pg: A La question '[O|N]?' taper la lettre 'O'"
    echo "$nom_pg: Ne pas tenir compte du message :";
    echo " "
    echo "Mettre le support numero   xx dans /appli/svgbase/svg_projet";
    echo " "
    echo "Taper Entree, pour indiquer que vous avez lu ce dernier message:
\c";
    read repons;
    $ESPX_PATH_P/restobas KSBASE $PATH_DICO $ESP_SUP_SAUV #2> /dev/null;
    Retour=$?;;
  *) Retour=0 ;;
  esac
  if [ "$Retour" = "0" ]; then
      echo "$nom_pg : `date` : activation restobas OK :restoration de la
base"
      echo "$nom_pg : nbre de fichiers de la base dans le rep.espace: \c"
      ls $meteor/espace/* | wc -l
  else
      echo "$nom_pg : `date` : Erreur $Retour lors de l activation de
restobas"
      echo "\007"; # bip
      exit 1
  fi
  debug

  echo "$nom_pg: Restauration fichiers sequentiels de type '.seq' ? (O/N) :
\c"
  read repons
  case "$repons" in
  O|o|OUI|oui|Oui)
     echo "$nom_pg : restauration cpio des fichiers de type: .seq";
     cpio -icvBdumx < $DIR_SUP_SAUV/svg_seq;;
     *) :
     ;;
  esac
  debug

  echo "$nom_pg: Restauration fichiers files d'a., de type '.fap' ? (O/N) :
\c"
  read repons
  case "$repons" in
  O|o|OUI|oui|Oui)
     echo "$nom_pg : `date` : restauration cpio des fichiers .fap";
     cpio -icvBdumx < $DIR_SUP_SAUV/svg_fap;;
     *) : ;;
  esac

  echo "$nom_pg: Restauration fichiers oscar de type '.csv' ? (O/N) : \c"
  read repons
  case "$repons" in
```

```
  O|o|OUI|oui|Oui)
     echo "$nom_pg : `date` : restauration cpio des fichiers oscar .csv";
     cpio -icvBdumx < $DIR_SUP_SAUV/svg_oscar;;
     *) : ;;
  esac

  echo "$nom_pg: Restauration fichiers divers de type '.fic' ? (O/N) : \c"
  read repons
  case "$repons" in
  O|o|OUI|oui|Oui)
      echo "$nom_pg : restauration cpio du fichier des fichiers divers";
      cpio -icvBdumx < $DIR_SUP_SAUV/svg_fic;;
      *) : ;;
  esac

  echo "$nom_pg: restauration des fichiers Masques KV0099 et POINTS"

  for fic in `cat $PATH_DICO/HORSLOG`
  do
      test -f $meteor/REF/${fic}.idx && mv $meteor/REF/${fic}.idx $opr/.
      test -f $meteor/REF/${fic}.dat && mv $meteor/REF/${fic}.dat $opr/.
  done

  echo "$nom_pg: En cas de recouvrement, lancer le scripts RECOUVR_PROJET"
  echo "$nom_pg: situe dans le repertoire : /appli/projet/data/scripts \n"
  echo "$nom_pg: Demander au BT de relancer la base PROJET, GOLD et
minitel"
  echo "$nom_pg: et l'acces aux utilisateurs"
  echo "$nom_pg: FIN NORMALE de la restauration disque des donnees de
PROJET"
else
   echo "$nom_pg: restoration impossible, fichier/support sauvegarde
inconnu"
   echo "$nom_pg : `date` : fin Anormale de $nom_pg"
fi

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

38 38 Exemple de script de restauration par « cpio » à partir d'une bande

```
# ------------------------------------------------------------------------
#
# Projet        : CENTRE                Sous-Projet  : sauv/resto
#
# Auteur        : B. LISAN              Date creation: 03/08/98  ..:..
#
# Nom programme : RESTO_JOUR_CPIO_SUN   Type Langage : Bourne SHELL
#
# Objet du trait:                       Version      : 1.1
#
#           restauration fichier a partir de la sauvegarde faite par
#
#           SAUVE_JOUR_CPIO_SUN
#
# Commentaire   : script present sur les  machines SUN du centre
informatiq.#
# ------------------------------------------------------------------------
#
# BL 21/10/99 : restauration en mode compressee
```

```
# ------------------------------------------------------------------------
#
#ident "@(#)RESTO_JOUR_CPIO_SUN  Vers.1.1  03 Aout 1998"

debug()
{
    if [ "$DBG" = "Y" ] ; then
       echo "$nom_pg: taper une touche pour continuer (Retour=$Retour): \c"
       read repons
    fi
}
# ------------------------------------------------------------------------
-
# Debut  du programme principal
# ------------------------------------------------------------------------
-
DBG="N" # mode debut : Y : mode active , N : mode deactive
nom_pg=`basename $0`
# BL 21/10/99 : restauration en mode compressee (no rewind)
DEV_SUP_SAUV=/dev/rmt/0nc

echo "$nom_pg: Debut de la restauration disque des donnees"
echo "$nom_pg: ----------------------------------------"
if [ "$LOGNAME" != "root" ]; then
   echo "$nom_pg: il faut etre loggue 'root' pour lancer ce programme"
   echo "$nom_pg: votre loggin n est pas 'root' -> sortie du programme"
   echo "\007"; # bip
   exit 1
fi
echo "$nom_pg: date de debut sauveg. : `date`"
echo "$nom_pg: support de sauvegarde : $DEV_SUP_SAUV"
echo "$nom_pg:votre repertoire actuel: `pwd`"
echo "$nom_pg: Note : ce script doit etre lance avec le login : root"
echo "$nom_pg: votre loggin          : $LOGNAME"
echo "\nCes parametres vous conviennent?"
echo "si Oui taper la touche Entree, sinon ensemble les touches Ctrl et C:
\c"
read repons
echo "Nb d'utilisateurs encore actifs : \c"
who -u | wc -l
echo "taper Entree : \c"
read repons
echo "$nom_pg: verification presence device du lecteur de bande :"
if test ! -c $DEV_SUP_SAUV ; then
  echo "$nom_pg: pas de presence du device du lecteur de bande:
$DEV_SUP_SAUV"
  echo "$nom_pg: -> sortie du script !!!!!"
  echo "\007"; # bip
  exit 1
fi
ls -l $DEV_SUP_SAUV
echo "Est-ce que cela vous convient? si Oui taper touche Entree, sinon ^C:
\c"
read repons

debug
tput rev
tput smul
echo "$nom_pg: Restauration fichiers ? (O/N): \c"
tput sgr0
read repons
```

```
case "$repons" in
O|o|OUI|oui|Oui)
  echo "$nom_pg: Debut restauration fichiers";
  echo "$nom_pg: --------------------------";
  echo " "
  echo "Mettre la cassette DAT du bon jour dans le lecteur";
  echo " "
  echo "Taper Entree, pour indiquer que vous avez lu ce dernier message:
\c";
  read repons;;
*) echo "Pas de restauration"; exit 0 ;;
esac
debug
fic=""
while [ -z "$fic" ]
do
   echo "$nom_pg: nom du fichier ou du repertoire a restaurer ?"
   echo "$nom_pg: suggestion: saisir son nom precede de son chemin en
absolu"
   echo "$nom_pg: exemples: /appli/siam/cdm/fic* /data_021/oracle_gem  ..."
   echo "$nom_pg: pour restaurer un repertoire, faire suivre son nom d un
'*' "
   echo "$nom_pg: exemples: /appli/siam/cdm/fic* /usr/local/divers*"
   echo "$nom_pg: nom du/des fichier(s)/repertoire(s): \c"
   read fic
done
echo "$nom_pg: verification deja presence du/des fichiers a restaurer, \c"
echo "taper Entree : \c"
read repons
ls -l $fic

echo "$nom_pg: La restauration peut prendre qq mn a 2H max."
echo "$nom_pg: Restauration du/des fichier(s): $fic ? (O/N) : \c"
read repons
case "$repons" in
O|o|OUI|oui|Oui)
   echo "$nom_pg : rembobinage de la bande"
   mt rew
   Retour="$?"
   if [ "Retour" -ne "0" ]
   then
       echo "pas de bande dans le lecteur ou pb bande ou lecteur : $Retour"
       exit 1
   fi
   echo "$nom_pg : recherche/restauration cpio du/des fichiers: $fic"
   cd /
   cpio -icvBdum "${fic}*" <  $DEV_SUP_SAUV
   Retour="$?"
   if [ "Retour" -ne "0" ]
   then
       echo "$nom_pg: restauration impossible, fichier inconnu"
       echo "$nom_pg : `date` : fin Anormale de $nom_pg : $Retour"
       exit 1
   else
       echo "$nom_pg: FIN NORMALE de la restauration sur disque des
donnees"
       echo "$nom_pg: verification presence du/des fichiers restaures, "
       echo "taper Entree _pour sortir de la liste affichee, taper 'q' _ :
\c"
       read repons
       ls -l $fic | pg
```

```
        echo "taper Entree : \c"
        read repons
    fi ;;
    *)  echo "pas de restauration demandee"
    ;;
esac
debug
mt rew
echo "$nom_pg: fin du script de restauration"

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

39 39 Exemple de script de restauration par « ufsrestore » à partir d'une bande (Sun)

```
# ---------------------------------------------------------------------------
#
# Projet        : CENTRE                 Sous-Projet  : sauv/resto
#
# Auteur        : B. LISAN               Date creation: 03/08/98  ..:..
#
# Nom programme : RESTO_JOUR_UFS_SUN1    Type Langage : Bourne SHELL
#
# Objet du trait:                        Version      : 1.1
#
#           restauration fichier a partir de la sauvegarde faite par
#
#           SAUVE_JOUR_UFS_SUN1
#
# Commentaire   : script present sur les  machines SUN du centre
informatiq.#
# ---------------------------------------------------------------------------
#
# BL 21/10/99 : restauration « ufsrestore » en mode compressee
# ---------------------------------------------------------------------------
#
#ident "@(#)RESTO_JOUR_UFS_SUN1  Vers.1.1  03 Aout 1998"

debug()
{
    if [ "$DBG" = "Y" ] ; then
       echo "$nom_pg: taper une touche pour continuer (Retour=$Retour): \c"
       read repons
    fi
}
# ---------------------------------------------------------------------------
-
# Debut  du programme principal
# ---------------------------------------------------------------------------
-
DBG="N" # mode debut : Y : mode active , N : mode deactive
nom_pg=`basename $0`
# BL 21/10/99 : restauration en mode compressee (no rewind)
DEV_SUP_SAUV=/dev/rmt/0nc

echo "$nom_pg: Debut de la restauration disque des donnees"
echo "$nom_pg: -----------------------------------------"
if [ "$LOGNAME" != "root" ]; then
   echo "$nom_pg: il faut etre loggue 'root' pour lancer ce programme"
   echo "$nom_pg: votre loggin n est pas 'root' -> sortie du programme"
```

```
   echo "\007"; # bip
   exit 1
fi
echo "$nom_pg: date de debut sauveg. : `date`"
echo "$nom_pg: support de sauvegarde : $DEV_SUP_SAUV"
echo "$nom_pg:votre repertoire actuel: `pwd`"
echo "$nom_pg: Note : ce script doit etre lance avec le login : root"
echo "$nom_pg: votre loggin          : $LOGNAME"
echo "\nCes parametres vous conviennent?"
echo "si Oui taper la touche Entree, sinon ensemble les touches Ctrl et C:
\c"
read repons
echo "Nb d'utilisateurs encore actifs : \c"
who -u | wc -l
echo "taper Entree : \c"
read repons
echo "$nom_pg: verification presence device du lecteur de bande :"
if test ! -c $DEV_SUP_SAUV ; then
  echo "$nom_pg: pas de presence du device du lecteur de bande:
$DEV_SUP_SAUV"
  echo "$nom_pg: -> sortie du script !!!!!"
  echo "\007"; # bip
  exit 1
fi
ls -l $DEV_SUP_SAUV
echo "Est-ce que cela vous convient? si Oui taper touche Entree, sinon ^C:
\c"
read repons

debug
tput rev
tput smul
echo "$nom_pg: Restauration fichiers ? (O/N): \c"
tput sgr0
read repons
case "$repons" in
O|o|OUI|oui|Oui)
  echo "$nom_pg: Debut restauration fichiers";
  echo "$nom_pg: ----------------------------";
  echo " "
  echo "Mettre la cassette DAT du bon jour dans le lecteur";
  echo " "
  echo "Taper Entree, pour indiquer que vous avez lu ce dernier message:
\c";
  read repons;;
*) echo "Pas de restauration"; exit 0 ;;
esac
debug
fic=""
while [ -z "$fic" ]
do
   echo "$nom_pg: nom du fichier ou du repertoire a restaurer ?"
   echo "$nom_pg: suggestion: saisir son nom precede de son chemin en
absolu"
   echo "$nom_pg: exemples: /appli/siam/cdm/fic* /data_021/oracle_gem  ..."
   echo "$nom_pg: pour restaurer un repertoire, faire suivre son nom d un
'*' "
   echo "$nom_pg: exemples: /appli/siam/cdm/fic* /usr/local/divers*"
   echo "$nom_pg: nom du/des fichier(s)/repertoire(s): \c"
   read fic
done
```

```
echo "$nom_pg: verification deja presence du/des fichiers a restaurer, \c"
echo "taper Entree : \c"
read repons
ls -l $fic
# traitement du nom du fichier pour ufsrestore
nom_rep=`echo $fic|cut -d'/' -f1`
nom_fic_rel=`echo $fic|cut -d'/' -f2-`

echo "$nom_pg: La restauration peut prendre qq mn a 2H max."
echo "$nom_pg: Restauration du/des fichier(s): $fic ? (O/N) : \c"
read repons
case "$repons" in
O|o|OUI|oui|Oui)
   echo "$nom_pg : rembobinage de la bande"
   mt rew
   Retour="$?"
   if [ "Retour" -ne "0" ]
   then
       echo "pas de bande dans le lecteur ou pb bande ou lecteur : $Retour"
       exit 1
   fi
   echo "$nom_pg : recherche/restauration ufsrestore du/des fichiers: $fic"
   cd $nom_rep
   ufsrestore xf $DEV_SUP_SAUV $nom_fic_rel
   Retour="$?"
   if [ "Retour" -ne "0" ]
   then
       echo "$nom_pg: restauration impossible, fichier inconnu"
       echo "$nom_pg : `date` : fin Anormale de $nom_pg : $Retour"
       exit 1
   else
       echo "$nom_pg: FIN NORMALE de la restauration sur disque des
donnees"
       echo "$nom_pg: verification presence du/des fichiers restaures, "
       echo "taper Entree _pour sortir de la liste affichee, taper 'q' _ :
\c"
       read repons
       ls -l $fic | pg
       echo "taper Entree : \c"
       read repons
   fi ;;
   *)  echo "pas de restauration demandee"
   ;;
esac
debug
mt rew
echo "$nom_pg: fin du script de restauration"

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

40 40 Exemple de script de savegarde sur bande par « cpio » (HP)

```
# ------------------------------------------------------------------------
#
# Projet        : PROJET              Sous-Projet  : sauv/resto
#
# Auteur        : B. LISAN            Date creation: 03/08/95  ..:..
#
# Nom programme : SAUV_DSK_BASE_PROJET Type Langage : Bourne SHELL
#
```

```
# Objet du trait:                       Version      : 1.1
#
#           sauvegarde fichier base et hors base + fichiers variables
#
#           de type .seq et .txt
#
# Commentaire   :
#
# ------------------------------------------------------------------------
#
#ident "@(#)SAUV_DSK_BASE_PROJET  Vers.1.1  03 Aout 1995 SMTT SRC"

compression()
{
  if test -f $DIR_SUP_SAUV/svg_$1 ; then
     echo "$nom_pg : compression fichiers sauvegarde disque svg_$1 ?"
     echo "          cela prend 5 mn, O: oui, N: non :\c"
     read repons
     case "$repons" in
       O|o|OUI|oui) echo "$nom_pg: compression des fichiers sauvg.svg_$1";
                    compress -f $DIR_SUP_SAUV/svg_$1;;
       *) : ;;
     esac
  fi
}

liste_connectes()
{
   # 1) extraction par  wc du nombre de lignes de finger

   nb_enr_finger=`finger 2> /dev/null| grep -v "Login" |wc -l`

   num_enr=1
   while [ "$num_enr" -lt "$nb_enr_finger" ]
   do
      # 2) extraction du contenu de chaque enregistrement
      #    et calcul de la longueur de chaque enregistrement

      contenu_enr=\
`finger 2>/dev/null | sort | grep -v Login | head -n "$num_enr" | tail -n
1`
      contenu_enr2=`who -u | head -n "$num_enr" | tail -n 1`
      chp1=`echo "$contenu_enr" | cut -c10-30`
      chp2=`echo "$contenu_enr" | cut -c67-72`
      test -z "$chp2" && chp2="?????"
      chp3=`echo "$contenu_enr" | cut -c1-8`
      chp4=`echo "$contenu_enr2" | cut -c12-36`
      chp5=`echo "$contenu_enr2" | cut -c52-66`
      echo "$chp1 $chp2 $chp3 $chp4 $chp5"
      num_enr=`expr $num_enr + 1`
   done

}
test_KSBASE_histo()
{
KSto=/appli/projet/data/DSK/dico/KSBASE.histo
etat_base=`cat $KSto`
if [ "$etat_base" != "0" ]; then
    echo "fichiers KSBASE.histo contenant la valeur : $etat_base \c"
    echo "(il devrait etre a 0)"
    echo "-> vous ne pouvez pas relancer la base"
```

```
    echo "(KSBASE.histo a 1 veut dire: base lance, arretee brutalement ou"
    echo " dernier recouvrement mal termine)"
    echo "Voulez-vous remettre cet indicateur a 0? O: oui, N: non : \c"
    read repons
    case "$repons" in
    O|o|OUI|oui) echo "0" > $KSto;
       echo "reset a 0 de KSBASE.histo (valeur: `cat $KSto`)"
       ;;
    *) echo "Vous ne pouvez pas relancer la base";
       echo "\007"; # bip
       exit 1
       ;;
    esac
fi
}
debug()
{
    if [ "$DBG" = "Y" ] ; then
       echo "$nom_pg: taper une touche pour continuer (Retour=$Retour) :\c"
       read repons
    fi
}
# ------------------------------------------------------------------------
-
DBG="N" # mode debugging : Y : mode debug active, N : mode desactive
nom_pg=`basename $0`
ESP_SUP_SAUV=appli/svgbase/svg_projet
export ESP_SUP_SAUV
DIR_SUP_SAUV=`dirname $ESP_SUP_SAUV`
export DIR_SUP_SAUV
PATH_DICO=/appli/projet/data/DSK/dico/
export PATH_DICO
ESPX_SAUV_R=O
export ESPX_SAUV_R
cd /

echo "$nom_pg: Debut de la sauvegarde disque des donnees de PROJET"
echo "$nom_pg: ----------------------------------------------------"
if [ "$LOGNAME" != "projet" ]; then
   echo "$nom_pg: il faut etre loggue 'projet' pour lancer ce programme"
   echo "$nom_pg: votre loggin n est pas 'projet' -> sortie du programme"
   echo "\007"; # bip
   exit 1
fi
echo "$nom_pg: date de debut sauveg. : `date`"
echo "$nom_pg: support de sauvegarde : $ESP_SUP_SAUV"
echo "$nom_pg: dictionnaire donnees  : $PATH_DICO"
echo "$nom_pg: repertoire de la base : $meteor/espace"
echo "$nom_pg:votre repertoire actuel: `pwd`"
echo "$nom_pg: rep.fichiers hors base: $opr"
echo "$nom_pg: sauv.relative?        : $ESPX_SAUV_R"
echo "$nom_pg: Note : ce script doit etre lance avec le login : projet"
echo "$nom_pg: votre loggin          : $LOGNAME"
echo "\nCes parametres vous conviennent?"
echo "si Oui taper la touche Entree, sinon ensemble les touches Ctrl et C :
\c"
read repons
echo "$nom_pg: `finger 2> /dev/null | grep -v "Login" | wc -l` \c"
echo "utilisateurs encore actifs : "
echo "Nom Utilisateur       Tel   Login    Num.Terminal Date connect
Adresse IP
```

```
-------------------- ----- -------- ------------ ------------ ------------
---"
liste_connectes;
echo "taper Entree :\c"
read repons
echo "$nom_pg: `etat | wc -l` programmes LEM encore actifs : \n"
etat | grep -v "A    "
echo "$nom_pg: verification du lancement de la base : \n"
ps -ef | grep "runesp" | grep -v grep
echo "Est-ce que cela vous convient? si Oui taper touche Entree, sinon ^C :
\c"
read repons

echo "$nom_pg: creation liste des fichiers S.I. supplementaires a
sauvegarder"
find $opr \( -name "*.dat" -o -name "*.idx" \) -print | \
   egrep -v "/tmp|/bin|/sauv|/old" >  $PATH_DICO/sauvcompl
#vi  $PATH_DICO/sauvcompl

if ps -ef | grep "KP39" | grep -v "grep" ; then
   echo "$nom_pg: `date`: minitel lancee"
   echo "demander l'arret du minitel aux pilotes"
   exit 1
fi

if ps -ef | grep "SF0C" | grep -v "grep" ; then
   echo "$nom_pg: `date`: Reception gold active"
   echo "demander l'arret de gold aux pilotes"
   exit 1
fi

if [ "`etat | wc -l`" != "0" ] ; then
   echo "$nom_pg: `date`: programmes LEM encore actifs :"
   etat
   echo "demander l'arret des utilisateurs aux pilotes"
   exit 1
fi

if ps -ef | grep "runesp" | grep -v "grep" ; then
   echo "$nom_pg: `date`: base lancee"
   echo "demander l'arret de la base PROJET aux pilotes"
   exit 1
fi
echo "$nom_pg : controle de la coherence de la base par bcheck"
echo "$nom_pg : Note : controle TRES FORTEMENT RECOMMANDE !!!"
echo "$nom_pg : controle coherence de base? O pour oui, N pour non : \c"
read repons
case "$repons" in
  O|o|OUI|oui)
  $meteor/scripts/checkbase;
  Retour=$?
  ;;
  *) Retour=0
  ;;
esac

if [ "$Retour" != "0" ]; then
   echo "$nom_pg: erreur physique dans base arret ANORMAL de la sauveg.!";
   echo "$nom_pg: ou utilisateur encore connecte -> si c'est le cas : ";
   echo "$nom_pg: a. lui demander de se deconnecter";
   echo "         b. et relancer de nouveau ce script.";
```

```
   exit $Retour;
else
   echo "Pas d'erreur lors du check de base -> base OK, taper Retour : \c";
   read repons;
fi


echo "$nom_pg: appel prog.C de sauvegarde de la base & des fichiers hors
base"
echo "$nom_pg: DEBUT DE LA SAUVEGARDE DES DONNEES SUR DISQUE"
echo "$nom_pg: ------------------------------------------- \n"
echo "$nom_pg: Attention, ne pas tenir compte du message suivant : \n"
echo " "
tput smul
echo "Mettre un support dans /appli/svgbase/svg_projet"
tput sgr0
echo " "
echo "$nom_pg: Sauvegarde dsk.fichiers base & hors base? O: oui N: non: \c"
read repons
case "$repons" in
  O|o|OUI|oui)
               $ESPX_PATH_P/sauvebas KSBASE $PATH_DICO $ESP_SUP_SAUV -s ;
               Retour=$?
               compression projet
               ;;
  *) Retour=0
     ;;
esac
if [ "$Retour" = "0" ]; then
   echo "$nom_pg : `date` : activation sauvebas OK sauvegarde de la base"
else
   echo "$nom_pg : `date` : Erreur $Retour lors de l'activation de
sauvebas"
   echo "\007"; # bip
   exit 1
fi

echo "Sauvegarde disque des fichiers sequentiels de type .seq? (O/N) : \c"
read repons
case "$repons" in
  O|o|OUI|oui) echo "$nom_pg: debut sauvegarde cpio disque des fichiers
.seq";
      find $meteor/DSK -name "*.seq" -print | \
        cpio -ocvBdumx > $DIR_SUP_SAUV/svg_seq 2> $DIR_SUP_SAUV/svg_seq.log
        compression seq
    ;;
  *) :
    ;;
esac

echo "Sauvegarde disque des fichiers files d'attente de type .fap? (O/N) :
\c"
read repons
case "$repons" in
  O|o|OUI|oui) echo "$nom_pg: debut sauvegarde cpio disque des fichiers
.fap";
      find $opr -name "*.fap" -print | \
         cpio -ocvBdumx > $DIR_SUP_SAUV/svg_fap 2>
$DIR_SUP_SAUV/svg_fap.log
         compression fap
   ;;
```

```
 *) :
   ;;
esac

echo "Sauvegarde disque des fichiers oscar de type .csv? (O/N) : \c"
read repons
case "$repons" in
  O|o|OUI|oui) echo "$nom_pg: debut sauvegarde cpio des fichiers .csv";
    find $opr/oscar  -print | \
    grep -E "k[0-2][0-9]|m[0-2][0-9]|p[0-2][0-9]|c[0-2][0-9]|g[0-2][0-9]" |
\
    cpio -ocvBdumx > $DIR_SUP_SAUV/svg_oscar 2> $DIR_SUP_SAUV/svg_oscar.log
    compression oscar
   ;;
 *) : ;;
 esac

echo "Sauvegarde disque des fichiers ascii divers (type .fic)? (O/N) : \c"
read repons
case "$repons" in
  O|o|OUI|oui) echo "$nom_pg: debut sauvegarde disque fichiers ASCII textes
";
     find $meteor/DSK -print | egrep -v \
"/oscar|/espace|/bin|\.fap|\.seq|\.idx|\.dat|\.st|\.iu|\.w|/fico|/old|/new|
/TOTSEQ|/erresp|/errlem|/dech" |\
       cpio -ocvBdumx > $DIR_SUP_SAUV/svg_fic 2> $DIR_SUP_SAUV/svg_fic.log
       compression fic
   ;;
 *) :
   ;;
esac

echo "$nom_pg: verification creation & date des fichiers de sauvegarde
disque:"
echo "$nom_pg: (en general presence de 1 a 5 fichiers:"
echo "$nom_pg: svg_projet svg_seq svg_fap svg_oscar svg_fic)"
ls -l $DIR_SUP_SAUV/svg_*


echo "Pour restaurer ces donnees sauvegardee sur disque ->"
echo "Vous devez ensuite lancer le script de restauration disque : \n"
tput rev
echo "RESTO_DSK_BASE_PROJET \n"
tput sgr0
echo "situe dans le repertoire: /appli/projet/data/scripts \n"
echo "$nom_pg: Appelez maintenant le BT pour relancer la base, GOLD,
minitel"
echo "$nom_pg: et l'acces aux utilisateurs \n"
echo "$nom_pg: FIN NORMALE de la sauvegarde disque des donnees de PROJET"

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

41 41 Exemple de script de savegarde sur bande par « ufsdump » (SUN)

```
#@(#) Sauvegarde de tous les disques avec base stoppee, par 'ufsdump'.
# BL : 99/01/26

PATH=$PATH:/usr/sbin:/usr/bin:/usr/openwin/bin:/usr/local/bin:/etc:/usr/ccs
/bin:.
```

```
PATH=$PATH:/usr/sbin:/usr/bin:/usr/ucb:/logiciel/pgo:.:/usr/platform/sun4u/
sbin:
export PATH

DATE=`date '+%d/%m/%y %H:%M'`
LOG=/var/log/SAUV_TOT_AN2M.log`date +%w`

# arret base
echo "$DATE \n" > $LOG
#su - oracle -c "stop_bd ALUR" >> $LOG

cd /

for part in \
`df -k | grep -v Mounted|grep -v /proc|grep -v /dev/fd|grep -v swap|awk
'{print $6}'`
do
  echo "debut sauvegarde partition : $part"
  /usr/lib/fs/ufs/ufsdump 0uf /dev/rmt/0mn $part 2>> $LOG
  echo "fin sauvegarde partition $part : $?"
done

echo "\n\t ############################################################"
>>$LOG
DATE=`date '+%d/%m/%y %H:%M'`
echo "\n\n$DATE \n" >> $LOG

#relance base
#su - oracle -c "start_bd ALUR" >> $LOG

# rembobiner
mt -f /dev/rmt/0m rew

echo "fin de sauvegarde journaliere: `date`"

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

42 42 Exemple de script de savegarde sur bande par « fbackup »

```
#!/bin/ksh
#------------------------------------------------------------------------
---
# @(#) SAUVE_JOUR_BANDE
# Type / Type    : job shell (UNIX)
# But / Aim      : sauvegarde journaliere par fbackup avec arret des bases
ORACLE
# Auteur / Author: Benjamin LISAN  ->    date creation:  01-AUG-90
# Usage / Usage  : SAUVE_JOUR_BANDE
# utilise par le : cron
#------------------------------------------------------------------------
---
PATHORA=/logiciel/oracle

# arret des bases
for SID in `ps -ef|grep "ora_pmon"|grep -v grep|sed "s:.*ora_pmon_::"|grep
-v "::"`
do
      /usr/bin/su - oracle -c "$PATHORA/stop_bd $SID < /dev/null"
done

/usr/sam/lbin/br_backup DAT FULL Y /dev/rmt/0m /etc/sam/br/graphDCAa06056
root Y 1 N > \
        /var/tmp/SAM_br_msgs 2>&1 #sambackup

# relance des bases
for SID in `cat /etc/oratab | cut -f1 -d:`
do
      /usr/bin/su - oracle -c "$PATHORA/start_bd $SID < /dev/null"
done

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

43 43 Exemple de script de savegarde sur bande par « cpio » (SUN)

```
#!/bin/sh
# ------------------------------------------------------------
# script : SAUVE_JOUR_BANDE   Auteur : BL   Date : 20/08/98
# but : sauvegarde journaliere de la machine SUN SUN1 par cpio
# s donnees modifiees par les utilisateurs (sources...)
# ------------------------------------------------------------
# BL 21/10/99 : sauvegarde en mode compressee
# ------------------------------------------------------------
dat=`date '+ %Y%m%d'`
export dat
hr=`date '+ %H:%M'`
ficsav=/tmp/ls_fic_sav
logsauv=/var/adm/log/sauvj$$
echo "$dat$hr" > /tmp/pseudolabel

echo "debut sauveg.journ.du $dat a $hr sur `hostname`"| tee -a $logsauv
mt rew

# arret de toutes les bases ORACLE
su - oracle -c 'arret_toutes_bases'
```

```
cd /
ls /tmp/pseudolabel                                          >  $ficsav
ls /datapil_1/admin/login.txt                                >> $ficsav
find /pilot/version_encours/config                           >> $ficsav
find /datapil/data1/base???                                  >> $ficsav
find /etc/rc0.d                                              >> $ficsav
find /etc/rc1.d                                              >> $ficsav
find /etc/rc3.d                                              >> $ficsav
find /etc/rcS.d                                              >> $ficsav
find /usr/local/bin                                          >> $ficsav
# 12/10/98 : lignes rajoutees a la demande de Rene Goselin
find /intg/commun                                            >> $ficsav
find /intg/piloarc                                           >> $ficsav
# 20/07/99 : lignes rajoutees a la demande de Pierre Giron
find /intg/saphir/version_encours                            >> $ficsav
# 20/07/99 : lignes rajoutees a la demande de Joelle Melzassard
find /intg/capa/sources_V1.4                                 >> $ficsav
#find /intg/capa/include_V1.4                                 >> $ficsav
find /intg/capa/oracle_C                                     >> $ficsav
find /logiciel/oracle/oracle_C/create_tables                 >> $ficsav

find /logiciel/tuxedo | grep "tuxedo_" |grep -v "tuxedo_v5" |\
  grep -v "ULOG" | grep -v "ulog_pil"                        >> $ficsav

# 12/10/98 : lignes rajoutees a la demande de Alain Benichou
find /intg|grep "carn..."|grep -v "/exe/"|grep -v "/objets/"|\
  grep -v "/lib/"|grep -v ".a"|grep -v ".ln"|grep -v ".pure" >> $ficsav


sleep 2
# BL 21/10/99 : sauvegarde en mode compressee
cat $ficsav   | cpio -ocBdum > /dev/rmt/0c
Retour=$?

if [ "$Retour" = "0" ]
then
    echo "Fin Sauveg.`hostname` (Retour=$Retour) Relec.bande"|tee -a
$logsauv
    head -2 $logsauv
    sleep 5
    # BL 21/10/99 : sauvegarde en mode compressee
    cpio -icvB /tmp/pseudolabel  < /dev/rmt/0c
    mt rew
    echo "fin sauveg.jour.du $dat a $hr sur `hostname`"| tee -a $logsauv
else
    echo "Err. Sauveg.`hostname`: code retour=$Retour"|tee -a $logsauv
fi

# relance des bases ORACLE qui etaient lances avant la sauvegarde
su - oracle -c 'relance_des_bases'

# nettoyage des fichiers log
find /var/adm/log -name "sauvj*" -mtime +7 -exec rm {} \;

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>

#!/bin/sh
# ------------------------------------------------------------
# script : SAUVE_JOUR_CPIO_SUN1   Auteur : BL   Date : 20/08/98
# but : sauvegarde journaliere de la machine SUN SUN1 par cpio
```

```
# s donnees modifiees par les utilisateurs (sources...)
# ------------------------------------------------------------
# BL 12/10/98 : 2 lignes rajoutees a la demande de Rene Goselin
# BL 21/10/99 : sauvegarde en mode compressee
# ------------------------------------------------------------
dat=`date '+ %Y%m%d'`
export dat
hr=`date '+ %H:%M'`
ficsav=/tmp/ls_fic_sav
logsauv=/var/adm/log/sauvj$$
echo "$dat$hr" > /tmp/pseudolabel

echo "debut sauveg.journ.du $dat a $hr sur `uname -n`"| tee -a $logsauv
mt rew

# arret de toutes les bases ORACLE
# BL 12/1/2000 : en commentaire car suppres.ORACLE sur SUN1 le 4/1/2000
#su - oracle -c 'arret_toutes_bases'

cd /
ls /tmp/pseudolabel                                          >  $ficsav
ls /datapil_1/admin/login.txt                                >> $ficsav
find /pilot/version_encours/config                           >> $ficsav
find /datapil/data1/base???                                  >> $ficsav
find /etc/rc0.d                                              >> $ficsav
find /etc/rc1.d                                              >> $ficsav
find /etc/rc3.d                                              >> $ficsav
find /etc/rcS.d                                              >> $ficsav
find /usr/local/bin                                          >> $ficsav
# 12/10/98 : lignes rajoutees a la demande de Rene Goselin
find /intg/commun                                            >> $ficsav
find /intg/piloarc                                           >> $ficsav
# 20/07/99 : lignes rajoutees a la demande de Pierre Giron
find /intg/saphir/version_encours                            >> $ficsav
# 20/07/99 : lignes rajoutees a la demande de Joelle Melzassard
find /intg/capa/sources_V1.4                                 >> $ficsav
#find /intg/capa/include_V1.4                                 >> $ficsav
find /intg/capa/oracle_C                                     >> $ficsav
find /logiciel/oracle/oracle_C/create_tables                 >> $ficsav

find /logiciel/tuxedo | grep "tuxedo_" |grep -v "tuxedo_v5" |\
  grep -v "ULOG" | grep -v "ulog_pil"                        >> $ficsav

# 12/10/98 : lignes rajoutees a la demande de Alain Benichou
find /intg|grep "carn..."|grep -v "/exe/"|grep -v "/objets/"|\
 grep -v "/lib/"|grep -v "\.a"|grep -v "\.ln"|grep -v "\.pure" >> $ficsav


sleep 2
# BL 21/10/99 : sauvegarde en mode compressee
mt rew
Retour=$?
if [ "$Retour" != "0" ]
then
    echo "`uname -n` pb au rembob de la bande : $Retour"|tee -a $logsauv
fi
# debut de la sauvegarde sur bande
cat $ficsav   | cpio -ocBdum > /dev/rmt/0c
Retour=$?

if [ "$Retour" = "0" ]
```

```
then
    echo "Fin Sauveg.`uname -n` (Retour=$Retour) Relec.bande"|tee -a
$logsauv
    head -2 $logsauv
    sleep 5
    # BL 21/10/99 : sauvegarde en mode compressee
    cpio -icvBdum "/tmp/pseudolabel"  < /dev/rmt/0c
    dat=`date '+ %Y%m%d'`
    hr=`date '+ %H:%M'`
    echo "fin sauveg.jour.du $dat a $hr sur `uname -n`"| tee -a $logsauv
else
    echo "Err. Sauveg.`uname -n`: code retour=$Retour"|tee -a $logsauv
fi
mt rew

# relance des bases ORACLE qui etaient lances avant la sauvegarde
# BL 12/1/2000 : en commentaire car suppres.ORACLE sur SUN1 le 4/1/2000
#su - oracle -c 'relance_des_bases'

# nettoyage des fichiers log
find /var/adm/log -name "sauvj*" -mtime +7 -exec rm {} \;

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

44 44 Exemple de script de savegarde sur bande par « ufsdump » (SUN)

```
#!/sbin/sh
#@(#) Sauvegarde journalieres des disques contenant des datas
#-------------------------------------------------------------------------
---
# @(#) SAUVE_JOUR_UFS_SUN1
# Nom / Name     : SAUVE_JOUR_UFS_SUN1
# Type / Type    : job shell (UNIX)
# But / Aim      : sauvegarde journaliere par ufsdump avec arret bases
ORACLE
# Auteur / Author: Benjamin LISAN  date creation:  05/01/2000
#
# Usage / Usage  : SAUVE_JOUR_UFS_SUN1
# Exemple / Ex.  : "SAUVE_JOUR_UFS_SUN1"
# Contexte / Co. : cron
# Commentaires   : version pour Solaris 2.6 et > pour SUN1
# Modif. : BL: 11/01/99: on sauv.que les data
# Modif. : BL: 05/01/00: on change le systeme d'arret/relance des bases
#             on supprime les arret/relances par les commandes du type :
#             su - oracle -c "export ORACLE_SID=MIDV;stop_bd MIDV"|tee -a
$LOG
#-------------------------------------------------------------------------
---
#ajout BL:99/01/26:car manque les chemins pour lancer les cde unix tels
qu'uname...
PATH=$PATH:/usr/sbin:/usr/bin:/usr/openwin/bin:/usr/local/bin:/etc:/usr/ccs
/bin:.
PATH=$PATH:/users/pgo
export PATH

DATE=`date '+%d/%m/%y %H:%M'`
LOG=/users/manager/log/dump0.log`date +%w`
echo "$DATE debut sauvegarde \n" > $LOG

# arret de toutes les  bases ORACLE
```

```
su - oracle -c 'arret_toutes_bases'

# rembobiner
mt -f /dev/rmt/0m rew

# sauvegarde ufsdump
 /usr/sbin/ufsdump 0uf /dev/rmt/0c /dev/dsk/c0t2d0s0   1>> $LOG 2>> $LOG#
/datapil
 /usr/sbin/ufsdump 0uf /dev/rmt/0c /dev/dsk/c0t1d0s0   1>> $LOG 2>> $LOG#
/home
 /usr/sbin/ufsdump 0uf /dev/rmt/0c /dev/dsk/c0t11d0s7  1>> $LOG 2>> $LOG#
/intg
cat $LOG

mt -f /dev/rmt/0m rew

echo "\n\t ############################################################"
| tee -a $LOG
DATE=`date '+%d/%m/%y %H:%M'`
echo "\n\n$DATE fin sauvegarde \n" >>  $LOG

#relance de toutes les bases
su - oracle -c 'relance_des_bases'


echo "`basename $0` fin sauvegarde journaliere du `date '+%d/%m/%Y'`"

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

45 45 Exemple de script de savegarde de la partie système sur bande par « cpio » (SUN)

```
# ------------------------------------------------------------------------
# script : SAUVE_SYS_BANDE   Auteur : BL   Date : 20/08/98
# but : sauvegarde la partie systeme de la machine SUN SUN1 par cpio
# ------------------------------------------------------------------------
echo "Debut sauvegarde systeme du `date` sur `hostname` sur bande"
mt rew
find /dev      >  /tmp/ls_fic_to_sav
find /devices  >> /tmp/ls_fic_to_sav
find /etc      >> /tmp/ls_fic_to_sav
find /home     >> /tmp/ls_fic_to_sav
find /intg     >> /tmp/ls_fic_to_sav
find /kernel   >> /tmp/ls_fic_to_sav
find /logiciel >> /tmp/ls_fic_to_sav
find /opt      >> /tmp/ls_fic_to_sav
find /sbin     >> /tmp/ls_fic_to_sav
find /usr      >> /tmp/ls_fic_to_sav
find /var      >> /tmp/ls_fic_to_sav
cat /tmp/ls_fic_to_sav | cpio -ocvBdum > /dev/rmt/0m

echo "Test de relecture de la bande"
sleep 2
cpio -icvtB /etc/inittab < /dev/rmt/0m
echo "Fin sauvegarde systeme du `date` sur `hostname` sur bande"

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

46 46 Exemple de script de savegarde de la partie système sur bande par « cpio » (HP)

```
# ----------------------------------------------------------------------
# script : SAUVE_SYS_TAPE HP   Auteur : BL   Date : 20/08/98
# but : sauvegarde la partie systeme de la machine par cpio
# ----------------------------------------------------------------------
rm -f /tmp/ls_fic_a_sauv 2> /dev/null
mt rew
ls /.dtprofile >> /tmp/ls_fic_a_sauv
ls /.kshrc >> /tmp/ls_fic_a_sauv
ls /.profile >> /tmp/ls_fic_a_sauv
ls /.rhosts >> /tmp/ls_fic_a_sauv
ls /.sh_history >> /tmp/ls_fic_a_sauv
ls /.vueprofile >> /tmp/ls_fic_a_sauv
ls /.x11startlog >> /tmp/ls_fic_a_sauv

find /.softbench >> /tmp/ls_fic_a_sauv
find /.sw      >> /tmp/ls_fic_a_sauv
find /.vue     >> /tmp/ls_fic_a_sauv
find /dev      >> /tmp/ls_fic_a_sauv
find /etc      >> /tmp/ls_fic_a_sauv
find /extra    >> /tmp/ls_fic_a_sauv
find /home     >> /tmp/ls_fic_a_sauv
find /logiciel >> /tmp/ls_fic_a_sauv
find /opt      >> /tmp/ls_fic_a_sauv
find /sbin     >> /tmp/ls_fic_a_sauv
find /stand    >> /tmp/ls_fic_a_sauv
find /usr      >> /tmp/ls_fic_a_sauv
find /var      >> /tmp/ls_fic_a_sauv
echo "debut sauvegarde sur BANDE"
cat /tmp/ls_fic_a_sauv | cpio -ocvBdum > /dev/rmt/0m

echo "fin sauvegarde sur BANDE"
echo "Test de relecture de la bande"
sleep 2
cpio -icvtB /etc/* < /dev/rmt/0m

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

47 47 Autre exemple de script de savegarde de la partie système sur bande par « cpio » (SUN)

```
# ----------------------------------------------------------------------
# script : SAUVE_SYS_TAPE SUN1   Auteur : BL   Date : 20/08/98
# but : sauvegarde la partie systeme de la machine par cpio
# ----------------------------------------------------------------------
rm -f /tmp/ls_fic_to_sav 2> /dev/null
mt rew
echo "debut constitution fic liste rep/fic a sauver"
ls /.dtprofile >> /tmp/ls_fic_to_sav
ls /.kshrc >> /tmp/ls_fic_to_sav
ls /.profile >> /tmp/ls_fic_to_sav
ls /.rhosts >> /tmp/ls_fic_to_sav
ls /.sh_history >> /tmp/ls_fic_to_sav
ls /.x11startlog >> /tmp/ls_fic_to_sav
find /TT_DB    >> /tmp/ls_fic_to_sav
find /dev      >> /tmp/ls_fic_to_sav
find /devices  >> /tmp/ls_fic_to_sav
find /etc      >> /tmp/ls_fic_to_sav
```

```
find /home     >> /tmp/ls_fic_to_sav
find /kernel   >> /tmp/ls_fic_to_sav
find /logiciel >> /tmp/ls_fic_to_sav
find /mnt      >> /tmp/ls_fic_to_sav
find /net      >> /tmp/ls_fic_to_sav
find /opt      >> /tmp/ls_fic_to_sav
find /platform >> /tmp/ls_fic_to_sav
find /sbin     >> /tmp/ls_fic_to_sav
find /tmp      >> /tmp/ls_fic_to_sav
find /usr      >> /tmp/ls_fic_to_sav
find /users    >> /tmp/ls_fic_to_sav
find /var      >> /tmp/ls_fic_to_sav
find /vol      >> /tmp/ls_fic_to_sav
find /xfn      >> /tmp/ls_fic_to_sav
echo "fin constitution fic liste rep/fic a sauver"
echo "debut de la sauvegarde"
cat /tmp/ls_fic_to_sav | cpio -ocvBdum > /dev/rmt/0m
echo "\007"

echo "fin de la sauvegarde"
echo "Test de relecture de la bande"
sleep 2
cpio -icvtB /etc/* < /dev/rmt/0m

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

48 48 Exemple de script de savegarde de la partie système sur bande par « ufsdump » (SUN)

```
#!/sbin/sh
#---------------------------------------------------------------------------
---
# @(#) SAUVE_SYS_UFS
# Nom / Name     : SAUVE_SYS_UFS
# Type / Type    : job shell (UNIX)
# But / Aim      : faire une sauvegarde system par ufsdump
# Auteur / Author: Benjamin LISAN  date creation:  30 Dec 99
#
# Usage / Usage  : SAUVE_SYS_UFS
# Exemple / Ex.  : "SAUVE_SYS_UFS"
# Contexte / Co. : cron
# Commentaires   : version pour SUN SOLARIS
#---------------------------------------------------------------------------
---
log=/usr/local/divers/sauv_sys_ufs.log
mt rew
for dev in `df -k|grep -v /data_01|awk '{print $1}'|\
 grep dev| sed 's:dsk:rdsk:'`
do
   /usr/sbin/ufsdump 0uf /dev/rmt/0c $dev |tee -a $log
done
mt rew

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

49 49 Exemple de script de relance d’une base Oracle (on donnant son SID)

```
#!/sbin/sh
```

```
#-------------------------------------------------------------------------
---
# @(#) start_base
# Nom / Name     : start_base Oracle (pour SUN)
# Type / Type    : job shell (UNIX)
# But / Aim      : start la base avec un compte utilisateur different d
oracle
# Auteur / Author: Benjamin LISAN  ->    date creation:  01-AUG-90
# Usage / Usage  : start_base ORACLE_SID avec ORACLE_SID celui de la base
# Exemple / Ex.  : start_base pmtd
# Commentaires   : cet utilisateur doit avoir un groupe secondaire : dba
#                  version pour SUN Solaris 2.5 et >
#-------------------------------------------------------------------------
---
USAGE="Usage= $0 ORACLE_SID"
echo "\$# = $#"
case $# in
    0) echo $USAGE # > &2
         exit 1 ;;
esac
GRP=`/usr/xpg4/bin/id -ng`
case $GRP in
   dba) : ;;
   *) echo "vous devez etre du groupe dba (votre groupe actuel: $GRP)"
      exit 1 ;;
esac
HOME_COP=$HOME
HOME=/logiciel/oracle
cd $HOME
. /logiciel/oracle/oracle_util
/logiciel/oracle/start_bd $1
export HOME=$HOME_COP
cd $HOME

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

50 50 Exemple de script d'arrêt d'une base Oracle (on donnant son SID)

```
#!/sbin/sh
#-------------------------------------------------------------------------
---
# @(#) stop_base
# Nom / Name     : stop_base Oracle (pour SUN)
# Type / Type    : job shell (UNIX)
# But / Aim      : stop la base avec un compte utilisateur different d
oracle
# Auteur / Author: Benjamin LISAN  ->    date creation:  01-AUG-90
# Usage / Usage  : stop_base ORACLE_SID avec ORACLE_SID celui de la base
# Exemple / Ex.  : stop_base pmtd
# Commentaires   : cet utilisateur doit avoir un groupe secondaire : dba
#                  version pour SUN Solaris 2.5 et >
#-------------------------------------------------------------------------
---
USAGE="Usage= $0 ORACLE_SID"
echo "\$# = $#"
case $# in
    0) echo $USAGE # > &2
         exit 1 ;;
esac
GRP=`/usr/xpg4/bin/id -ng`
```

```
case $GRP in
   dba) : ;;
   *) echo "vous devez etre du groupe dba (votre groupe actuel: $GRP)"
      exit 1 ;;
esac
HOME_COP=$HOME
HOME=/logiciel/oracle
cd $HOME
. /logiciel/oracle/oracle_util
/logiciel/oracle/stop_bd $1
export HOME=$HOME_COP
cd $HOME

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

51 51 Script de surveillance de la mémoire (SUN)

```
#!/sbin/sh
#-------------------------------------------------------------------------
---
# @(#) surv_mem
# Nom / Name     :  surv_mem
# Type / Type    : job shell (UNIX)
# But / Aim      : surveillance de la memoire
# Auteur / Author: Benjamin LISAN  ->    date creation:  01/03/1999
# Usage / Usage  : surv_mem
# Exemple / Ex.  : surv_mem
# Contexte / Co. : cron
# Commentaires   : version pour Solaris 2.5 et >
#-------------------------------------------------------------------------
---
DAT="`date '+%y/%m/%d:%H:%M:%S'` \c"
test ! -d /var/adm/log && mkdir /var/adm/log # si le repertoire n'existe
pas on le cree
log=/var/adm/log/survmem.log

# pour eviter que le fichier log augmente trop
touch $log
if test "`wc -l $log| awk '{print $1}'`" -gt "1000"
then
  tail -1000 $log > /tmp/logmem$$
   mv /tmp/logmem$$ $log
fi

ps -eo comm,vsz,pmem,s,pid,pcpu,time,etime|grep exeAlimCarnets|grep -v
grep|\
#awk '{print $1" "$2" "$3" "$4" "$5" "$6" "$7" "$8}'| \
nawk '
BEGIN {vsztot=0;memtot=0;cputot=0};
      {
   if ( FNR > 0 )
      {
          system("echo \"`date '+%y/%m/%d:%H:%M:%S'` \\c\"");
          printf("cde: %s vsz:%s pmem:%s s:%s pid:%s pcpu:%s time:%s
etim:%s\n",$1,$2,$3,$4,$5,$6,$7,$8);

          vsztot=vsztot+$2;
          memtot=memtot+$3;
          cputot=cputot+$6;
      }
      };
  END {
   if ( FNR > 0 )
      {
         printf "vsz tot:%s ,pc mem tot:%s ,pc cpu tot:%s ,nb occur:%s,
free swp:",vsztot,memtot,cputot,FNR
         system("swap -l| grep -v free | cut -c42-");
      }
};
' >> $log
#fi

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

52 52 Script de surveillance de la swap (HP)

```
#!/sbin/sh
#-------------------------------------------------------------------------
---
# @(#) surv_mem_swap
# Nom / Name     : surv_mem_swap
# Type / Type    : job shell (UNIX)
# But / Aim      : alerte si memoire ou swap prim/secondaire > a 50 %
# Auteur / Author: Benjamin LISAN  ->    date creation:  01-AUG-90
# Usage / Usage  : surv_mem_swap
# Exemple / Ex.  : surv_mem_swap
# Contexte / Co. : cron
# Commentaires   : version pour HPUX 10.20
#-------------------------------------------------------------------------
---
i=0
for siz in `swapinfo -a|grep -v TYPE|grep -v Kb|grep -v reserve|awk '{print
$5}'|sed 's:%::'`
do
  test $i -eq 0 && test $siz -gt 50 && echo "swap primaire > a $siz %"
  test $i -eq 1 && test $siz -gt 50 && echo "swap secondaire > a $siz %"
  test $i -eq 2 && test $siz -gt 50 && echo "mem.ram occupee > a $siz %"
  #echo "i=$i , siz=$siz"
  i=`expr $i + 1`
done

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
# 2ieme variante du script ci-avant :

#!/sbin/sh
#-------------------------------------------------------------------------
---
# @(#) surv_mem_swap
# Nom / Name     : surv_mem_swap
# Type / Type    : job shell (UNIX)
# But / Aim      : alerte si memoire ou swap prim/secondaire > a 50 %
# Auteur / Author: Benjamin LISAN  ->    date creation:  01-AUG-90
# Usage / Usage  : surv_mem_swap
# Exemple / Ex.  : surv_mem_swap
# Contexte / Co. : cron
# Commentaires   : version pour HPUX 10.20
#-------------------------------------------------------------------------
---
i=0
for siz in `swapinfo -a|grep -v TYPE|grep -v Kb|grep -v reserve|\
awk '{print $5}'|sed 's:%::'`
do
 test $i -eq 0 && test $siz != '-' && test $siz -gt 50 && \
                                      echo "swap prim. > a $siz %"
 test $i -eq 1 && test $siz != '-' && test $siz -gt 50 && \
                                      echo "swap second.> a $siz %"
 test $i -eq 2 && test $siz != '-' && test $siz -gt 50 && \
                                      echo "occup.mem.ram > a $siz %"
 #echo "i=$i , siz=$siz"
 i=`expr $i + 1`
done

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

53 53 surveillance du non dépassement du nombre d'utilisateurs de la licence Oracle

```
#!/sbin/sh
#-------------------------------------------------------------------------
---
# @(#) surv_nb_usr_ora
# Nom / Name     : surv_nb_usr_ora
# Type / Type    : job shell (UNIX)
# But / Aim      : surveillance du non depassement du nbre utilisateurs max
#                  de la licence ORACLE sur ce serveur
# Auteur / Author: Benjamin LISAN  ->    date creation:  01-AUG-90
#
# Usage / Usage  : surv_nb_usr_ora
# Exemple / Ex.  : "surv_nb_usr_ora"
# Contexte / Co. : cron
# Commentaires   : version pour HPUX 10.20
#-------------------------------------------------------------------------
---
# constition du script extrayant le nbre de users max de la licence oracle
rm -f /tmp/sessions_max.sql 2> /dev/null
cat >> /tmp/sessions_max.sql << %%
connect internal
set termout off
spool /tmp/sessions_max.lst
select sessions_max xxxxx from v\$license;
spool off
exit
%%
# extraction du nbre de users max de la licence oracle
su - oracle -c 'svrmgrl command=@/tmp/sessions_max.sql' << %% > /dev/null
2> /dev/null
gbq
%%
# nombre d'utilisateurs max de la licence oracle du SRV
NB_MAX_USR_LIC_ORA=`tail -2 /tmp/sessions_max.lst | head -1|sed "s: ::g"`
# nombres utilisateurs connectes
NB_USER_ORA_ON_LINE=`ps -ef|grep ora |grep -i PROTOCOL |wc -l`
# comparaison entre les deux
if [ $NB_USER_ORA_ON_LINE -ge $NB_MAX_USR_LIC_ORA ]
then
   echo "nbre users on-line oracle $NB_USER_ORA_ON_LINE > nbre usrs max
licence ora $NB_MAX_USR_LIC_ORA"
fi

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

54 54 Script de purges des fichiers log de Lotus Note

```
#!/sbin/sh
#------------------------------------------------------------------------
---
# @(#) surv_notes_log
# Nom / Name     : surv_notes_log
# Type / Type    : job shell (UNIX)
# But / Aim      : supprimer tous les fichiers temporaires anciens de LOTUS
#                  (but : faire de la place sur la machine)
# Auteur / Author: Benjamin LISAN  ->    date creation:  01-AUG-90
#
# Usage / Usage  : surv_notes_log
# Exemple / Ex.  : "surv_notes_log"
# Contexte / Co. : cron
# Commentaires   : version pour machine SUN (en Solaris 2.6)
#------------------------------------------------------------------------
---
NOTES_LOG=/data_03/notes/tmp/notes.log
if [ -f $NOTES_LOG ]
then
    taille_notes_log=`ls -l $NOTES_LOG|awk '{print $5}'`
    if [ $taille_notes_log -gt 50000000 ]
    then
        msg="Taille fichier notes.log > a 50 Mo"
        echo $msg
        # envoi d'un email d'alerte vers l'admin.notes
        MAIL_ADDR=benjamin.lisan@toto.com
        HOST=`uname -n`
        mailx -s "$HOST `whoami`" $MAIL_ADDR << %%
$msg
%%
    fi
fi

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

55 55 scripts pour surveiller que les droits systèmes de certains fichiers sensibles ne changent pas (version pour HP)

```
#!/bin/sh
#------------------------------------------------------------------------
---
# @(#) surv_securite
# Nom / Name     : surv_securite
# Type / Type    : job shell (UNIX)
# But / Aim      : 1) surveillance securite, que fichiers systeme pas droit
#                     change
#                  2) pas de nouveaux fichiers avec droit suid, sgid
# Auteur / Author: Benjamin LISAN  ->    date creation:  15/5/00 V2
# Usage / Usage  : surv_securite
# Exemple / Ex.  : surv_securite
# Contexte / Co. : cron
# Commentaires   : version pour HPUX 10.20
#------------------------------------------------------------------------
---
DIR_LOG=/usr/local/divers
nom_prog=`basename $0`
test ! -d $DIR_LOG && mkdir $DIR_LOG
```

```
# si le fichiers historique precedent pas cree on le cree, sinon on cree un
nouveau
if test ! -f $DIR_LOG/drt_fic_sys.prec
then
   > $DIR_LOG/drt_fic_sys.prec # raz
   # on cherche les fichiers systeme importants
   ls -l /.rhosts           >> $DIR_LOG/drt_fic_sys.prec 2> /dev/null
   ls -l /.profile          >> $DIR_LOG/drt_fic_sys.prec 2> /dev/null
   ls -l /.kshrc            >> $DIR_LOG/drt_fic_sys.prec 2> /dev/null
   ls -l /etc/passwd        >> $DIR_LOG/drt_fic_sys.prec 2> /dev/null
   ls -l /etc/group         >> $DIR_LOG/drt_fic_sys.prec 2> /dev/null
   ls -l /etc/exports       >> $DIR_LOG/drt_fic_sys.prec 2> /dev/null
   ls -l /etc/fstab         >> $DIR_LOG/drt_fic_sys.prec 2> /dev/null
  #ls -l /etc/wtmp          >> $DIR_LOG/drt_fic_sys.prec 2> /dev/null
   ls -l /etc/hosts         >> $DIR_LOG/drt_fic_sys.prec 2> /dev/null
   ls -l /etc/hosts.equiv   >> $DIR_LOG/drt_fic_sys.prec 2> /dev/null
   ls -l /etc/syslog.conf   >> $DIR_LOG/drt_fic_sys.prec 2> /dev/null
   ls -l /etc/inetd.conf    >> $DIR_LOG/drt_fic_sys.prec 2> /dev/null
   ls -l /etc/inittab       >> $DIR_LOG/drt_fic_sys.prec 2> /dev/null
   ls -l /etc/profile       >> $DIR_LOG/drt_fic_sys.prec 2> /dev/null
   ls -l /etc/mail/sendmail.cf           >> $DIR_LOG/drt_fic_sys.prec 2>
/dev/null
   ls -l /usr/spool/cron/crontabs/root   >> $DIR_LOG/drt_fic_sys.prec 2>
/dev/null
   ls -l /var/adm/cron/cron.allow        >> $DIR_LOG/drt_fic_sys.prec 2>
/dev/null
   # on cherche les fichiers avec les droits suid ou sgid
   find / -type f \( -perm -2000 -o -perm 4000 \) -exec ls -l {} \; \
      >> $DIR_LOG/drt_fic_sys.prec 2> /dev/null
else
   > $DIR_LOG/drt_fic_sys.new # raz
   # on cherche les fichiers systeme importants
   ls -l /.rhosts           >> $DIR_LOG/drt_fic_sys.new 2> /dev/null
   ls -l /.profile          >> $DIR_LOG/drt_fic_sys.new 2> /dev/null
   ls -l /.kshrc            >> $DIR_LOG/drt_fic_sys.new 2> /dev/null
   ls -l /etc/passwd        >> $DIR_LOG/drt_fic_sys.new 2> /dev/null
   ls -l /etc/group         >> $DIR_LOG/drt_fic_sys.new 2> /dev/null
   ls -l /etc/exports       >> $DIR_LOG/drt_fic_sys.new 2> /dev/null
   ls -l /etc/fstab         >> $DIR_LOG/drt_fic_sys.new 2> /dev/null
  #ls -l /etc/wtmp          >> $DIR_LOG/drt_fic_sys.new 2> /dev/null
   ls -l /etc/hosts         >> $DIR_LOG/drt_fic_sys.new 2> /dev/null
   ls -l /etc/hosts.equiv   >> $DIR_LOG/drt_fic_sys.new 2> /dev/null
   ls -l /etc/syslog.conf   >> $DIR_LOG/drt_fic_sys.new 2> /dev/null
   ls -l /etc/inetd.conf    >> $DIR_LOG/drt_fic_sys.new 2> /dev/null
   ls -l /etc/inittab       >> $DIR_LOG/drt_fic_sys.new 2> /dev/null
   ls -l /etc/profile       >> $DIR_LOG/drt_fic_sys.new 2> /dev/null
   ls -l /etc/mail/sendmail.cf           >> $DIR_LOG/drt_fic_sys.new 2>
/dev/null
   ls -l /usr/spool/cron/crontabs/root   >> $DIR_LOG/drt_fic_sys.new 2>
/dev/null
   ls -l /var/adm/cron/cron.allow        >> $DIR_LOG/drt_fic_sys.new 2>
/dev/null
   # on cherche les fichiers avec les droits suid ou sgid
    find / -type f \( -perm -2000 -o -perm 4000 \) -exec ls -l {} \; \
      >> $DIR_LOG/drt_fic_sys.new 2> /dev/null
fi

# detection droits modifies

diff $DIR_LOG/drt_fic_sys.prec $DIR_LOG/drt_fic_sys.new \
  > $DIR_LOG/cmp_drt_fic_sys
```

```
# modif. droit des fichiers essentiels
> $DIR_LOG/maj_drt_fic # raz
cd /
test "`ls -l /usr/local/bin/$nom_prog | awk '{print $1}'`" != "-r--r-----"
\
    && ls -l /usr/local/bin/$nom_prog  >> $DIR_LOG/maj_drt_fic
test "`ls -l .rhosts     |awk '{print $1}'`" != "-r--r-----" \
    && ls -l .rhosts     >> $DIR_LOG/maj_drt_fic
test "`ls -l .profile    |awk '{print $1}'`" != "-r--r-----" \
    && ls -l .profile    >> $DIR_LOG/maj_drt_fic
test "`ls -l /etc/profile   |awk '{print $1}'`" != "-r--r--r--" \
    && ls -l /etc/profile   >> $DIR_LOG/maj_drt_fic
test "`ls -l /etc/passwd   |awk '{print $1}'`" != "-r--r--r--" \
    && ls -l /etc/passwd   >> $DIR_LOG/maj_drt_fic
test "`ls -l /etc/group    |awk '{print $1}'`" != "-r--r--r--" \
    && ls -l /etc/group    >> $DIR_LOG/maj_drt_fic
test "`ls -l /etc/syslog.conf  |awk '{print $1}'`" != "-r--r--r--" \
    && ls -l /etc/syslog.conf  >> $DIR_LOG/maj_drt_fic
test "`ls -l /etc/inetd.conf  |awk '{print $1}'`" != "-r--r--r--" \
    && ls -l /etc/inetd.conf  >> $DIR_LOG/maj_drt_fic
test "`ls -l /etc/hosts    |awk '{print $1}'`" != "-r--r--r--" \
    && ls -l /etc/hosts    >> $DIR_LOG/maj_drt_fic
test "`ls -l /etc/exports  |awk '{print $1}'`" != "-r--r--r--" \
    && ls -l /etc/exports  >> $DIR_LOG/maj_drt_fic
test "`ls -l /             |awk '{print $1}'`" != "drwxr-xr-x" \
    && ls -l /             >> $DIR_LOG/maj_drt_fic
test "`ls -l /etc          |awk '{print $1}'`" != "drwxr-xr-x" \
    && ls -l /etc          >> $DIR_LOG/maj_drt_fic
test "`ls -l /etc/mail     |awk '{print $1}'`" != "drwxr-xr-x" \
    && ls -l /etc/mail     >> $DIR_LOG/maj_drt_fic
test "`ls -l /stand        |awk '{print $1}'`" != "drwxr-xr-x" \
    && ls -l /stand        >> $DIR_LOG/maj_drt_fic
test "`ls -l /var          |awk '{print $1}'`" != "drwxr-xr-x" \
    && ls -l /var          >> $DIR_LOG/maj_drt_fic
test "`ls -l /usr          |awk '{print $1}'`" != "drwxr-xr-x" \
    && ls -l /usr          >> $DIR_LOG/maj_drt_fic
test "`ls -l /bin          |awk '{print $1}'`" != "drwxr-xr-x" \
    && ls -l /bin          >> $DIR_LOG/maj_drt_fic
test "`ls -l /usr/local/bin|awk '{print $1}'`" != "drwxr-xr-x" \
    && ls -l /usr/local/bin>> $DIR_LOG/maj_drt_fic

# envoi mail a l administrateur unix si ce fichier n est pas vide
if [ -s $DIR_LOG/cmp_drt_fic_sys ]
then

   echo "maj recente date fic sys"                    >
$DIR_LOG/cmp_drt_fic_sys2
   echo "ou maj/ajout recent fic avec droit suid sgid" >>
$DIR_LOG/cmp_drt_fic_sys2
   echo " " >> $DIR_LOG/cmp_drt_fic_sys2
   cat $DIR_LOG/cmp_drt_fic_sys                       >>
$DIR_LOG/cmp_drt_fic_sys2
   MAIL_ADDR=benjamin.lisan@renault.com
   HOST=`uname -n`
   mailx -s "$HOST pb_securite" $MAIL_ADDR < $DIR_LOG/cmp_drt_fic_sys2
fi

mv $DIR_LOG/drt_fic_sys.prec $DIR_LOG/drt_fic_sys.old
mv $DIR_LOG/drt_fic_sys.new  $DIR_LOG/drt_fic_sys.prec
```

#>>>
>>>>>>>

```
#!/bin/sh
# variante du script ci-avant
#-------------------------------------------------------------------------
---
# @(#) surv_securite
# Nom / Name     : surv_securite
# Type / Type    : job shell (UNIX)
# But / Aim      : 1) surveillance securite, que fichiers systeme pas droit
#                     change
#                  2) pas de nouveaux fichiers avec droit suid, sgid
# Auteur / Author: Benjamin LISAN  ->    date creation:  15/5/00
# Usage / Usage  : surv_securite
# Exemple / Ex.  : surv_securite
# Contexte / Co. : cron
# Commentaires   : version pour HPUX 10.20
#-------------------------------------------------------------------------
---
DIR_LOG=/usr/local/divers
test ! -d $DIR_LOG && mkdir $DIR_LOG
test ! -f $DIR_LOG/drt_fic_sys.prec &&
   find / -type f \( -perm -2000 -o -perm 4000 \) -exec ls -l {} \; \
      > $DIR_LOG/drt_fic_sys.prec

find / -type f \( -perm -2000 -o -perm 4000 \) -exec ls -l {} \; \
   > $DIR_LOG/drt_fic_sys.new

diff $DIR_LOG/drt_fic_sys.prec $DIR_LOG/drt_fic_sys.new \
  > $DIR_LOG/cmp_drt_fic_sys

# envoi mail a l administrateur unix si ce fichier n est pas vide
if [ -s $DIR_LOG/cmp_drt_fic_sys ]
then

   echo "maj recente date fic sys"                    >
$DIR_LOG/cmp_drt_fic_sys2
   echo "ou maj/ajout recent fic avec droit suid sgid" >>
$DIR_LOG/cmp_drt_fic_sys2
   echo " " >> $DIR_LOG/cmp_drt_fic_sys2
   cat $DIR_LOG/cmp_drt_fic_sys                       >>
$DIR_LOG/cmp_drt_fic_sys2
   MAIL_ADDR=benjamin.lisan@renault.com
   HOST=`uname -n`
   mailx -s "$HOST `whoami`" $MAIL_ADDR < $DIR_LOG/cmp_drt_fic_sys2
fi

mv $DIR_LOG/drt_fic_sys.prec $DIR_LOG/drt_fic_sys.old
mv $DIR_LOG/drt_fic_sys.new  $DIR_LOG/drt_fic_sys.prec

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

56 56 Script de surveillance de la swap (SUN)

```
#!/sbin/sh
#-------------------------------------------------------------------------
---
# @(#) surv_swap_sun
# Nom / Name     : surv_swap_sun
# Type / Type    : job shell (UNIX)
# But / Aim      : alerte si memoire ou swap prim/secondaire >= a 95 %
# Auteur / Author: Benjamin LISAN  ->    date creation:  01-AUG-90
# Usage / Usage  : surv_swap_sun
# Exemple / Ex.  : surv_swap_sun
# Contexte / Co. : cron
# Commentaires   : version pour Solaris 2.6
#-------------------------------------------------------------------------
---
si_swap_sup_90_then_mail()
{
    siz=$1
    echo "`date` swap >= a $siz %" | tee -a /tmp/swapfull$$
    echo "`date` swap >= a $siz %" >> /var/adm/log/survswap.log
    # reroutage du message d'alerte dans la mailbox de l'administrateur
UNIX
    MAIL_ADDR=benjamin.lisan-tmaservices@renault.com
    #MAIL_ADDR=root@mlbadm.mlb.renault.fr
    #MAIL_ADDR=Olivier.Cornic@renault.com
    test -f /tmp/swapfull$$ && mailx -s "`uname -n` swap_full" $MAIL_ADDR <
/tmp/swapfull$$
    rm -f /tmp/swapfull$$ 2> /dev/null
}

i=0
for siz in `swap -l|grep -v free | awk '{p=(1 - $5/$4)*100; print p}'`
do
  #echo  "siz= $siz i=$i"
  test $i -ge 0 && test $siz -gt 95 && si_swap_sup_90_then_mail $siz
  #echo "i=$i , siz=$siz"
  i=`expr $i + 1`
done

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

57 57 Script de surveillance de la saturation des disques (HP)

```
# test si capa des disques est pleine (version pour HP).

for `bdf|grep -v capacity| awk '{print $5}' | sed "s:%::"`
do
  if [ "$val" -gt "96" ]
  then
     val=`bdf|grep -v capacity| awk '{print $5}' | sed "s:%::"`
     part=`bdf|grep -v capacity| awk '{print $6}'`
     echo "partition $part > $val"
done

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

58 58 Exemple de script calculant le top 50 Mo des applications, les plus consommatrices (Sun)

```
# calcule le top 50 Mo des appl.les + consommatrices dans le repertoire
/appli

test -f /tmp/top_50_dsk && mv /tmp/top_50_dsk /tmp/top_50_dsk.old
cd /appli
echo "occupation disque de `pwd` sur `uname -n` sem.`date '+%W'` :" >>
/tmp/top_50_dsk
echo "-----------------------------------------"   >> /tmp/top_50_dsk
echo " "                                           >> /tmp/top_50_dsk

echo "app\ttaille Mo" > /tmp/top_50_dsk
echo "---\t---------" >> /tmp/top_50_dsk
du -s * |sort -rn| awk '{print $2 "\t" $1/2000 }' >> /tmp/top_50_dsk
cat /tmp/top_50_dsk
# reroutage du message d'alerte dans la mailbox de l'administrateur UNIX
MAIL_ADDR=benjamin.lisan@toto.com
test -f /tmp/top_50_dsk && mailx -s "`uname -n` `/usr/ucb/whoami`"
$MAIL_ADDR < /tmp/top_50_dsk
#rm -f /tmp/top_50_dsk 2> /dev/null

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
# variante du script ci-avant

# calcule le top 50 Mo des appl.les + consommatrices dans /intg
#
test -f /tmp/top_50_dsk && mv /tmp/top_50_dsk /tmp/top_50_dsk.old
test -f /tmp/top_50_app && mv /tmp/top_50_app /tmp/top_50_app.old
cd /intg

echo "occupation disque de `pwd` sur `uname -n` sem.`date '+%W'` :" |tee -a
/tmp/top_50_app
echo "-----------------------------------------"   |tee -a /tmp/top_50_app
echo " "                                           |tee -a /tmp/top_50_app
echo "appli\ttaille Mo"                         |tee -a /tmp/top_50_app
echo "-----\t---------"                         |tee -a /tmp/top_50_app

du -s * |sort -nr| awk '{print $2 "\t" int($1/2000) }'  >  /tmp/top_50_dsk
#cat /tmp/top_50_dsk

vol=0
for volcarn in `cat /tmp/top_50_dsk |grep carn| awk '{print int($2)}'`
do
   vol=`expr $volcarn + $vol`
done
echo "carnets\t$vol" |tee -a /tmp/top_50_app

vol=0
for volpilo in `cat /tmp/top_50_dsk |grep pilo| awk '{print int($2)}'`
do
   vol=`expr $volpilo + $vol`
done
echo "pilotag\t$vol" |tee -a /tmp/top_50_app

vol=0
for voldec in `cat /tmp/top_50_dsk |grep dec| awk '{print int($2)}'`
do
   vol=`expr $voldec + $vol`
```

```
done
echo "decoupe\t$vol" |tee -a /tmp/top_50_app

cat /tmp/top_50_dsk|grep -v pilo|grep -v carn|\
    grep -v dec|grep -v NULL| grep -v lost|grep -v oc|grep -v TT|\
    grep -v saphir2| grep -v divers| tee -a /tmp/top_50_app

# reroutage du message d'alerte dans la mailbox de l'administrateur UNIX
MAIL_ADDR=benjamin.lisan-tmaservices@renault.com
#MAIL_ADDR=root@mlbadm.mlb.renault.fr
#MAIL_ADDR=Olivier.Cornic@renault.com
test -f /tmp/top_50_app && mailx -s "`uname -n` top_50_place_disque"
$MAIL_ADDR < /tmp/top_50_app

#rm -f /tmp/top_50_dsk 2> /dev/null
#rm -f /tmp/top_50_app 2> /dev/null

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

59 59 script pour tester la connexion avec les machines listées dans le fichier hosts

```
# ping_ux : script pour tester la connexion avec les machines listees dans
le fichier hosts :
for mach in `cat /etc/hosts|grep -v ^$|grep -v "^#"|awk '{print $2}'`
do
  echo "$mach : \c"
  ping $mach 64 1 | head -1
done

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

60 60 script de test de la commande mailx sur une machine donnee

```
#@ testmail : test de la commande mailx sur une machine donnee

MAIL_ADDR=benjamin.lisan@toto.com
#mailx -s "`uname -n` `whoami`" $MAIL_ADDR < /tmp/ficmail
HOST=`uname -n`
mailx -s "$HOST `whoami`" $MAIL_ADDR << %%
test mailx sur $HOST
B.LISAN
%%

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

61 61 script de vérification du lancement du serveur http Netscape (sur Sun)

```
#!/sbin/sh
#-------------------------------------------------------------------------
---
# @(#) verif_app_http
# Nom / Name     : verif_app_http
# Type / Type    : job shell (UNIX)
# But / Aim      : verif presence appli. netscape serveur http
# Auteur / Author: Benjamin LISAN  ->    date creation: 25/5/00
#
```

```
# Usage / Usage  : verif_app_http
# Exemple / Ex.  : "verif_app_http"
# Contexte / Co. : cron
# Commentaires   : version pour machine SUN (et SOLARIS)
#-------------------------------------------------------------------------
---
PATH=/usr/bin:/usr/sbin:/sbin:/home/root:/usr/local/bin:.:$PATH
# liste des appli. netscape de la machine SUN
list_proc="./uxwdog ns-httpd"
nb_proc=`echo $list_proc|wc -w`
#for appli in `ls -ld /appli/ns*|sed "s:.*appli/ns::"`
for appli in `ls -d /logiciel/netscape/suitespot3.6/https-*|sed "s:.*https-
::"`
do
   cpt_proc=0
   for proc in `echo "$list_proc"`
   do
     ps -ef|grep $appli | grep $proc > /dev/null
     test $? = 0 && cpt_proc=`expr $cpt_proc + 1`
   done

   if [ $cpt_proc -eq $nb_proc ]
   then
      echo "lancement appli netscape http $appli : OK ($cpt_proc proc)"
   else
      rm -f  /tmp/lancehttp.log > /dev/null
      echo "lancement appli netscape http $appli : KO !!! ($cpt_proc
proc)"|tee -a /tmp/lancehttp.log
      MAIL_ADDR=benjamin.lisan-tmaservices@renault.com
      mailx -s "`uname -n` lancment_appli_nestscape" $MAIL_ADDR <
/tmp/lancehttp.log
   fi
done

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

62 62 script de vérification du lancement des bases Oracle (sur machine SUN)

```
#!/sbin/sh
#-------------------------------------------------------------------------
---
# @(#) verif_base_ora
# Nom / Name     : verif_base_ora
# Type / Type    : job shell (UNIX)
# But / Aim      : verif presence base oracle
# Auteur / Author: Benjamin LISAN  ->    date creation: 26/4/00
#
# Usage / Usage  : verif_base_ora
# Exemple / Ex.  : "verif_base_ora"
# Contexte / Co. : cron
# Commentaires   : version pour machine SUN (SOLARIS)
#-------------------------------------------------------------------------
---
PATH=/usr/bin:/usr/sbin:/sbin:/home/root:/usr/local/bin:.:$PATH
# liste des bases oracle de la machine SUN
# liste obtenue par la commande :
# ps -ef|grep ora_pmon|grep -v grep| sed "s/.*ora_pmon_//"|grep -v "//"
for SID in coac C barco # ALUR  : on a inhibe la base ALUR-WINCAP
do
```

```
   VERS=`grep "^$SID:" /var/opt/oracle/oratab|grep -v "#"|sed -e
"s:.*oracle_::" -e "s/:Y//" -e "s/:N//"`
   case $VERS in
   806)  # liste process residents base oracle
         list_proc="pmon dbw0 lgwr ckpt smon reco s000 d000 d001 d002"
         nb_proc=10 ;;
   734)  list_proc="dbwr pmon reco lgwr ckpt smon"
         nb_proc=6 ;;
   7162) list_proc="dbwr pmon lgwr ckpt smon"
         nb_proc=5 ;;
   *)    echo "base $SID non homologue VERS=$VERS"
         exit 1 ;;
   esac

   cpt_proc=0
   for proc in `echo "$list_proc"`
   do
     ps -ef|grep "ora_${proc}_${SID}" > /dev/null
     test $? = 0 && cpt_proc=`expr $cpt_proc + 1`
   done

   if [ $cpt_proc -eq $nb_proc ]
   then
      # ce test evite que le message soit affiche si le script est utilise
dans un cron
      #test `tty` && echo "lancement base oracle $SID (vers:$VERS) : OK
($cpt_proc proc)"
   else
      echo "lancement base oracle $SID (vers:$VERS) : KO !! ($cpt_proc
proc)"
      MAIL_ADDR=benjamin.lisan-tmaservices@renault.com
      mailx -s "`uname -n` `whoami`" $MAIL_ADDR << %%
base oracle $SID non lancee !!
%%
   fi
done

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

63 63 script de vérification du lancement des bases Oracle (sur machine HP)

```
#!/sbin/sh
#------------------------------------------------------------------------
---
# @(#) verif_base_ora
# Nom / Name     : verif_base_ora
# Type / Type    : job shell (UNIX)
# But / Aim      : verif presence base oracle
# Auteur / Author: Benjamin LISAN  ->    date creation: 26/4/00
#
# Usage / Usage  : verif_base_ora
# Exemple / Ex.  : "verif_base_ora"
# Contexte / Co. : cron
# Commentaires   : version pour machine HP (HPUX)
#------------------------------------------------------------------------
---
PATH=/usr/bin:/usr/sbin:/sbin:/home/root:/usr/local/bin:.:$PATH
# liste des bases oracle de la machine HP
# liste obtenue par la commande :
# ps -ef|grep ora_pmon|grep -v grep| sed "s/.*ora_pmon_//"|grep -v "//"
```

```
for SID in annu gemd sumeq gemr gemo siam ciraf orus PAM barco gbq c str
do
   cpt_proc=0
   VERS=`grep "^$SID:" /etc/oratab|grep -v "#"|sed -e "s:.*oracle_::" -e
"s/:Y//" -e "s/:N//"`
   case $VERS in
   806)  # liste process residents base oracle
         list_proc="pmon dbw0 lgwr ckpt smon reco s000 d000 d001 d002"
         nb_proc=10 ;;
   734)  list_proc="dbwr pmon reco lgwr ckpt smon"
         nb_proc=6 ;;
   7162) list_proc="dbwr pmon lgwr ckpt smon"
         nb_proc=5 ;;
   *)    echo "base $SID non homologue VERS=$VERS"
         exit 1 ;;
   esac

   for proc in `echo "$list_proc"`

   do
     ps -ef|grep "ora_${proc}_${SID}" > /dev/null
     test $? -eq 0 && cpt_proc=`expr $cpt_proc + 1`
   done
   #echo "nb_proc=$nb_proc VERS=$VERS cpt_proc=$cpt_proc"
   if [ $cpt_proc -eq $nb_proc ]
   then
      echo "lancement base oracle $SID (vers:$VERS) : OK ($cpt_proc proc)"
      # ce test evite que le message soit affiche si le script est utilise
dans un cron
      #test `tty` && echo "lancement base oracle $SID (vers:$VERS) : OK
($cpt_proc proc)"
   else
      echo "lancement base oracle $SID (vers:$VERS) : KO !! ($cpt_proc
proc)"
      MAIL_ADDR=benjamin.lisan-tmaservices@renault.com
      mailx -s "`uname -n` `whoami`" $MAIL_ADDR << %%
base oracle $SID non lancee !!
%%
   fi
done

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

64 64 script de vérification des processus occupant trop de mémoire et cpu (sur HP)

```
#!/sbin/sh
#-------------------------------------------------------------------------
---
# @(#) verif_occup_cpu_hp
# Nom / Name     :  verif_occup_cpu_hp
# Type / Type    : job shell (UNIX)
# But / Aim      : recherche ce qui occupe trop de cpu ou trop de memoire
#                  ou qui a trop d'occurence du meme job en memoire
# Auteur / Author: Benjamin LISAN  ->    date creation:  01/03/1999
# Usage / Usage  : verif_occup_cpu_hp
# Exemple / Ex.  : verif_occup_cpu_hp
# Contexte / Co. : cron
# Commentaires   : version pour HP 10.20 et >
```

```
#-------------------------------------------------------------------------
---
DAT="`date '+%y/%m/%d:%H:%M:%S'` \c"

vers=`uname -a | awk '{print $3}' | awk -F"." '{print $2}'`
if [ "$vers" = "09" -o "$vers" = "9" ]
then
    test ! -d /usr/adm/log && mkdir /usr/adm/log # si le repertoire
n'existe pas on le cree
    log=/usr/adm/log/surv.log; export log
    UNIX95=XPG3; export UNIX95
else
    test ! -d /var/adm/log && mkdir /var/adm/log # si le repertoire
n'existe pas on le cree
    log=/var/adm/log/surv.log; export log
    UNIX95=XPG4; export UNIX95
fi
# pour eviter que le fichier log augmente trop
tail -1000 $log > /tmp/logcpu$$
mv /tmp/logcpu$$ $log

if [ "$vers" = "10" ]
then
   ps -eo cpu,comm,ruser,etime,vsz,pid,state|sort -r -k 1,1|grep -v
COMMAND|\
   #awk '{print $1" "$2" "$3" "$4" "$5" "$6" "$7}'| \
   awk '
   {
    # si le % cpu > 100 ou si memoire virtuelle > 200 Mo alors alerte
    if ((int($1) > 100) || (int($5) > 200000))
       {
        system("echo \"`date '+%y/%m/%d:%H:%M:%S'` \\c\"");
        printf "cde: %s usr:%s pCPU:%s elapse:%s vsz:%s pid:%s st:%s\n",\
              $2,$3,$1,$4,$5,$6,$7
       }
   }
   ' | tee -a $log

   # si nombre occurence du meme programme depasse 20 alors alerte
   for cmd in `ps -C comm|grep -v COMM| grep -v "sh"|grep -v "^$"|sort -u`
   do
      #echo "cmd: $cmd"
      nb_occur=`ps -eo comm|grep -v COMM| grep -v "sh"| grep "$cmd"| wc -l`
      if test $nb_occur -gt 20
      then
           echo "$DAT: $cmd nbre occurence: $nb_occur" | tee -a $log
      fi
   done
fi

# si nombre moyens processus par mn > a 5 alors alerte
tx_cpu=`uptime | sed "s/.*load average: //"|awk '{print $3}'`
if test $tx_cpu -gt 5
then
     echo "$DAT: nb proc. par minute: $tx_cpu" | tee -a $log
fi

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

65 65 script de vérification des processus occupant trop de mémoire et cpu (sur SUN)

```
#!/sbin/sh
#-------------------------------------------------------------------------
---
# @(#) verif_occup_cpu_sun
# Nom / Name     :  verif_occup_cpu_sun
# Type / Type    : job shell (UNIX)
# But / Aim      : recherche ce qui occupe trop de cpu ou trop de memoire
#                  ou qui a trop d'occurence du meme job en memoire
# Auteur / Author: Benjamin LISAN  ->    date creation:  01/03/1999
# Usage / Usage  : verif_occup_cpu_sun
# Exemple / Ex.  : verif_occup_cpu_sun
# Contexte / Co. : cron
# Commentaires   : version pour Solaris 2.5 et >
#-------------------------------------------------------------------------
---
DAT="`date '+%y/%m/%d:%H:%M:%S'` \c"
test ! -d /var/adm/log && mkdir /var/adm/log # si le repertoire n'existe
pas on le cree
log=/var/adm/log/surv.log

# pour eviter que le fichier log augmente trop
tail -1000 $log > /tmp/logcpu$$
mv /tmp/logcpu$$ $log

ps -eo pcpu,comm,ruser,etime,vsz,pid,pmem,s|sort -r -k 1,1|grep -v COMM|\
#awk '{print $1" "$2" "$3" "$4" "$5" "$6" "$7" "$8}'| \
nawk '
{
 # si le % cpu > 40 ou si memoire virtuelle > 250 Mo alors alerte
 if ( (int($1) > 40) || (int($5) > 250000) || (int($7) > 20) )
    {
     system("echo \"`date '+%y/%m/%d:%H:%M:%S'` \\c\"");
     printf "cde: %s usr:%s pCPU:%s elap:%s vsz:%s pid:%s pmem:%s
st:%s\n",\
           $2,$3,$1,$4,$5,$6,$7,$8
    }
}
' | tee -a $log
# si nombre occurence du meme programme (non carnet) depasse 20 alors
alerte
for cmd in \
`ps -eo comm|grep -v COMM|grep -v sh|grep -v term|grep -v smbd|grep -v
login|grep -v "^$"|sort -u`
do
   #echo "cmd: $cmd"
   nb_occur=`ps -eo comm|grep -v COMM| grep -v "sh"| grep "$cmd"| wc -l`
   if test $nb_occur -gt 20 -a "$cmd" -ne "exeCarnets"
   then
        echo "$DAT:$cmd nbre occurence: $nb_occur" | tee -a $log
   fi
done

# si nombre moyens processus par mn > a 5 alors alerte
tx_cpu=`uptime | sed "s:,::g" | awk '{print $10}'`
if test $tx_cpu -gt 5
then
     echo "$DAT: nb proc. par minute: $tx_cpu" | tee -a $log
fi
```

#>>>
   >>>>>>

66 66 script de vérification de la non saturation des partitions (sur HP)

```
#!/sbin/sh
#-----------------------------------------------------------------------------
---
# @(#) verif_parti_full_hp
# Nom / Name     : verif_parti_full_hp
# Type / Type    : job shell (UNIX)
# But / Aim      : faire une alerte si la partition est occupee a + de 95%
# Auteur / Author: Benjamin LISAN  ->    date creation:  01-AUG-90
# Usage / Usage  : verif_parti_full_hp
# Exemple / Ex.  : verif_parti_full_hp
# Contexte utili.: cron
# Comment.       : version pour HPUX 10.20
# avec reroutage du message d'alerte dans la mailbox de l'administrateur
UNIX
#-----------------------------------------------------------------------------
---
rm -f  /tmp/partfull$$ 2> /dev/null
for part in `df -vl |  grep -v "^     " | awk '{print $1}'`
do
    siz=`df -vl| grep -v "^      "|  grep "$part "| awk '{print $5}'`
    #echo "$part $siz"
    if [ $siz -ge 90 ]
    then
        echo "attention occupation partition $part a $siz %"| tee -a
/tmp/partfull$$
    fi
done
# reroutage du message d'alerte dans la mailbox de l'administrateur UNIX
MAIL_ADDR=benjamin.lisan@toto.com
test -f /tmp/partfull$$ && mailx -s "`uname -n` `/usr/bin/whoami`"
$MAIL_ADDR < /tmp/partfull$$
rm -f  /tmp/partfull$$ 2> /dev/null
```

#>>>
   >>>>>>

67 67 script de vérification de la non saturation des partitions (sur SUN)

```
#!/sbin/sh
#-----------------------------------------------------------------------------
---
# @(#) verif_parti_full_sun
# Nom / Name     : verif_parti_full_sun
# Type / Type    : job shell (UNIX SUN)
# But / Aim      : faire une alerte si la partition est occupee a + de 95%
# Auteur / Author: Benjamin LISAN date creation:  1/6/98
# Usage / Usage  : verif_parti_full_sun
# Exemple / Ex.  : verif_parti_full_sun
# Contexte utili.: cron
# Comment.       : version pour Solaris 2.5/2.6...
# avec reroutage du message d'alerte dans la mailbox de l'administrateur
UNIX
#-----------------------------------------------------------------------------
---
rm -f /tmp/partfull$$ 2> /dev/null
```

```
for part in \
`df -k|grep -v "^/proc"|grep -v datacar|grep -v kbytes|grep -v "^fd"|\
grep -vi cdrom|grep -v /datagpi_0|awk '{print $6}'`
do
    siz=`df -k|grep "$part\$" | awk '{print $5}' | sed -e 's:%::'`
    #echo "<<$part>> <<$siz>>"
    if test $siz -gt 95
    then
        echo "attention occupation partition $part a $siz %" | tee -a
/tmp/partfull$$
    fi
done

# reroutage du message d'alerte dans la mailbox de l'administrateur UNIX
MAIL_ADDR=benjamin.lisan-tmaservices@renault.com
#MAIL_ADDR=root@mlbadm.mlb.renault.fr
#MAIL_ADDR=Olivier.Cornic@renault.com
test -f /tmp/partfull$$ && \
  mailx -s "`uname -n` partition_full" $MAIL_ADDR < /tmp/partfull$$
rm -f /tmp/partfull$$ 2> /dev/null

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

68 68 Exemple de script de vérification du lancement du serveurs Lotus Notes (sur SUN)

```
#!/sbin/sh
#-------------------------------------------------------------------------
---
# @(#) verif_app_lotus
# Nom / Name     : verif_app_lotus (verify du lancement des appli et demons
'notes')
# Type / Type    : job shell (UNIX)
# But / Aim      : verif presence appli. notes serveur lotus
# Auteur / Author: Benjamin LISAN  ->    date creation: 25/5/00
#
# Usage / Usage  : verif_app_lotus
# Exemple / Ex.  : "verif_app_lotus"
# Contexte / Co. : cron
# Commentaires   : version pour machine SUN (SOLARIS)
#-------------------------------------------------------------------------
---
PATH=/usr/bin:/usr/sbin:/sbin:/home/root:/usr/local/bin:.:$PATH
# liste des appli. notes de la machine JAVA
rm -f /tmp/proc_notes.lis > /dev/null
list_proc="/opt/lotus/notes/latest/sunspa/replica
/opt/lotus/notes/latest/sunspa/router
/usr/bin/sh /logiciel/lotus/bin/server
/opt/lotus/notes/latest/sunspa/server
/opt/lotus/notes/latest/sunspa/update
/opt/lotus/notes/latest/sunspa/stats
/opt/lotus/notes/latest/sunspa/amgr
/opt/lotus/notes/latest/sunspa/amgr
/opt/lotus/notes/latest/sunspa/adminp
/opt/lotus/notes/latest/sunspa/sched
/opt/lotus/notes/latest/sunspa/calconn
/opt/lotus/notes/latest/sunspa/update
/opt/lotus/notes/latest/sunspa/stats
/opt/lotus/notes/latest/sunspa/lotus"
nb_proc=`echo $list_proc|wc -w`
cpt_proc=0
for proc in `echo "$list_proc"`
do
     #echo "\"=== $proc ==\""
     ps -ef|grep "$proc" > /dev/null
     test $? = 0 && cpt_proc=`expr $cpt_proc + 1`
done

if [ $cpt_proc -eq $nb_proc ]
then
      echo "lancement appli notes lotus $appli : OK ($cpt_proc proc)"
else
      rm -f  /tmp/lancelotus.log > /dev/null
      echo "lancement appli notes lotus $appli : KO !!! ($cpt_proc
proc)"|tee -a /tmp/lancelotus.log
      MAIL_ADDR=benjamin.lisan-tmaservices@renault.com
      mailx -s "`uname -n` lancment_serveur_notes" $MAIL_ADDR <
/tmp/lancelotus.log
fi

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

69 69 script permettant de lancer à distance l'outil de gestion de configuration « PVCS »

```
# vm2 : script permettant de lancer a distance l outil de gestion de config
PVCS
# sur SUMATRA en conservant le meme groupe que sur
# la machine UNIX ou l on est.  Dev de BL 1 juin 99
GRP=`id -ng`;export GRP
remsh sumatra "(
newgrp $GRP << FIN
# ici adresse ip du PC de l utilisateur B LISAN
export DISPLAY=172.25.49.133:0.0;
/logiciel/pvcs/vm/pvcsvmux
FIN
)"

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

70 70 Exemple de script de calcul du volume des donnees des applications sur une machine HP

```
# ------------------------------------------------------------------------
#
# Projet        : FACTURATION UNIX   Sous-Projet  :
#
# Auteur        : B. LISAN           Date creation: 03/08/95  ..:..
#
# Nom programme : vol_appli          Type Langage : Bourne SHELL
#
# Objet du trait:                    Version      : 1.0
#
# Calcul du volume des donnees des applications sur une machine HP
#
# ------------------------------------------------------------------------
#
#echo "Calcul du volume des donnees des applications"
#echo "--------------------------------------------"
#echo " "
#vol_tot_tt_appli=0
rm /tmp/ls_appli /tmp/appli.vol 2> /dev/null

# constitution d'un fichier listant les repertoires des applications ORACLE

cd /appli
ls /appli | egrep -v \
"best1|lost+found|peliunix|natural.old|svgbase|oracle|oratst|tst|exploit|to
to|tr
ansfert"\
 > /tmp/ls_appli

# calcul taille appli. CLIENT EFFECTIF FACTURE_UX
# FOURNISSUR IMPACT INFOLOG STOCK PISTE RER RISQUE SAGE SEARCH
# SIEL STATISTIQUES etc ...

for nom_appli in `cat /tmp/ls_appli`
do
    cd /appli/$nom_appli
    vol_appli=`du -s /appli/$nom_appli 2> /dev/null | awk '{print $1}'; `
    vol_esp=`expr $vol_appli \* 512`
```

```
    #echo "Volume des donnees en Mo (on divise par 1048576): $vol_esp"

    vol_appli=`expr $vol_esp / 1048576`

    #echo "volume de tout le repertoire espace en octets : $vol_appli"

    echo "$nom_appli\t$vol_esp" | tee -a /tmp/appli.vol
    #echo "$nom_appli $vol_esp" #>> /appli/factux/D/log/vol_appli.res

    #vol_tot_tt_appli=`expr $vol_tot + $vol_appli`
done
echo "Le resultat des tailles en Mo des appli est dans fichier
/tmp/appli.vol"
echo "fin du programme $0 \007"

# ------------------------------------------------------------------------
---
# Nom procedure: extr_sid
# ------------------------------------------------------------------------
---
# setting des variables ORACLE
. /etc/opt/var.d/oracle.env
FIC_SID=$ORACLE_HOME/network/admin/tnsnames.ora
FIC_OUT=/tmp/appli_sid
rm $FIC_OUT 2> /dev/null

# extraction des donnees oracle
nb_ligne=0
for ligne in `awk '/\.world = /||/SID/ {print $1 $3}' $FIC_SID`
do
     mod=`expr $nb_ligne % 2`
     if [ $mod = 0 ]
     then
          appli=`echo $ligne | sed -e "s:\.world::"`
     else
          data_SID=`echo $ligne | sed -e "s:(SID::" -e "s:)::"`
          echo "$appli\t$data_SID"
          echo "$appli\t$data_SID" > $FIC_OUT
     fi
     nb_ligne=`expr $nb_ligne + 1`
done
grep "^[FTXZ][0-9]" /etc/passwd|awk -F: '{print $1}'| grep -v "Z000004" | \
sort > /tmp/USER_$hostname

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>>
```

71 71 Script de vérification de la taille des fichiers de contrôle Oracle

```
#@verify_ora_sys

vol_ora=0
for fic in `ls /data*/oracle_*/*.dbf /data*/oracle_*/*.rdo
/data*/oracle_*/*.ctl`
do
   vol_dbf=`ls -l $fic | awk '{print $5}'`
   vol_ora=`expr $vol_ora + $vol_dbf`
   vol_ora2=`expr $vol_ora / 1048576`
done
echo "volume total base oracle en Mo `uname -n`: $vol_ora2"
```

#>>
>>>>>>

72 72 Script de vérification de la taille restante des volumes disque sur une machine HP

```
# verify volume restant HP

i=0
for vol in `bdf| grep -v kby| awk '{print $4}' `
do
  i=`expr $i + $vol`
done
echo "volume restant: $i"
for vol in `bdf| grep -v kby| awk '{print $2}' `
do
  i=`expr $i + $vol`
done
echo "volume disques: $i"
for vol in `bdf| grep -v kby| awk '{print $3}' `
do
  i=`expr $i + $vol`
done
echo "volume utilise: $i"

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>
```

73 73 Script de vérification de la taille restante des volumes disque sur une machine Sun

```
# verify volume restant SUN

i=0
for vol in `df -k | grep -v fd | grep -v proc| grep -v kby|grep -v swap|
awk '{print $4}' `
do
  i=`expr $i + $vol`
done
echo "volume restant: $i"
for vol in `df -k | grep -v fd | grep -v proc| grep -v kby|grep -v swap|
awk '{print $2}' `
do
  i=`expr $i + $vol`
done
echo "volume disques: $i"
for vol in `df -k | grep -v fd | grep -v proc| grep -v kby|grep -v swap|
awk '{print $3}' `
do
  i=`expr $i + $vol`
done
echo "volume utilise: $i"

#>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
>>>>>>
```

74 74 Script de vérification du fonctionnement d’un processus (utilisable par le « cron »)

```
#!/bin/sh
# Rev 1.0
# proc_check - vérifie si un processus fonctionne, envoie le statut
# à l'utilisateur principal et tente de redémarrer le processus s'il
# ne fonctionne pas.
#
# usage: proc_check <proc_to_check_as_reported_by_ps_command>

# Modifie la variable MAILTO en conséquence

MAILTO="root"
PROC_TO_CHECK= $1
OPENWINHOME=/usr/openwin

CHECK_USAGE()
{
if [ "X$1" = "X" ]
   then
      echo "usage: $0 <proc_name_from_ps>"
      exit 1
   fi
}

FETCH_PID()
{
 pid=`/usr/bin/ps -e |
      /usr/bin/grep -w $1 |
      /usr/bin/sed -e 's/^  *//' -e 's/ .*//`
}

MAIL_GOOD_NEWS()
{
echo "${PROC_TO_CHECK} on "uname -n" IS running at "date"" \
| mailx -s" ${PROC_TO_CHECK} on "uname -n" \
IS running at "date"" $MAILTO
}

RESTART_PROCESS()
{
${OPENWINHOME}/bin/$1 &
}

MAIL_BAD_NEWS()
{
écho "${PROC_TO_CHECK} on "uname -n" NOT running at "date"" \
| mailx -s" ${PROC_TO_CHECK} on "uname -n" \
NOT running at "date"" $MAILTO

    RESTART_PROCESS ${PROC_TO_CHECK}
    sleep 10
    CHECK_IF_RUNNING
}

CHECK_IF_RUNNING()
{
    FETCH_PID   ${PROC_TO_CHECK}
    # echo $pid
    if [ "${pid}" = "" ]
    then
       MAIL_BAD_NEWS
    else
```

```
        MAIL_GOOD_NEWS
    fi
}

# main - it all starts here

CHECK_USAGE $1
CHECK IF RUNNING
```