

## Blog de développement

### A l'écoute des mails (Listener)

Un changement majeur et profond a eu lieu, et on la voit de la façon la plus flagrante au niveau de la Logistique. Certes, nos fournisseurs voulaient bien, il y a quelques années, nous informer d'un numéro de suivi sur l'interface Internet que nous leur offrons. Il y a alors alors correspondance 1 à 1 entre l'information logistique du fournisseur et le dossier du client. L'information migre et le client est informé par nous de l'expédition de la commande. Ca, c'était avant...

Aujourd'hui, le fournisseur a fourni votre adresse email au transporteur (ex: UPS) qui vous informe par mail, et, s'en tient là. De même, il voulait encore bien cliquer sur un lien pour confirmer un délai. Aujourd'hui il vous envoie un A/R, avec un PDF en PJ et c'est tout!

De même nos clients prennent le pli d'attendre que l'information apparaisse sur leur téléprompteur (aka. messagerie). Vous risquez de passer un temps fou à pallier à ce problème alors que de nouvelles solutions existent.

### Mettre le Serveur à l'écoute de la boîte Mail

IBM décrit un Listener qui semble écouter, trier, classer des demandes de support (avec gestion de ticket) [Lire →](#).

On identifie plusieurs étapes: récupérer les mails, les filtrer/trier, digérer, créer une action, répondre au client. Certaines sont banales, d'autres assez spécifiques. Par exemple reconnaître un numéro de suivi chez GLS, dépendra de la langue, pourra changer au grès des modifications chez GLS, des pays, etc. La personnalisation d'un package "clé en main" ne me paraît pas évidente. Les points durs sont le rapatriement de la boîte mail, le détachement des pièces jointes, la 'lecture' des PDF (ex: commandes, accusés de réception de commande...), le filtrage, sa sensibilité et sa spécificité.

## Télécharger sa boîte POP3

Il y a pléthore d'outils commerciaux... On peut aussi instrumentaliser Outlook, qui est lourd et fait crasher le serveur. J'évite... Un point dur est le détachement des pièces jointes, point sur lesquels les solutions proposées ne sont pas toujours très claires. On veut rapatrier les mails, les sauvegarder, les 'lire', y compris les pièces jointes: c'est possible!

Les transporteurs (ex: UPS) envoient des mails sans pièce jointe. Un bon vieux (trop vieux ?) composant comme Dypso POP3, ferait l'affaire. Je mets le composant, aujourd'hui disparu, dans le ZIP de ressources. [Télécharger →](#). On trouve encore le manuel. Ca marche pour la logistique, mais on ne peut pas extraire les pièces jointes, ni lire plus de 20 mails présents sur le serveur: il faut consulter souvent.

Fetch ou getmail6 sont uniquement pour Linux. Un vieux getmail par Tim Charron a les fonctions qui m'intéressent [Lire et Télécharger →](#). Il peut extraire automatiquement les fichiers binaires codés MIME ou UU lors du téléchargement. Ne fonctionne qu'avec POP3. On peut contourner le problème en renvoyant automatiquement sur une boîte mail aveugle, en POP3. L'utilisation de Thunderbird avec l'extension FiltaQilla est à considérer aussi.

Mon Listener fait appel à de nombreuses techniques: CURL, [MUNPACK](#), UUDECODE, PDFTOTXT, TESSERACT, etc.. Je pense même à intégrer Chat GPT et DEEP-L. Commençons par relever la boîte aux lettres...

## CURL, encore et toujours

C'est un outil versatile! Gratuit, robuste, intégré à Windows, populaire dans le monde Linux: il fait aussi POP3, IMAP et ...Gmail (? [à vérifier...](#)). [Lire le Manuel →](#). La commande intégrée de Windows a quelques rares limitations, donc, si c'est possible, on installera la dernière version. [Télécharger →](#)

On initialise nos valeurs... A noter que Curl ne crée pas les fichiers de sortie *en tant que de besoin*, il faut les avoir créés d'entrée de jeu.

```
Set objFSO = CreateObject("Scripting.FileSystemObject")
```

```

sDossierComptesFolderPath = "C:\Users\MonServeur\Documents\POP3\Comptes\" ' // à p
sMessagesPresentsFileName = "Messages_presents_sur_le_serveur_distant.txt"
sAlreadyDownloadedFileName = "Already_Downloaded.txt"
sATelechargerFileName = "A_Telecharger.txt"
' // On définit les comptes à surveiller.
Dim ArComptes (10, 4)
ArComptes (0,0) = "granuloshop"           'Nom de dossier
ArComptes (0,1) = "pop3.free.fr"          'serveur
ArComptes (0,2) = "granuloshop"          'utilisateur
ArComptes (0,3) = "xxxxxxx"              'mot de passe

ArComptes (1,0) = "manager"
ArComptes (1,1) = "mail.granuloshop.com"
ArComptes (1,2) = "manager@granuloshop.com"
ArComptes (1,3) = "xxxxxxx"
' // etc.
For i = 0 to 9
    If ArComptes (i,0) <> "" then
        If NOT objFSO.FolderExists(sDossierComptesFolderPath & ArComptes (i
        If NOT objFSO.FolderExists(sDossierComptesFolderPath & ArComptes (i
        If NOT objFSO.FolderExists(sDossierComptesFolderPath & ArComptes (i
        ' // CURL ne créera pas les fichiers de sortie: il faut s'assurer c
        ' // CURL va redocumenter le fichier de messages présents sur le se
        If NOT objFSO.FileExists(sDossierComptesFolderPath & ArComptes (i,0
        ' // Eventuellement, si on souhaite un log pour debugger ...
        ' // If NOT objFSO.FileExists(sDossierComptesFolderPath & ArComptes
    End If
Next

```

*Une fois au plus, on pourra vouloir consulter les services proposés par le serveur distant, en particulier la capacité à enrichir la liste des messages présents d'un identifiant unique: le nom est UIDL. Voici la commande: `curl.exe -L -v -u username:password pop3://pop.free.fr`*

On va conserver l'historique afin de réduire les requêtes au serveur distant:

```

' // on reconstitue l'historique, si besoin. Si le téléchargement précédent
For i = 0 to 9
    If ArComptes (i,0) <> "" then
        sHistorique = ""
        sFilePath = sDossierComptesFolderPath & ArComptes (i,0) & "\Already
        Set objOpen = objFSO.OpenTextFile(sFilePath, 1, true) ' // 1 = For
        If NOT objOpen.AtEndOfStream then sHistorique = objOpen.ReadAll
    End If
Next

```

```

        objOpen.Close
        Set objOpen = nothing
        Set Folder = objFSO.GetFolder (sDossierComptesFolderPath & ArCompte
        for each File in Folder.Files
            If NOT InStr(sHistorique, Replace (File.Name, ".eml", ""))
        Next
        Set objFile = objFSO.OpenTextFile(sDossierComptesFolderPath & ArCom
        objFile.Write sHistorique
        objFile.Close
        Set objFile = nothing
    End If
Next

```

Curl va interroger le serveur distant, qui va donner une liste de message présents avec un identifiant unique.

```

sBat = ""
For i = 0 to 9
    If ArComptes (i,0) <> "" then
        sBat = sBat & VbCrLf & ""curl.exe"" -X ""UIDL"" -v -o "" & sDossi
    End If
Next
sCurlBatListFilePath = "C:\Users\MonServeur\Documents\POP3\Curl_Bat.bat" ' // a p
Set objFile = objFSO.OpenTextFile(sCurlBatListFilePath, 2,true) ' // 2= ForWritir
objFile.Write sBat
objFile.Close
Set objFile = nothing

Command = """" & sCurlBatListFilePath & """"
oShell.Run Command, 0, True

```

On dispose maintenant de la liste des messages, chacun avec un identifiant unique. On construit une liste de messages à télécharger, en excluant ceux qui ont déjà été téléchargés.

```

"curl.exe -o "" & sDossierComptesFolderPath & ArComptes (i,0) & "\InBox\" & sUID &
".eml"" -u " & ArComptes (i,2) & ":" & ArComptes (i,3) & " pop3://" & ArComptes
(i,1) & "/" & sNumero

```

Dans la commande CURL:

-o désigne le dossier et le nom attribué au mail. On veille à ne pas avoir de

caractères interdits

-u désigne l'utilisateur suivi de : et le mot de passe

- ensuite, le protocole et le serveur suivi du numéro d'ordre du message (non pas son Indentifiant Unique...) Exemple: **pop3://free.fr/Numero**

POP3 est un protocole assez rustique et ancien, il reste accessible avec la commande TELNET.

```
' // on reconstitue l'historique, si besoin. Si le telechargement precedent
For i = 0 to 9
    If ArComptes (i,0) <> "" then
        sHistorique = ""
        sFilePath = sDossierComptesFolderPath & ArComptes (i,0) & "\Already
Set objOpen = objFSO.OpenTextFile(sFilePath, 1,true) ' // ForReac
If NOT objOpen.AtEndOfStream then sHistorique = objOpen.ReadAll
objOpen.Close
Set objOpen = nothing
Set Folder =objFSO.GetFolder (sDossierComptesFolderPath & ArComptes
for each File in Folder.Files
    If NOT InStr(sHistorique, Replace (File.Name, ".eml", ""))
Next
Set objFile = objFSO.OpenTextFile(sDossierComptesFolderPath & ArCom
objFile.Write sHistorique ' & VbCrLf
objFile.Close
Set objFile = nothing

sA_telecharger = ""
sFilePath = sDossierComptesFolderPath & ArComptes (i,0) & "\Message
Set objOpen = objFSO.OpenTextFile(sFilePath, 1,true) ' // ForRea
ObjOpen.Close
Set objOpen = nothing
If NOT objOpen.AtEndOfStream then
    Do While NOT objOpen.AtEndOfStream
        sLine = trim(objOpen.ReadLine)
        If sLine <> "" then
            If InStr(sLine, " ") > 0 then
                nMessage = nMessage + 1
                sNumero = ""
                sNumero = Trim (Left(sLine, InStr(s
                sUID = replace (sLine, sNumero, "")
                sUID = trim(replace (sUID, " ", ""))
                sUID = Normalize(sUID)
                if sNumero <> "" AND InStr(sHistori
                    if sBat <> "" then sBat = s
                    StrBat = StrBat & "curl.exe
                    if sA_telecharger <> "" the
```

```

sA_telecharger = sA_telecha
End If
End If
End If
Loop
End If
End If
Next

```

Sauvegardons notre ligne de commande pour déverminage éventuel. Puis on y va!

```

Set objFile = objFSO.OpenTextFile(sCurlBatMessagesFilePath, 2,true) ' // ForWritir
objFile.Write sBat
objFile.Close
Set objFile = nothing

Command = "" & sCurlBatMessagesFilePath & ""
if sBat <> "" then oShell.Run Command, 0, True

```

## On renomme les mails pour plus de lisibilité

Avec POP3, les mails sont identifiés par un GUID, unique certes, mais illisible: ce n'est pas pratique pour l'inévitable débogage! Donc, à coté de nos dossiers Inbox, nous créons des dossiers Mail, où la nom de fichier est lisible et permet de vérifier les fonctions de filtrage, par exemple.

## Un peu de nettoyage...

## Et ensuite ? Le WorkFlow post

[https://forums.ivanti.com/s/article/Creating-a-workflow-to-send-an-email-when-a-listener-stops-working?language=en\\_US](https://forums.ivanti.com/s/article/Creating-a-workflow-to-send-an-email-when-a-listener-stops-working?language=en_US)

[Previous](#)

[Next](#)

0 Comments

 1 Login ▼

G

Start the discussion...

LOG IN WITH

OR SIGN UP WITH DISQUS 

Name



Share

Best

Newest


Oldest

Be the first to comment.

Subscribe

Privacy

Do Not Sell My Data

Designed with  by [Xiaoying Riley](#) for developers