Nouvelle supervision
HP Openview IT/Operations
Juin 2001

Fichier	Supervision ITO 2001.doc
Format	Word 95
Emetteur	Jean Philippe BOCQUENET – DTSI/IT/ISPE – Support ITO
Statut	*Ecriture*
Objet	Norme de supervision des systèmes ouverts par ITO.
Mots clés	Application, ITO, supervision, consignes
Acteurs	DTSI, DTM, DSIs
Couverture	Ce document concerne tous les serveurs Unix, exploités par la DTSI.
Approbation	?

Version	**Date**	**Raison(s) de la diffusion**	**Auteurs**
0.1 alpha	15/12/2000	Ecriture de la norme	Jean Philippe BOCQUENET
1.0	22/05/2001	Définition des normes, responsabilités	Sébastien GISLAIN
1.1	05/07/2001	Source du script verif_app_x.sh	Jean Philippe BOCQUENET

1 Domaines

Ce document décrit les nouvelles régles à mettre en place afin de réaliser le pilotage et la supervision des systèmes d'exploitation, des systèmes, des produits et des applications. Il couvre l'ensemble des domaines de supervision : gestion Unix/NT, IAO/Calcul et Usines.

2 Constat

La supervision mise en place actuellement couvre de plus en plus d'éléments disparates. Les produits et les applications étant nombreux et complexes, il devient inadéquate de faire suivre ces évolutions par les équipes ITO (support, intégration et exploitation).

3 Objectifs

Dans le but, de simplifier et d'améliorer la qualité de la supervision, il paraît indispensable de mettre en place des procédures permettant de statuer du bon fonctionnement d'un système d'exploitation, d'un produit ou d'une application. Ces procédures pourront être appelées par ITO (en mode batch) ou par les opérateurs (en mode interactif).

De plus, il est important de spécifier à nouveau le format des fichiers de logs que les applications peuvent créer. Ces fichiers sont surveillés par ITO afin de détecter tout dysfonctionnements.

4 Définitions

Dans ce chapitre sont décrits les termes génériques utilisés dans ce document.

4.1 Supervision ITO

La supervision d'un objet par ITO peut se faire :

- par un script, réalisant différentes vérifications : présence de processus, requête d'interrogation, présence de fichiers, … : appelé "Monitor".
- par la recherche de chaîne de caractères dans des fichiers de logs : appelé "Logfile".
- par la réception de messages directement émis par l'objet ou bien des procédures de gestion de cet objet (démarrage, arrêt, …) : appelé "Message".

4.2 Système d'information (SI)

Un SI est composé d'un ensemble de serveurs sur lesquels fonctionnent une ou plusieurs applications. Une application est constituée d'un ensemble d'objets :

- un système d'exploitation,
- des produits systèmes (NFS, sendmail, …),
- des produits (Oracle, Tuxedo, CFT, …),
- des développements spécifiques,
- …

Chaque élément de cette application pourront être surveillés au travers de scripts et de fichiers de logs.

Remarque La supervision au sens ITO est celle d'un serveur physique. En effet chaque alerte se rapporte à celui-ci. Dans ITO, on ne peut pas envisager facilement la supervision d'un SI, au sens global du terme, mais uniquement sur un des serveurs physique de ce SI. La supervision global du SI sera donc un ensemble d'alertes caractérisant son dysfonctionnement. La visualisation globale (appelé hypervision) fait actuellement l'objet d'un projet.

4.3 Objet

Un objet est un élément constitutif d'un système d'information. Cela peut être un système d'exploitation, une ressource de ce système, un produit, une application, …

4.4 Application

Il faut faire une différence entre les développements spécifiques qui sont la partie caractéristique de l'application et l'ensemble des objets qui constitue le système d'information et par extension l'application.

4.5 Instance

Une instance est un ensemble de processus d'un produit présent sur un serveur. Plusieurs instances d'un même produit peuvent être présent sur un même serveur. Chacune de ces instances sera associée à une ou plusieurs applications.

4.6 Fichier de LOCK

Un fichier de LOCK est un flag. Il est associé à une application ou une instance d'un produit. Il est créé par la procédure "standard" de démarrage du produit/application concerné. Il est supprimé par la procédure "standard" d'arrêt produit/application concerné.

Il permet de définir par sa présence que le produit/application associé doit fonctionner correctement.

Il permet de définir par son absence que le produit/application associé ne fonctionne pas à cet instant. Le produit/application n'a donc pas besoin d'être vérifié.

4.7 Fichier de logs

Un fichier de logs contient des informations de fonctionnement d'un système d'exploitation, d'un produit ou d'une application. L'écriture est réalisée soit directement, soit au travers du principe de syslogd (voir en annexe).

4.8 Brique

Une brique est un élément de supervision. Elle est composée de fichiers de logs, de scripts, d'un ensemble de scripts, de programme, d'un ensemble de programmes, … Ceux-ci permettent de statuer sur le fonctionnement d'un objet.

Il existe trois sortes de briques : systèmes d'exploitation, produit, applicative.

4.9 Macro-brique

Une macro-brique est composée de briques. Elle permet de vérifier le fonctionnement d'un système d'information ou d'un serveur.

Remarque Le champ d'action d'une macro-brique est limitée à un serveur physique. Elle peut valider les moyens de connexion ou les moyens d'écoute par rapport aux autres serveurs/produits/applications, mais en aucun cas contenir l'appel d'une brique (ou d'une macro-brique) sur un serveur autre, que celui où elle est utilisée.

4.10 verif_appli.sh

Il existe aujourd'hui des scripts, appelés verif_appli.sh. Ils sont situés dans le répertoire /usr/local/bin/. Ils permettent de vérifier le fonctionnement du serveur sur lequel ils sont installés. Ils correspondront à la macro-brique verif_serveur.ksh.

5 Acteurs/Rôles

Les personnes intervenant dans la réalisation de cette nouvelle supervision sont :

XXXXXXXXXXXXXXXXXX

Acteur	Rôle	Nom(s)
développeurs/intégrateurs applicatifs	écriture et la mise en place des procédures de supervision de leur(s) application(s)	?
intégrateurs systèmes et produits	adaptation des procédures aux SI	?
intégrateurs ITO fonctionnel	mise en place de la supervision sur les serveurs	Jean-Philippe Bocquenet Thibault Delplanque
supports ITO technique et fonctionnel	définition et réalisation de nouvelles supervisions	Pascal Dupuis Jérôme Hedoire Sébastien Gislain
supports systèmes	écriture, mise en place et maintenance des procédures de supervision, des systèmes d'exploitation dont ils sont les supports	ISSD
supports produits	écriture, mise en place et maintenance des procédures de supervision, des produits dont ils sont les supports	?
supports applicatifs	maintenance et l'évolution des procédures	ESI

	applicatives	
exploitants	l'utilisation des procédures de diagnostic	ESI Pilotage

6 Briques

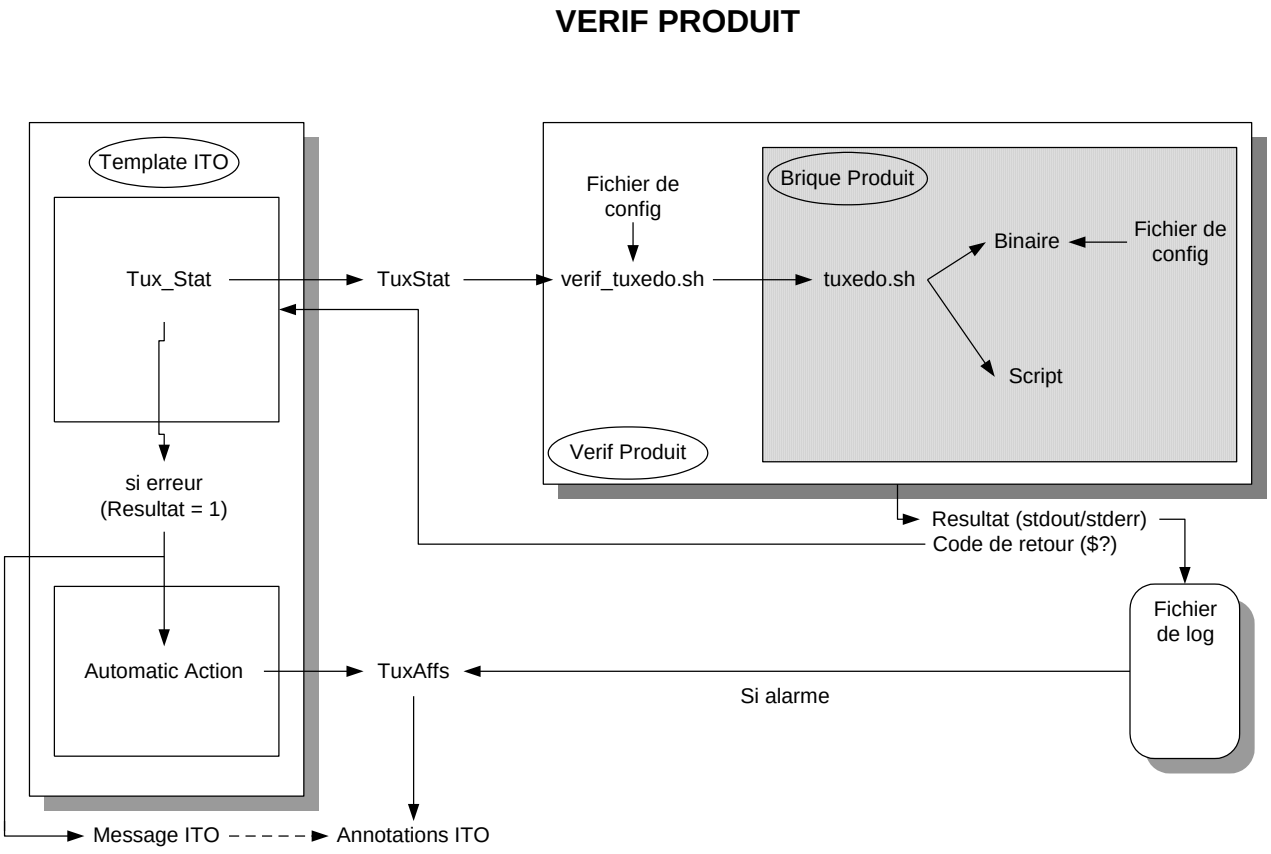
Les règles ci-dessous définissent des bases communes de développement de ces briques.

Les briques peuvent être composées :
- de scripts en Korn Shell,
- de scripts en langage Perl,
- de programmes binaires (préalablement compilés),
- d'une commande avec un ou plusieurs paramètres (par exemple -status),
- de fichiers de logs,
- …

Remarques La compilation des programmes binaires, dans un environnement dédié, est à la charge des équipes les utilisant dans leur brique.

Le déploiement "global" de Perl n'est pas programmé à ce jour. Sa mise en place est de la responsabilité des équipes l'utilisant dans leur brique (sous couvert d'autorisation auprès des supports concernés).

Les briques peuvent utiliser des fichiers de configuration externes : liste des instances, paramètrages, …



6.1 Procédures de vérification

Responsabilité/maintenance : maintenance et évolution des procédures de vérification.
Personnalisation initiale : configuration effectuée lors de la mise en place sur le serveur.
Maintenance personnalisation : évolution de la personnalisation.

6.1.1 Système d'exploitation

Nom	verif_os.ksh
Répertoire	/usr/local/maintenance/verif/
Portée	Ensemble des éléments à vérifier sur un système d'exploitation
Responsabilité/maintenance	Support système
Personnalisation initiale	Intégrateur système
Maintenance personnalisation	Exploitant

6.1.2 Produit

Nom	verif_<produit>.ksh
Répertoire	Arborescence du produit, sous-répertoire "verif" Par exemple : /logiciel/tuxedo/verif/verif_tuxedo.ksh
Portée	Ensemble des instances actives d'un produit sur un serveur
Responsabilité/maintenance	Support produit
Personnalisation initiale	Intégrateur produit
Maintenance personnalisation	Exploitant

6.1.2.1 Application

Nom	verif_<appli>.ksh
Répertoire	Arborescence de l'application, sous-répertoire "verif" Par exemple : /pndope01/pnd/verif/verif_pnd.ksh
Portée	Ensemble des instances actives d'une application
Responsabilité/maintenance	Support applicatif associé à l'application
Personnalisation initiale	Intégrateur applicatif
Maintenance personnalisation	Support applicatif

6.2 Fichiers de logs

6.2.1 Log système

Les fichiers de logs système sont écrits par le système d'exploitation :

Aix	/var/adm/ras/errlog
HP-UX	/var/adm/syslog/syslog.log
Sun Solaris	/var/adm/messages

6.2.2 Log produits

Chaque produit peut générer des fichiers de log. Les formats de ces fichiers sont différents, et ils sont situés dans des répertoires différents.

6.2.3 Log applicatif : Journal de bord

Le journal de bord respecte un format normalisé. Il est supervisé par l'agent ITO du serveur. La remontée d'une alerte est fonction de niveau de gravité du message dans le log applicatif. Ce fichier contient toutes les informations nécessaires au suivi du séquencement et du déroulement des tâches applicatives : début de traitement, fin de traitement, échec, …

6.2.3.1 Nom

Le nom du fichier est fonction du nom de l'application : `<appli>.opc`

6.2.3.2 Répertoire

Ce fichier est situé dans le répertoire :

```
/<host_logique>/<sia>/logs/
```

<host_logique> représente le nom du serveur logique associé à l'application.
<sia> représente le nom du SIA (Système d'Information Applicatif) auquel appartient l'application.

Par exemple : `/temope01/temis/logs/`

6.2.3.3 Responsabilité

Le journal de bord est sous la responsabilité du support applicatif. Il doit être remis à zéro périodiquement.

Remarque Cette purge n'est pas réalisée par ITO.

Exemple de ligne à placer dans la crontab applicative pour effectuer la purge, qui sera lancé tous les jours à 0h05 :

```
05 00 * * * cd /<host_logique>/<sia>/[<sdb>/]logs ; cp <appli>.opc <appli>.opc.old ; \
       cat /dev/null > <appli>.opc
```

ATTENTION Le journal de bord ne doit pas être tronqué. La suppression d'une ou plusieurs lignes dans ce fichier entraînerait la relecture complète du journal de bord par ITO, et donc la remontée en grand nombre d'anciennes alertes.

6.2.3.4 Format

Un message du journal de bord est constitué d'une seule ligne. Cette ligne est découpée en plusieurs champs texte ASCII séparés par un blanc/espace.

Chaque champ suit les règles suivantes :
- Il est toujours situé au même emplacement (ordre).
- Il doit être renseigné. Dans le cas contraire, le champ doit être remplacé par deux guillemets (double quote) : "".
- Il ne doit pas contenir de blanc/espace à l'exception du dernier champ.

Un message est constitué de quatre ensembles d'informations :
- Le premier, en entête de message généré automatiquement par syslogd (voir annexe).
- Le deuxième identifiant l'origine.
- Le troisième situe l'erreur dans les sources de l'application.
- Le quatrième est une description de l'erreur.

```
<Mois> <Jour> <Heure> <Serveur> <Application>[<PID>]: <Terminal>
<Utilisateur> <Programme> <Source> <Ligne> <Gravité> <Domaine> <Numéro>
<Message>
```

1	*<Mois>*

Description Mois du message
Format Chaîne de 3 caractères
Exemple Jan
Commentaire Ne pas envoyer cette information.

2	*<Jour>*

Description Jour du mois

Format Nombre de 2 chiffres
Exemple 03
Commentaire Ne pas envoyer cette information.

3	*<Heure>*

Description Heure
Format Chaîne de caractères au format HH:MM:SS (HH=heures, MM=minutes, SS=secondes)
Exemple 14:22:12
Commentaire Ne pas envoyer cette information.

4	*<Serveur>*

Description Nom du serveur
Format Chaîne de caractères
Exemple aohpc001
Commentaire Ne pas envoyer cette information.

5	*<Application>*

Description Nom de l'application
Format Chaîne de caractères
Exemple Temis

6	*<PID>*

Description Numéro du processus Unix
Format Nombre de 5 chiffres au maximum
Exemple 28252

7	*<Terminal>*

Description Nom du terminal
Format Chaîne de caractères
Exemple /dev/pts/3
Commentaire Dans le cas ou le terminal est inconnu, ce champ doit avoir la valeur not_a_tty.

8	*<Utilisateur>*

Description Nom de l'utilisateur Unix
Format Chaîne de caractères
Exemple temis

9	*<Programme>*

Description Nom du programme Unix
Format Chaîne de caractères
Exemple batch

10	*<Source>*

Description Nom du source du programme
Format Chaîne de caractères
Exemple batch.c
Commentaire Si inconnu mettre la même valeur que le nom du programme (champ 9)

11	*<Ligne>*

Description Numéro de la ligne dans le source du programme
Format Nombre
Exemple 122

12	*<Gravité>*

Description Niveau de criticité de 1 à 5
Format Nombre
Exemple 1
Commentaire

1 – ALERT	message de 'panic', bloquant pour l'application.
2 – WARNING	message d'alerte non bloquant pour l'application.
3 – NOTICE	message d'information destiné à l'exploitation à utiliser avec beaucoup de parcimonie.
4 – INFO	message d'information destiné au support applicatif.
5 – DEBUG	message de trace destiné aux développeurs.

Seuls les messages de gravité 1 à 3 entraîne une remontée ITO.

13	*<Domaine>*

Description Domaine du programme
Format Chaîne de 3 caractères
Exemple APP
Commentaire

APP – Application	message applicatif.
ORA – Oracle	message Oracle.

14	*<Erreur>*

Description Code d'erreur
Format Nombre
Exemple 00000052
Commentaire Ce code d'erreur pourra être associé des consignes

15	*<Message>*

Description Message succinct décrivant l'erreur
Format Chaîne de caractères pouvant contenir des blancs/espaces
Exemple Erreur dans l'intégration des données

6.2.4 Fichier de consignes applicatives

Le fichier de consignes applicatives contient les consignes annotées aux alarmes remontées vers ITO. Il permet d'associer des consignes à une alerte remontée aux exploitants.

6.2.4.1 Nom

Le nom du fichier est fonction du nom de l'application : *<appli>*.txt

6.2.4.2 Localisation

Ce fichier est situé dans le répertoire :

/<host_logique>/<sia>/logs/

<host_logique>	représente le nom du serveur logique associé à l'application.
<sia>	représente le nom du SIA (Système d'Information Applicatif) auquel appartient l'application.

Par exemple : /temope01/temis/logs/

6.2.4.3 Responsabilité

Le fichier de consignes applicatives est créé par le projet.
Il est ensuite mis à jour par le support applicatif.

6.2.4.4 Format

Le format de ce fichier est composé d'un paragraphe pour chaque alerte.
Ces paragraphes sont séparés par une ligne vide (blanche).

```
<numéro_d_erreur> : <libellé>
// Cause :
//   <explication de la cause du problème
//   sur autant de lignes que nécessaire
// Action :
//   détail des actions à réaliser
//   sur autant de lignes que nécessaire
```

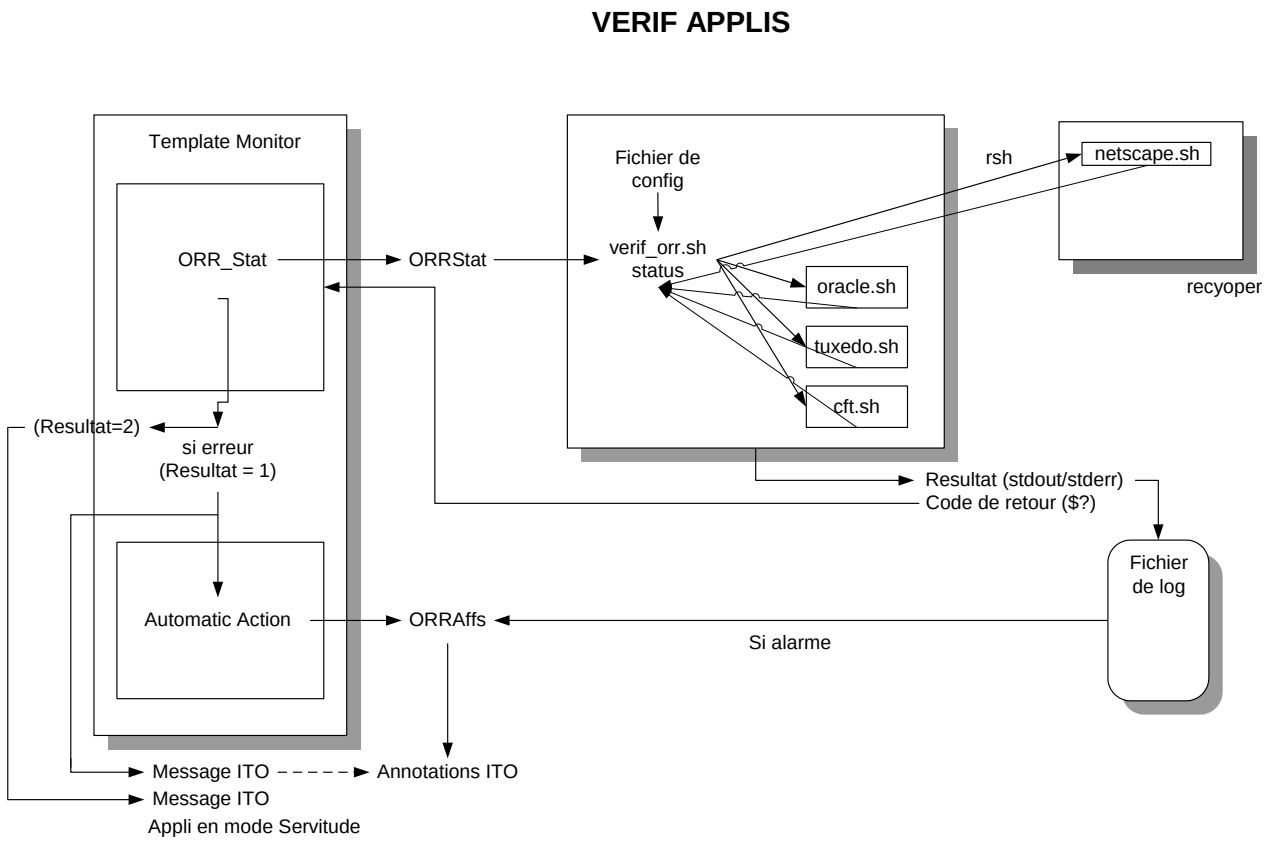
<numéro_d_erreur> correspond au code d'erreur utilisé dans le journal de bord (en un seul mot sans blanc/espace).

<libellé> correspond à une description succincte de l'erreur.

Voir l'exemple d'un fichier de consignes en annexe.

7 Macro-briques

Les macro-briques regroupent les résultats de plusieurs briques.



7.1.1 SI

Nom	verif_<SI>.ksh
Localisation	/usr/local/bin/
Portée	Ensemble des instances actives d'un système d'information
Responsabilité/maintenance	Support applicatif et des exploitants associés à l'application
Personnalisation initiale	Intégrateurs fonctionnels et les équipes projet
Maintenance personnalisation	Support applicatif

7.1.2 Serveur

Nom	verif_serveur.ksh
Localisation	/usr/local/bin/
Portée	Ensemble du serveur : du système d'exploitation jusqu'à l'application
Responsabilité/maintenance	Exploitants du serveur
Personnalisation initiale	Lors de la configuration du serveur par les intégrateurs fonctionnels
Maintenance personnalisation	Exploitant

7.2 Procédures de vérification

Le lancement d'une des procédures (briques ou macro-briques) a pour résultats :
- un code de retour (numérique),
- un affichage de diagnostic.

Remarques Ces procédures doivent pouvoir être lancé depuis le compte administrateur root. Elles devront charger l'environnement nécessaire à leur exécution.

7.2.1 Code de retour

Ce code de retour permet la récupération simple du status de fonctionnement de l'objet vérifié.

Les valeurs autorisées sont :
- 0 (zéro) pour définir un fonctionnement correct,
- 1 (un) pour une absence de fichier de LOCK,
- > 1 pour tout autre dysfonctionnement.

7.2.2 Diagnostic

La procédure doit afficher sur la "sortie standard" (fenêtre de lancement en mode interactif) un diagnostic permettant de visualiser les problèmes trouvés.

La fin du diagnostic doit comporter une synthèse globale :
- OK pour une vérification sans erreur,
- KO pour une vérification avec une ou plusieurs erreurs.

L'opérateur au vue du status global KO pourra revenir sur le diagnostic situé au dessus pour visualiser le ou les problèmes mis en évidence par la procédure de vérification et au besoin relancer la procédure.

Remarque Le contenu du diagnostic doit rester succinct car il permettra de situer rapidement où le dysfonctionnement est situé.

Voir exemple en annexe.

8 Annexes

8.1 Principe de génération des messages via syslog

Les systèmes Unix mettent à disposition un système de gestion des logs grâce à syslog. Ce système permet :
- d'homogénéiser le principe.
- de gérer les écritures concurrentielles dans les fichiers de logs.
- de porter simplement ce mécanisme d'un système d'exploitation à l'autre.
- de faciliter la gestion des droits d'écriture dans les fichiers de logs.

Par contre, il est limité à 8 fichiers de logs.

8.1.1 Fichier de configuration de syslogd

Le fichier de configuration est `/etc/syslog.conf`. Il contient les méthodes d'aiguillage des alertes : fichier de logs, message sur la console, … Ce fichier est relu par syslogd avec la commande : `kill -HUP <PID_syslogd>` (en root).

Chaque méthode est liée à un ou plusieurs couples : domaine – criticité. Les domaines sont prédéfinis. Ceux utilisés pour les journaux de bord sont : `local0`, `local1`, `local2`, `local3`, `local4`, `local5`, `local6`, `local7`.

8.1.1.1 Niveaux de criticité de syslog

Les niveaux de criticité sont fixés et prédéfinis :

Niveau syslogd		Niveau ITO		Signification	Message remonté
0	emerg			Système HS	Non
1	alert			Intervention immédiate	Non
2	crit			Condition critique	Non
3	err	1	CRIT	Erreur bloquante	**Oui**
4	warning	2	WARNING	Erreur non bloquante	**Oui**
5	notice	3	NOTICE	Message d'information significatif	**Oui**
6	info	4	INFO	Information applicative	Non
7	debug	5	DEBUG	Message de debug	Non

Par défaut, le niveau 8 inclus tous les messages du niveau 7, le niveau 7 tout ceux du 6, …

La redirection du journal de bord d'une application, décrite dans le fichier de configuration de syslogd, sera donc :

```
local0.notice /<host_logique>/<sia>/[<sdb>/]logs/<appli>.opc
```

Attention, il faut déclarer la présence de la redirection dans les autres lignes qui utilisent des caractères génériques, pour éviter que les messages soient aussi écrits dans les autres fichiers de log et/ou sur la console du serveur (voir les exemples de fichiers de configuration de syslogd dans les annexes) :

```
…;local0.none /…/syslog.log
```

8.1.1.2 Reponsabilité

La modification du fichier de configuration de syslog est réalisé par l'intégration fonctionnel.
La mise à jour est réalisée par les exploitants.

8.1.2 Exemple de sortie d'une procédure de diagnostic

```
$ ./verif_os_def

#===============================================================================
Recherche des processus 'defunc' (ps -edf | grep defunct) :

  oracle 27154  8882 134 16:09:20 ?         0:02 <defunct>
  oracle 29223 23004 88 16:20:48 ?         0:00 <defunct>

2 processus trouves.


####### #    #
#      # #   #
#      # #  #
#      # ###
#      # #  #
#      # #   #
####### #    #

#===============================================================================
```

8.1.3 Exemple de script de génération d'un message syslogd : `log_msg.sh`

```
#!/bin/sh
#===============================================================================
# log_msg.sh
#===============================================================================
#
function log_msg {
        S_MSG_LOCAL=local0.notice
        S_APPLICATION=APPLICATION
#
        S_UTILISATEUR=$LOGNAME
        S_PROGRAMME=$0
        S_SOURCE=$0
        S_LIGNE="_"
        S_DOMAINE=APP
        S_NUMERO=0
        S_PID=$$
#
        $CRITICITE=${1:-3}
        $MESSAGE="${2}"
###
        /usr/bin/logger -p $S_MSG_LOCAL -t "$S_APPLICATION[$S_PID]" "$S_UTILISATEUR \
$S_PROGRAMME $S_SOURCE $S_LIGNE $CRITICITE $S_DOMAINE $S_NUMERO $MESSAGE"

}

log_msg 1 "Message critical de test"

log_msg 2 "Message warning de test"

log_msg 3 "Message normal de test"

log_msg 4 "Message informatif de test"

#= eof ========================================================================
```

8.1.4 Exemple du contenu d'un fichier de log applicatif

```
Jun 26 14:52:06 cervin CAPA[27181]: /dev/ttyp9 icapa exemple exemple.c 16 1
     APP A01 Probleme applicatif bloquant
Jun 26 14:52:06 cervin CAPA[27181]: /dev/ttyp9 icapa exemple exemple.c 20 2
     APP A02 Probleme applicatif non bloquant
Jun 26 14:52:06 cervin CAPA[27181]: /dev/ttyp9 icapa exemple exemple.c 24 3
     APP A03 Message d'information vers OpC
Jun 26 14:52:06 cervin CAPA[27181]: /dev/ttyp9 icapa exemple exemple.c 28 4
     APP A04 Message d'information
```

```
Jun 26 14:52:06 cervin CAPA[27181]: /dev/ttyp9 icapa exemple exemple.c 32 5
     APP A05 Message de trace debug
```

8.2 Exemples de fichiers de configuration de syslogd

Remarque Les parties à modifier sont en gras.

8.2.1 Plateforme Sun Solaris

```
#
# /etc/syslog.conf
#
*.err;kern.notice;auth.notice;local0.none       /dev/console
*.err;kern.debug;daemon.notice;mail.crit;local0.none /var/adm/messages

*.alert;kern.err;daemon.err                     operator
*.alert                                         root

*.emerg                                         *
#
# Journal de bord de l'application <appli1>
#
local0.notice                   /temope01/logs/temis.opc
```

8.2.2 Plateforme HP-UX

```
#
# /etc/syslog.conf
#
mail.debug                      /usr/spool/mqueue/syslog
*.info;mail,local0.none         /usr/adm/syslog/syslog.log
*.alert                         /dev/console
*.alert                         root
*.emerg                         *
#
# Journal de bord de l'application <appli1>
#
local0.notice                   /temope01/logs/temis.opc
```

8.2.3 Exemple du contenu d'un fichier de consignes applicatives

```
////////////////////////////////////////////////////////////////////////////////////////
/ messages d'erreurs de l'application SIGNE-DOC Vehicules
/ - serveur central
////////////////////////////////////////////////////////////////////////////////////////
/ 50000001 - 50000702  erreurs relatives à la réplication du LEXIQUE
/ 60000001 - 60000601  erreurs relatives aux extractions pour uap's
////////////////////////////////////////////////////////////////////////////////////////
/  Création :
/ - indiquer SVP les modifications et les ajouts ci-après
////////////////////////////////////////////////////////////////////////////////////////

50000001 : delta LEXIQUE obsolète - bases LEXIQUE incohérentes
// CAUSE :
//   erreur de sequencement CFT ou tentative d'insertion manuelle
//   d'un fichier de delta LEXIQUE plus ancien que le dernier pris en compte .
// ACTION :
//   réalignement manuel des bases LEXIQUE
//   contacter le support applicatif

50000201 : échec insertion LEXIQUE
// CAUSE :
//   erreur applicative du programme i.exe
//   le code retour est entre [] dans le libelle.
// ACTION :
//   contacter le support applicatif
```

8.3 Verif_{SI} générique

8.3.1 Source du script verif_app_x.sh

```
#!/usr/bin/ksh
#set -x
#
#===============================================================================
# Nom            : verif_app_x.sh
# Projet         : Supervision applicative AUTOSYS
#                         ce script verifie tous les produits d'un projet
#                         (decommentes) a partir du fichier ../lib/list_proc_sia
#
# Repertoires     : script -> /siaope01/sia/bin/
#                   environnement -> /siaope01/sia/lib/list_proc_sia
#                   etat du projet (optionnel) -> /siaope01/sia/lib/etat_actuel_sia
#
#
# Cree le        : 20/02/2001
# Cree par       : Christian GRANDCOIN
# Objet          : Verifie le fonctionnement d'une application sur un serveur
# Appel(s)       : commande en ligne & HP-OV-ITO Template Vap_Stat_${NumeroInstance}
# Environnements : HP-UX 9.05 10.20
#                : Solaris 5.3 5.4 5.5 5.5.1 5.6
#                : SunOs 4.1.3
#                : Aix 3.x 4.2 4.3
#                : Irix 5.3
# Parametre(s)   : SIA : SIA de l'application
# Codes retour   : = 0 => OK
#                : = 1 => KO
#                : = 2 => Servitude
#                : = 3 => Erreur Environnement
#===============================================================================
# Modification(s)
#-------------------------------------------------------------------------------
# JJ/MM/AA - QUI
#        QUOI
#
# 20/02/01 - CGR
#        Modification pour rendre le script generique
#        ( SIA, affranchissement des repertoires )
#        Modification case Oracle
# 21/02/01 - CGR
#        Modification pour rendre le script generique
# 28/02/01 - CGR
#        Correction ambiguite chaine de car. pour case
# 08/03/01 - CGR
#        Simplification, ajout fonction result,case siebel,
#        cases serveurs locaux et distants.
# 09/03/01 - CGR
#        remsh correction cas H.P. (rcmd S.C.O., rsh autres)
#        Redirection entree standard >/dev/null
# 14/03/01 - CGR
#        resolution heterogeneite : export tous SIA
# 16/03/01 - CGR
#        resolution heterogeneite : ajout verif_lambda,
#        a parametrer suivant les applis.
# 14/06/01 - CGR
#        Modification pour prise en compte  STATUS distant
#        et log CFT a l'alternat.
# 20/06/01 - CGR
#        Integration specificite SITMO
# 26/06/01 - Francois JOLIET
#        Integration du verif package sur cluster HP
# 27/06/2001 - Tibo DELPLANQUE & CGR
#        rajout du case essbase
# 27/06/2001 - JPh BOCQUENET
#        rajout des cases progress et mqm
# 28/06/2001 - CGR
#        Preconisation SSD : Utilisation du script en local
#        Modification de l'init des variables d'environnement
# 02/07/2001 - CGR
#        Modification determination des repertoires
#        Fusion des fichiers des fichiers d'environnement
#        et traitement multi-instances du meme produit
# 10/07/2001 - CGR
#        Modification case Oracle  : - Chemin LOG ORACLE si non standard
#        Modification case Autosys : - Fonctionnement degrade
#
#===============================================================================

#===============================================================================
```

```
# result
# Affichage de l'etat
# Code retour    : 0 => OK
#                : 1 => KO
#                : 2 => MAINTENANCE
#===============================================================================

result()
        {
        if [ $1 -eq '0' ]; then
            echo "\n\tFonctionnement $2 OK" >> $BINDIR/result_${SIA}.tmp
            echo "\t------------------------------" >> $BINDIR/result_${SIA}.tmp
        elif [ $1 -eq '2' ]; then
            echo "\n\tMaintenance de $2" >> $BINDIR/result_${SIA}.tmp
            echo "\t------------------------------" >> $BINDIR/result_${SIA}.tmp
            MAINT='1'
        else
            echo "\n\tFonctionnement $2 KO" >> $BINDIR/result_${SIA}.tmp
            echo "\t##############################" >> $BINDIR/result_${SIA}.tmp
            RC='1'
        fi
        }

#-------------------------------------------------------------------------------
# Affichage de l'utilisation du script
#-------------------------------------------------------------------------------

script=$(basename $0)
SIA=$1

if [ $# -ne 1 ]
then
        echo "\n\tUsage : ${script} [sia du projet]\n"
        exit 3
fi


if [ -L ${script} ]; then
        echo "\n\tLien Non Gere\n"
        exit 3
fi

#-------------------------------------------------------------------------------
# determination des repertoires
#-------------------------------------------------------------------------------


PROG=$(which $0)
cd $(dirname $PROG)
BINDIR=`pwd`
PROJ=$(dirname $BINDIR)
LIBDIR=$PROJ/lib

#echo "\n\t PROG : $PROG"
#echo "\t BINDIR : $BINDIR"
#echo "\t PROJ : $PROJ"
#echo "\t LIBDIR : $LIBDIR\n"


#-------------------------------------------------------------------------------
# Liste des Produits a verifier
#-------------------------------------------------------------------------------

if [ ! -f $LIBDIR/list_proc_${SIA} ]; then
         echo "\nle fichier list_proc_${SIA} n'existe pas dans $(dirname $LIBDIR/list_proc_${SIA})\n"
         exit 3
else
         ENVFILE=$LIBDIR/list_proc_${SIA}
fi

#-------------------------------------------------------------------------------
# Fichier resultat
#-------------------------------------------------------------------------------

if [  -f $BINDIR/result_${SIA}.tmp ]; then
         rm $BINDIR/result_${SIA}.tmp
fi

#-------------------------------------------------------------------------------
# etat du projet (Optionnel)
#-------------------------------------------------------------------------------

cat $LIBDIR/etat_actuel_${SIA}

#-------------------------------------------------------------------------------
# etat des produits
#-------------------------------------------------------------------------------

RC='0'
MAINT='0'
```

```
PRODUIT_COURANT=$BINDIR/produit_courant.txt

cat $ENVFILE | grep -v "#" | while read ligne
        do
                echo $ligne > $PRODUIT_COURANT
                produit=`cat $PRODUIT_COURANT | awk -F"=" '{print $1}'`
                case ${produit} in
                        siebel)
                                SIA_PROD=`cat $PRODUIT_COURANT | awk -F"=" '{print $2}'`
                                  /logiciel/siebel/exploit/verif_siebel.sh ${SIA_PROD}
                                  result "$?" "${produit}"
                                  ;;
                        siebel_distant)
                                SERVEUR=`cat $PRODUIT_COURANT | awk -F"=" '{print $3}'`
                                SIA_PROD=`cat $PRODUIT_COURANT | awk -F"=" '{print $2}'`
                                  echo "\n----- ${produit} sur ${SERVEUR} -----"
                                remsh "${SERVEUR}" -n "/logiciel/siebel/exploit/verif_siebel.sh
${SIA_PROD}" > $BINDIR/${produit}.txt
                                  cat $BINDIR/${produit}.txt
                                  result_dis=`cat $BINDIR/${produit}.txt | grep STATUS | awk -F"=" '{print
$2}'`
                                  echo "\nSTATUS : $result_dis"
                                  rm $BINDIR/${produit}.txt
                                result "$result_dis" "${produit}"
                                ;;
                        tuxedo)
                                SIA_PROD=`cat $PRODUIT_COURANT | awk -F"=" '{print $2}'`
                                  /logiciel/tuxedo/verif/verif_tuxedo.ksh ${SIA_PROD}
                                result "$?" "${produit}"
                                  echo "LOG ${produit}"
                                  tail -3 /logiciel/tuxedo/tuxedo_${SIA_PROD}/ulog_${SIA_PROD}
                                  echo "\n"
                                  ;;
                        acstux)
                                SIA_PROD=`cat $PRODUIT_COURANT | awk -F"=" '{print $2}'`
                                /logiciel/acstux/verif_acstux.sh ${SIA_PROD}
                                  result "$?" "${produit}"
                                  echo "LOG ${produit}"
                                  tail -3
/${SIA_PROD}ope01/logiciel/acstux/acstux_${SIA_PROD}/logs/acsw3.opc
                                  echo "\n"
                                  ;;
                        cft)
                                SIA_PROD=`cat $PRODUIT_COURANT | awk -F"=" '{print $2}'`
                                /logiciel/cft/verif_cft.sh ${SIA_PROD}
                                result "$?" "${produit}"
                                  echo "LOG ${produit}"
                                  tail -3 /logiciel/cft/cft_${SIA_PROD}/fillog/cft_log
                                  tail -3 /logiciel/cft/cft_${SIA_PROD}/fillog/cft_loga
                                  echo "\n"
                                  ;;
                        cft_distant)
                                SERVEUR=`cat $PRODUIT_COURANT | awk -F"=" '{print $3}'`
                                SIA_PROD=`cat $PRODUIT_COURANT | awk -F"=" '{print $2}'`
                                  echo "\n----- ${produit} sur ${SERVEUR} -----"
                                remsh "${SERVEUR}" -n "/logiciel/cft/verif_cft.sh ${SIA_PROD}" >
$BINDIR/${produit}.txt
                                cat $BINDIR/${produit}.txt
                                result_dis=`cat $BINDIR/${produit}.txt | grep STATUS| awk -F"=" '{print
$2}'`
                                rm $BINDIR/${produit}.txt
                                result "${result_dis}" "${produit}"
                                echo "LOG ${produit}"
                                remsh "${SERVEUR}" -n "tail -3
/logiciel/cft/cft_${SIA_PROD}/fillog/cft_log"
                                remsh "${SERVEUR}" -n "tail -3
/logiciel/cft/cft_${SIA_PROD}/fillog/cft_loga"
                                echo "\n"
                                ;;
                        oracle)
                                SIA_PROD=`cat $PRODUIT_COURANT | awk -F"=" '{print $2}'`
                                LOG_ORA=`cat $PRODUIT_COURANT | awk -F"=" '{print $3}'`
                                  if [ -z "$LOG_ORA" ]; then
                                          LOG_ORA=/logiciel/oracle/oracle_${SIA_PROD}/alert_${SIA_PROD}.log
                                  fi
                                  /logiciel/oracle/lbin/verif_oracle.ksh ${SIA_PROD}
                                result "$?" "${produit}"
                                  echo "LOG ${produit}"
                                  tail -10 ${LOG_ORA}
                                  echo "\n"
                                  ;;
                        essbase)
                                /logiciel/essbase/scripts/ess_mngr status
                                result "$?" "${produit}"
                                echo "LOG ${produit}"
                                tail -10 /logiciel/essbase/Essbase.log
                                echo "\n"
                                ;;
                        progress)
```

```
                                SIA_PROD=`cat $PRODUIT_COURANT | awk -F"=" '{print $2}'`
                                /${SIA_PROD}ope01/logiciel/progress/verif_progress.sh ${SIA_PROD}
                                result "$?" "${produit}"
                                echo "\n"
                                ;;
                        netscape)
                                SIA_PROD=`cat $PRODUIT_COURANT | awk -F"=" '{print $2}'`
                                /logiciel/netscape/netscape.sh status ${SIA_PROD}
                                result "$?" "${produit}"
                                ;;
                         was)
                                SIA_PROD=`cat $PRODUIT_COURANT | awk -F"=" '{print $2}'`
                                /logiciel/was/wasAS.sh status ${SIA_PROD}
                                result "$?" "${produit}"
                                ;;
                         netscape_distant)
                                SERVEUR=`cat $PRODUIT_COURANT | awk -F"=" '{print $3}'`
                                SIA_PROD=`cat $PRODUIT_COURANT | awk -F"=" '{print $2}'`
                                  echo "\n----- ${produit} sur ${SERVEUR} -----"
                                remsh "${SERVEUR}" -n "/logiciel/netscape/netscape.sh status ${SIA_PROD}" >
$BINDIR/${produit}.txt
                                cat $BINDIR/${produit}.txt
                                result_dis=`cat $BINDIR/${produit}.txt | grep STATUS| awk -F"=" '{print
$2}'`
                                rm $BINDIR/${produit}.txt
                                result "${result_dis}" "${produit}"
                                  ;;
                        was_distant)
                                SERVEUR=`cat $PRODUIT_COURANT | awk -F"=" '{print $3}'`
                                SIA_PROD=`cat $PRODUIT_COURANT | awk -F"=" '{print $2}'`
                                echo "\n----- ${produit} sur ${SERVEUR} -----"
                                remsh "${SERVEUR}" -n "/logiciel/was/wasAS.sh status ${SIA_PROD}" >
$BINDIR/${produit}.txt
                                cat $BINDIR/${produit}.txt
                                result_dis=`cat $BINDIR/${produit}.txt | grep STATUS| awk -F"=" '{print
$2}'`
                                rm $BINDIR/${produit}.txt
                                result "${result_dis}" "${produit}"
                                ;;
                        autosys)
                                autosys_inst=`cat $PRODUIT_COURANT | grep ${produit} | awk -F"=" '{print
$2}'`
                                autosys_bd=`cat $PRODUIT_COURANT | awk -F"=" '{print $3}'`
                                autosys_proc=`cat $PRODUIT_COURANT | awk -F"=" '{print $4}'`
                                autosys_nbproc=`cat $PRODUIT_COURANT | awk -F"=" '{print $5}'`
                                autosys_lock=`cat $PRODUIT_COURANT | awk -F"=" '{print $6}'`
                                autosys_log=`cat $PRODUIT_COURANT |  awk -F"=" '{print $7}'`
                                autosys_fs=`cat $PRODUIT_COURANT | awk -F"=" '{print $8}'`
                                echo "\n----- ${produit} ${autosys_inst}-------------------------"
                                /logiciel/autosys/verif_autosys.sh ${autosys_inst} ${autosys_bd}
${autosys_proc} ${autosys_nbproc} ${autosys_lock} ${autosys_log} ${autosys_fs}
                                if [ "$?" -eq 3 ]; then
                                        echo "\n\tAUTOSYS FONCTIONNE AVEC DES PROBLEMES"
                                        echo "\t---------------------------------------"
                                          RC=1
                                  fi
                                  result "$RC" "${produit} ${autosys_inst}"
                                ;;
                        lambda)
                                  # Peut etre utilise pour verifier un produit non prevu dans la liste
                                  # -------------------------------------------------------------------
                                SIA_LAMBDA=`cat $PRODUIT_COURANT | awk -F"=" '{print $2}'`
                                lambda=`cat $PRODUIT_COURANT |  awk -F"=" '{print $3}'`
                                Nb_proc_lambda=`cat $PRODUIT_COURANT | awk -F"=" '{print $4}'`
                                lock_lambda=`cat $PRODUIT_COURANT | awk -F"=" '{print $5}'`
                                log_lambda=`cat $PRODUIT_COURANT | awk -F"=" '{print $6}'`
                                echo "\n----- ${lambda}-------------------------"
                                /usr/local/cron_script/verif_lambda.sh ${SIA_LAMBDA} ${lambda}
${Nb_proc_lambda} ${lock_lambda} ${log_lambda}
                                result "$?" "${lambda}"
                                ;;
                        lambda_distant)
                                SERVEUR=`cat $PRODUIT_COURANT | awk -F"=" '{print $7}'`
                                SIA_LAMBDA=`cat $PRODUIT_COURANT | awk -F"=" '{print $2}'`
                                lambda=`cat $PRODUIT_COURANT |  awk -F"=" '{print $3}'`
                                Nb_proc_lambda=`cat $PRODUIT_COURANT | awk -F"=" '{print $4}'`
                                lock_lambda=`cat $PRODUIT_COURANT | awk -F"=" '{print $5}'`
                                log_lambda=`cat $PRODUIT_COURANT | awk -F"=" '{print $6}'`
                                echo "\n----- ${lambda} sur ${SERVEUR} -----"
                                remsh "${SERVEUR}" -n "/usr/local/cron_script/verif_lambda.sh ${SIA_LAMBDA}
${lambda} ${Nb_proc_lambda} ${lock_lambda} ${log_lambda}" > $BINDIR/${produit}.txt
                                cat $BINDIR/${produit}.txt
                                result_dis=`cat $BINDIR/${produit}.txt | grep STATUS| awk -F"=" '{print
$2}'`
                                rm $BINDIR/${produit}.txt
                                result "${result_dis}" "${lambda}"
                                ;;
                        acsw3)
                                # SPECIFIQUE SITMO
                                SIA_PROD=`cat $PRODUIT_COURANT | awk -F"=" '{print $2}'`
```

```
                                /logiciel/acsw3tux/acsw3tux_sit/verif_acsw3.sh ${SIA_PROD}
                                result "$?" "${produit}"
                                echo "\n"
                                 ;;
                        tracker)
                                # SPECIFIQUE SITMO
                                SIA_PROD=`cat $PRODUIT_COURANT | awk -F"=" '{print $2}'`
                                /logiciel/tracker/tracker_sit/verif_tracker.sh status ${SIA_PROD}
                                result "$?" "${produit}"
                                ;;
                        acs)
                                # SPECIFIQUE SITMO
                                SIA_PROD=`cat $PRODUIT_COURANT | awk -F"=" '{print $2}'`
                                /logiciel/acstux/acstux_sit/verif_acs.sh ${SIA_PROD}
                                result "$?" "${produit}"
                                  ;;
                        mqm)
                                SIA_PROD=`cat $PRODUIT_COURANT | awk -F"=" '{print $2}'`
                                  /logiciel/mqm/verif_mqm.sh ${SIA_PROD}
                                result "$?" "${produit}"
                                ;;
                        cluster_hp)
                                /usr/local/bin/verif_cluster_hp.ksh ${SIA_CLUSTER_HP}
                                  result "$?" "${produit}"
                                ;;
                         *)
                                  echo "\n\t Produit NON Verifie : "${produit}" " >>
$BINDIR/result_${SIA}.tmp
                                  echo "\t ================================ " >> $BINDIR/result_${SIA}.tmp
                                  ;;
                esac
        done

cat $BINDIR/result_${SIA}.tmp
rm $BINDIR/result_${SIA}.tmp
rm $PRODUIT_COURANT

#-------------------------------------------------------------------------------
# Test si un KO ou Maintenance pour sortir
#-------------------------------------------------------------------------------

echo
if [ $RC -eq 1 ]
then
   banner KO
   exit 1
else
   if [ $MAINT -eq 1 ]
   then
        banner Servitude
# En attendant de trouver une solution dans ITO pour gerer les servitudes
# Remonte un code OK provisoirement
#        exit 2
         exit 0
   else
        banner OK
        exit 0
   fi
fi

#
# eof
#===============================================================================
```

8.3.2 Fichier d'environnement du verif_app_x.sh

```
# ------------------------------------------------------------------------
#
#       Auteur : C.GRANDCOIN
#
#       Liste des produits et de leurs environnements pour verif_app_x.sh
#
#       Repertoire        : /siaope01/sia/lib/
#       Utilisation       : De-documenter la ligne ou se trouve le produit
#                           qui doit etre appelé
#                           Les champs sont séparés par des '='
#       1er Champ         : Produit.
#                           Plusieurs occurences du meme produit sont possibles.
#                           ( traitement multi-instances )
#       2eme Champ        : Nom de l'instance du produit (Peut etre
#                           différente du SIA de l'application.
#       Champs suivants : Parametres des fonctions Produits (ordre d'appel)
#       Dernier Champ     : Nom du Serveur distant
#                           (Rappel : commandes remote non preconisees par SSD)
#
#       Contraintes       : Pas de lignes non renseignees
#
# ------------------------------------------------------------------------
#
#oracle=alc=/logiciel/oracle/oracle_autoalc/alert_autoalc.log=
#oracle=cms=
#essbase=cms=
#tuxedo=cms=
#acstux=cms=
#cft=bpd=
#cft_distant=bpd=aosus016.mc2.renault.fr=
#siebel=cms=
#siebel_distant=icr=aosus053.mc2.renault.fr=
#netscape=cms=
#was=cms
#netscape_distant=bpd=aosus016.mc2.renault.fr=
#was_distant=bpd=aosus016.mc2.renault.fr=
#mqm=cms=
#progress=cms
#autosys=ALC=autoalc=event_demon=1=/atsope01/logiciel/autosys/autouser/out/LOCK=/atsope01/logiciel/autosys/
#autouser/out/event_demon.ALC=atsope01=
#lambda=cms
#lambda_distant=bpd=java=1=/bpdope02/bpd/logs/LOCK-
#bpd=/bpdope02/bpd/logs/stderr.txt=aosus016.mc2.renault.fr=
#acsw3=cms
#acs=cms
#tracker=cms
#cluster_hp=
#
```