Un exemple de mouchard

Moucharder ce qu'il se passe dans une application est très intéressant d'une part pendant la mise au point : cela permet de vérifier que le programme prend bien les chemins que nous avons programmés; d'autre part après la mise en production où, en cas de plantage, on peut avoir l'historique de tout ce qui s'est passé avant le plantage.

Un mouchard doit être :

1. simple d'emploi
2. rapide
3. valide dans toutes les parties de l'application
4. exploitable
5. purgeable

Après avoir essayé diverses méthodes, j'en suis arrivé à celle-ci :

le mouchard est un fichier texte structuré (DELIMITED WITH TAB). On l'ouvre (ou on le crée) au lancement de l'application; son handle (le résultat de FOPEN() ou FCREATE()) est mis dans une variable PUBLIC qui est donc visible partout dans l'application. Chaque ligne est structurée comme suit (c'est une tabulation (CHR(9) qui sépare les 'champs')

1. un champ DATETIME : c'est l'horodateur
2. un champ INTEGER : c'est le code de l'événement. A chaque partie de l'application correspond une tranche de code (par exemple 0 à 999 démarrage, 1000 à 1999 paramétrage, etc .. On peut être très large : le nombre de possibilité dépassant le millard !
3. un champ texte contenant le nom de la form ou du prg
4. un champ texte contenant le nom de la méthode ou de la routine
5. un champ texte contenant un texte explicatif en clair
6. un champ numérique (normalement avec 2 décimales) contenant une valeur
7. ensuite, on peut personnaliser la structure : par exemple dans mon application de gestion d'école, j'ai un champ contenant la clef de la famille ou de l'élève.

Seules les 2 premiers champs sont 'obligatoires'; les textes ne sont pas (trop) limités en longueur. On écrit dans ce mouchard via une méthode dans une classe à qui on passe les données en paramètres.

Ce n'est pas parce que c'est un .TXT que sa taille est limitée : j'en ai un de 400 Mo.

On peut facilement lire ce mouchard avec un MODIFY FILE et voir ainsi le déroulement du programme. Mais on peut aussi, en cas de besoin, l'importer dans une table temporaire avec APPEND FROM (c'est rapide) et, à partir de là, faire des recherches, des filtres, des extractions SQL, etc … Lors de cet import, les textes trop longs sont tronqués mais comme, en général, on travaille sur les codes événement ou le nom des forms ou des méthodes, cela ne pose pas de problème.

J'ai créé une méthode de purge qui copie les lignes vers un nouveau fichier en éliminant celles avant une date donnée (800 jours dans le cas des écoles).
J'ai aussi créé une méthode qui en extraie les 10000 derniers caractères; cet extrait, zippé,

m'est envoyé en pièce jointe dans le courriel que l'application m'envoie en cas d'erreur

Voici la procédure d'ouverture ou de création de ce mouchard. Le nom des propriétés doit être suffisamment parlant ! gn_handle_logtxt est le nom de la variable PUBLIC qui contient le handle.

```
IF !EMPTY(.repertoire_mouchard_texte)
  * si le répertoire du mouchard texte, n'existe pas, on le crée
  * car on va y créer le fichier ici.
  lnaccu = ADIR(lttab, .repertoire_mouchard_texte, "D")
  IF m.lnaccu = 0
    MKDIR (This.repertoire_mouchard_texte)
    DOEVENTS
    * on vérifie que le répertoire ait bien été créé (parce que l'on
    * peut avoir des problèmes de droit par exemple)
    lnaccu = ADIR(lttab, .repertoire_mouchard_texte, "D")
    IF m.lnaccu =0
      * le répertoire n'a pas été créé
      * on remet à blanc le nom du mouchard pour éviter qu'il soit
      * ouvert dans standard_mau.init()
      .mouchard_texte_nom = ""
    ENDIF && m.lnaccu =0
  ENDIF && m.lnaccu = 0
ENDIF && !EMPTY(.repertoire_mouchard_texte)
* on regarde si le mouchard texte est ouvert sinon
* s'il existe on essaie de l'ouvrir sinon on le crée
.mouchard_texte_ya = .F.
.mouchard_texte_ouverture = 0
IF ISNULL(m.gn_handle_logtxt) OR m.gn_handle_logtxt = -999
  IF !EMPTY( .mouchard_texte_nom) && AND m.llok
    lcaccu = .repertoire_mouchard_texte+ .mouchard_texte_nom
    IF ADIR(lttab, m.lcaccu)> 0 && FILE(.mouchard_texte_nom)
      * jme 15/11/2009 en cas d'erreur, on va envoyer les 100000 derniers
      *   caractères du mouchard texte. Il faut donc pouvoir le lire ...
      * 12 : lecture/écriture sans buffer
      gn_handle_logtxt = FOPEN(m.lcaccu, 12)
      .mouchard_texte_ouverture = 1
    ELSE
      gn_handle_logtxt = FCREATE(m.lcaccu, 0) && 0 read/write
      .mouchard_texte_ouverture = 2
    ENDIF && FILE(m.gnmouchardtxt)
    IF m.gn_handle_logtxt >-1
      .mouchard_texte_ya = .T.
      = FSEEK(m.gn_handle_logtxt, 0,2) && positionne à la fin du mouchard
      = FPUTS(m.gn_handle_logtxt, "") && on met un 'séparateur'
      = FPUTS(m.gn_handle_logtxt, TTOC(DATETIME())+ CHR(9)+ "-1" + ;
       CHR(9) + IIF(.mouchard_texte_ouverture = 2, "création", "ouverture")+ ;
        " du mouchard")
    ELSE
      .mouchard_texte_ouverture = -1
    ENDIF && m.gn_handle_logtxt >-1
  ENDIF && !EMPTY( .mouchard_texte_nom)
ELSE
  IF VARTYPE(m.gn_handle_logtxt)=="N"
    IF m.gn_handle_logtxt > -1
      * le mouchard texte est déjà ouvert. C'est pas bien normal (on est au
      * début de l'application) mais bon ...
      .mouchard_texte_ya = .T.
      .mouchard_texte_ouverture = 3
      = FPUTS(m.gn_handle_logtxt, TTOC(DATETIME())+ CHR(9)+ "-2" + ;
       CHR(9) + "mouchard texte déjà ouvert !!"+ STR(m.gn_handle_logtxt,4))
    ENDIF && m.gnmouchardtxt > -1
  ENDIF && VARTYPE(m.gn_handle_logtxt)=="N"
ENDIF && ISNULL(m.gn_handle_logtxt)
```

Voici un exemple de méthode d'écriture dans le mouchard:

```
PROCEDURE mouchard( pnevenement AS Number, pcform AS String, ;
 pcmethode AS String, pcfamille AS String, pctexte AS String, ;
  pnvaleur AS Number)
  * on écrit une ligne dans le mouchard
  * pnevenement : numéro de l'événement
  * pcform : formulaire / programme courant
  * pcmethode : méthode / procédure courante
  * pctexte : texte libre
  * pnvaleur : valeur libre  non obligatoire
  * pcfamille : code de la famille + éventuellement code de l'enfant
  *    (on accepte 'B4030', 'B403001','B4030-01' ou 'B4030.01')
  *    (pcfamille peut être vide mais doit exister)
  IF This.mouchard_texte_ya
    = FPUTS(m.gn_handle_logtxt, TTOC(DATETIME())+ CHR(9)+ ;
     IIF(BETWEEN(m.pnevenement, -999,999), " ", "")+ ;
      ALLTRIM(STR(m.pnevenement,7))+ ;
       CHR(9) + m.pcform+ CHR(9)+ m.pcmethode+ CHR(9)+ ;
        m.pcfamille+ CHR(9)+ m.pctexte+ CHR(9)+ ;
         IIF(PCOUNT()<6, "0", ALLTRIM(STR(m.pnvaleur,14,3))))
  ENDIF && This.mouchard_texte_ya
ENDPROC && mouchard( pnevenement AS numeric, pctexte AS string)
```

NB : le test sur la tranche -999,999 ne sert que pour un meilleur alignement

Et voici ce que cela donne (41 secondes de déroulement) :

```
23/05/2010 16:08:15  -1      ouverture du mouchard
23/05/2010 16:08:15  -84     erreur_maurice_envoi = .F.
23/05/2010 16:08:15  -81     aucune @ pour l'envoi des erreurs
23/05/2010 16:08:15  -30     mémoire foreground initial: 1465122816
23/05/2010 16:08:15  -31     mémoire foreground limitée: 134217728 (128Mo)
23/05/2010 16:08:15  -32     mémoire background initial: 366477312
23/05/2010 16:08:15  -33     mémoire background limitée: 50331648 (48Mo)
23/05/2010 16:08:15  70600   structures 02 février 2010       structures 02 février 2010              début structures        1,000
23/05/2010 16:08:16  -40     2 bdd : bdd 'tables et bdd 'vues'
23/05/2010 16:08:16  -4      bdd elv déjà ouverte
23/05/2010 16:08:16  -6      opérateur par défaut 00
23/05/2010 16:08:16  1       elvclass    init                démarrage 58096,863
23/05/2010 16:08:16  10      welv.prg    principal           démarrage welv      58096,867
23/05/2010 16:08:16  -20     targette 2555006 -1
23/05/2010 16:08:16  11      welv.prg    principal           avant do initial    0,000
23/05/2010 16:08:16  20      elvinitw    initial             début ouverture tables communes         0,000
23/05/2010 16:08:16  26      elvinitw    initial             répertoire fichiers : E:\MAU\PROFESS9\ELEVES9\  12509,000
23/05/2010 16:08:19  27      elvinitw    initial             R2009-2010 03/09/2009-02/07/2010        2009,000
23/05/2010 16:08:19  44      elvinitw    initial             opérateur 00        0
23/05/2010 16:08:19  -400    mise à jour smtp
23/05/2010 16:08:19  70      elvinitw    initvar             ANNEE SCOLAIRE PARAMETREE: 2009-2010 date courante: 23/05/20100
23/05/2010 16:08:19  24      elvinitw.prg        radprevu            aucune radiation prévue         0
23/05/2010 16:08:23  77774   principaleleve       init_commun                 Gestion des élèves      1420,000
23/05/2010 16:08:26  11300   gestion_tables_parents.prg->parentan    init    B1024   instanciation classe parentan R
23/05/2010 16:08:26  11300   gestion_tables_parents.prg->parentan    init    B1024   instanciation classe parentan S
23/05/2010 16:08:26  112     welvecr1 (principaleleve)   majfamille  B1024   02 Guillaume            9532,000
23/05/2010 16:08:26  143     welvecr1 (principaleleve)   majfamille  B1024   lecture enfant 02 Guillaume     0
23/05/2010 16:08:26  115     welvecr1 (principaleleve)   majfamille  B1024   avant addobject 'enf' . 0
23/05/2010 16:08:26  12000   gestion_classes_normal      init                initialisation gestion_classes_normal       0
23/05/2010 16:08:26  12011   gestion_classes_normal      ouverture           ouverture normal        0
23/05/2010 16:08:26  12010   gestion_classes_normal      ouverture           R T         518,000
23/05/2010 16:08:26  12012   gestion_classes_normal      ouverture           lcla__2XX0YLFO1     14,000
23/05/2010 16:08:26  12100   gestion_classes_detail      init                initialisation gestion_classes_détail       0
23/05/2010 16:08:27  12117   gestion_classes_detail      ouverture           ouverture detail        0
23/05/2010 16:08:27  400     cntenfant   init        B1024-02    R,enf: B102402 Guillaume, .ntf: B102402     0
23/05/2010 16:08:27  12100   gestion_tables_enfant.prg->eleve_1an    init    B102402 instanciation classe eleve_1an R 0
23/05/2010 16:08:27  13000   cntenfant   init        B1024-02    classe réelle: CM2D, code : 10110       58107,185
23/05/2010 16:08:42  12199   gestion_classes_detail      destroy             destroy gestion_classes_détail      0
23/05/2010 16:08:42  12144   gestion_classes_detail      fermeture           début fermeture détail0
23/05/2010 16:08:42  12099   gestion_classes_normal      destroy             destroy gestion_classes_normal  0
```

```
23/05/2010 16:08:42  12035    gestion_classes_normal          fermeture              début fermeture normal          0
23/05/2010 16:08:42  12199    gestion_tables_enfant.prg->eleve_1an    destroy    B102402   destroy classe eleve_1an R    2009,000
23/05/2010 16:08:42  11399    gestion_tables_parents.prg->parentan    destroy    B1024     destroy classe parentan S     0
23/05/2010 16:08:42  11399    gestion_tables_parents.prg->parentan    destroy    B1024     destroy classe parentan R     0
23/05/2010 16:08:42   125     principalelve          Destroy event               16:08:42  58122,891
23/05/2010 16:08:55   89      welv.prg    principal  CLEAR EVENTS                58135,952
23/05/2010 16:08:55   88      welv.prg    principal  ##########################  0,000
23/05/2010 16:08:56   90      terminer_elv                       début terminer_elv    58136,071
23/05/2010 16:08:56   92      terminer_elv                       fermeture des tables  5,000
23/05/2010 16:08:56   36      terminer_elv                       Pas d'archivage: environnement de développement    0
23/05/2010 16:08:56  -99      fermeture du mouchard
```

L'essayer, c'est l'adopter !

Jean à Grenoble