Documentation des scripts et interfaces réalisés par Johan

Scripts commentés ; Variables à changer en fonction des clients et des setups ; toutes les expressions « entre guillemets » font référence au code.

Interface de contrôle

Interface d’intégration

Script d’automatisation des envois de fichiers le soir

Script de traitement de .xlsx to .ASC

Script de correspondance de code articles

Script de concaténation de « 0 » pour code article

Script d’interfaçage as400 – Kwisatz

# Interface de Contrôle

Interface permettant de vérifier si le service BCOMM/FCOMM est en cours de tâche et écoute le trafic sur les ports 12005 et 12006.

* Langage : Python
* Technologies utilisées : PyQt, Qt Designer, PyInstaller  
   technologies de Threads : concurrent.futures, asyncio.
* Considération : service créé avec NSSM. Changer ligne 35 et 46 si service créé autrement.

# Interface d’intégration

Interface permettant d’automatiser des tâches côté client. Variables à changer dans la classe « Ui\_MainWindow » l.81 à 83. Néanmoins il demande le chemin d’installation de kwisatz et modifie ces variables si nécessaire. Ce programme regroupe 3 fonctionnalités exécutables hors interface :

## Génération des BCA

Permet de récupérer les BCA prsént dans kwisatz. Ouvre une fenêtre où le client choisit la période et autres éléments grâce à l’automate. Cette fonctionnalité ne fonctionne pas sur tous les postes (automate non lancé)

## Importation BCA

Ce code récupère les BCA à « intFilePath », les renomme, les déplaces au chemin import de kwisatz, et ensuite lance l’automate afin que kwisatz les intègre.

## Importation des articles

Ce script récupère articles.dat à « articlesPath» convertit ce fichier en .ASC, et l’intègre à l’aide d’un automate kwisatz.

Même langage et technologies utilisée que pour l’interface de contrôle.

# Script d’automatisation des envois de fichiers le soir

(server.js)

Une fois mit dans une routine, ce code permet de faire remonter par mail les CA, les tickets ventes et des logs du processus, au gérant du magasin. Il envoie aussi un autre mail au support b2M qui concerne l’Etat du processus de remontée.

Il crée un objet Javascript nommé « report » avec au départ une clé « error » qu’il remplit en fonction des erreurs qu’il rencontre. Le programme rajoute des clefs à cet objet tout au long du processus. A la fin, cet objet est utilisé afin d’être envoyé par mail au support B2M.

* Langage : nodejs
* Technologies : bluebird, nodemailer
* Variables : Mail : l.12 à 17 ; Dossier client l.37 38

# Script de traitement de .xlsx to .ASC

(orderReceiptProcessing.py)

Ce programme permet de traiter les BCA en .xlsx et de les convertir en .ASC avec la bonne structure afin que Kwisatz puisse les lire. Le comportement du script se base sur les positions des cases excel afin de savoir comment traiter l’information.

* Langage : Python
* Technologies : xlrd : obsolète et déconseillée maintenant. Utiliser openpyxl si réécriture

# Script de correspondance de code sous famille

(uniqueSFId.py)

Permet de transformer les code sous famille classique en un format unique : « CodeSousFamille » 🡺 « codefamilleZeroPossiblesCodeCodeSousFamille».

* Langage : Python
* Technologies : openpyxl

# Script de concaténation de 0 pour code article

(codebarre.py) (autre version : zeros.py)

Ce programme permet de concaténer les code EAN13 en rajoutant des « 00 » si le code initial est initialement compris entre 10 et 13 char.

Entrée : Articles.csv   
Sortie : output.csv

# Script d’interfaçage as400 – Kwisatz

(MLFWFMToKwisatz.py)

Ce programme prend en entrée des factures et bons de livraison sortant du serveur as400 client et sort des BRA et BLI au format kwisatz.