Procédures

Ce document décrit les procédures utilisées pour manipuler les applications d’AIDER déployées en production. Elles sont déployées chez Krios, sur des machines virtuelles avec Windows Server 2008 et l’accès à ces machines passe par une connexion terminal serveur. La structure de ces machines est identique :

* L’application est dans le dossier C:\aider
* La base de données est dans C:\ProgramData\Epsitec\Firebird Databases
* Les backups sont dans D:\aider\backup

A part les données qui sont dessus et l’usage qu’on en fait, les seules différences qu’il y a entre les machines sont leur adresse IP et la couleur de leur fond d’écran (afin d’éviter les confusions et de pouvoir identifier immédiatement sur quelle machine on est connecté). Les procédures décrites plus loin sont donc strictement les mêmes, quelle que soit la machine qu’on utilise.

Les applications sont décrites dans le tableau ci-dessous :

|  |  |  |
| --- | --- | --- |
| **Application** | **Adresse IP** | **Fond d’écran** |
| Application avec les données réelles | 78.155.25.113 | Vert |
| Application avec les données de test | 78.155.25.115 | Rouge |

# Compiler l’application

La compilation de l’application et sa préparation pour la production se fait à partir du script S:\Epsitec.Cresus\App.Aider\build.bat.

Si on le lance avec -test ou -production sur la ligne de commande pour compiler et configurer l’application de test ou celle de production.

Le résultat est stocké dans S:\Epsitec.Cresus\App.Aider\bin\build\aider.zip et contient tout ce qu’il faut pour lancer AIDER sur le serveur (de test ou de production).

Ce script accède aux solutions suivantes :

* S:\Epsitec\dot.net\Epsitec.ShellPe
* S:\Epsitec\dot.net\Epsitec.SwissPost
* S:\Epsitec\dot.net\Epsitec.TwixClip
* S:\Epsitec\dot.net\Epsitec.ZipMe
* S:\Epsitec.Cresus

Le script passe par les étapes suivantes :

1. Suppression des fichiers ignorés par SVN dans ces dossiers
2. Suppression des fichiers non versionnés par SVN dans ces dossiers
3. Annulation des modifications des fichiers versionnés par SVN dans ces dossiers
4. Compilation des projets et solutions en mode RELEASE.
5. Copie des paramètres de configuration (différents selon les arguments -test ou -production)
6. Compression du résultat dans un fichier zip

Pour sauter les étapes de nettoyage SVN, on peut utiliser l’argument -nocleanup sur la ligne de commande. Il est fortement déconseillé d’utiliser cette option pour faire un build de production. Cela ne devrait être utilisé que dans des cas de tests, le but étant d’avoir un build totalement reproductif.

De plus, même si cela n’est pas strictement nécessaire dans ce script, il est fortement recommandé d’avoir les dossiers S:\Epsitec et S:\Epsitec.Cresus standard. On pourrait lancer le script depuis un autre dossier que S:\Epsitec.Cresus, pour autant qu’il y ait un dossier Epsitec au même endroit, mais il est fort possible que d’autres parties du build aient des chemins absolus S:\Epsitec.Cresus et S:\Epsitec.

Note : la compilation du code Win32 requiert la présence du SDK Windows 7.1A pour les fichiers \*.h et \*.lib, ainsi que les binaires correspondants (par ex. RC.EXE).

# Lancement de l’application

Pour démarrer l’application, il faut lancer :

C:\aider\server\App.Aider.exe -server

Si la base de données est incompatible, la mise à jour n’est pas possible en mode « serveur ». Il faut alors lancer l’application sans l’option -server.

# Fermeture de l’application

Pour fermer l’application proprement, il faut appuyer deux fois sur la touche [ENTER] dans la console de l’application, comme indiqué dans celle-ci.

# Application temporaire

Si pour une raison ou pour une autre, l’application devait être hors-ligne plus de quelques minutes, il est possible de lancer une application qui affiche simplement à l’utilisateur que l’application est hors ligne.

Pour ce faire, on utilise les scripts start.bat et stop.bat du dossier C:\aider\maintenance.

# Mise à jour du système d’exploitation

Lorsqu’une mise à jour du système d’exploitation est disponible, une fenêtre propose de faire la mise à jour. Voici la procédure à suivre :

1. Se connecter à la machine en terminal serveur.
2. Fermer l’application selon la marche à suivre.
3. Redémarrer le serveur pour faire la mise à jour. A ce moment-là, la connexion terminal serveur est interrompue.
4. Attendre que le serveur redémarre et ouvrir une nouvelle connexion terminal serveur.
5. Redémarrer l’application selon la marche à suivre.

# Mise à jour de l’application

Voici la procédure à suivre pour faire une mise à jour de l’application. Notez bien qu’il faut effectuer la même procédure sur pour le serveur de test et pour le serveur de production.

1. Mettre les dossiers S:\Epsitec et S:\Epsitec.Cresus à la bonne révision
2. Builder l’application avec le script build.bat
3. Uploader la nouvelle application sur le serveur
4. Fermer l’application qui tourne sur le serveur
5. Faire une copie des fichiers de l’application (C:\aider) et du fichier de base de données (C:\ProgramData\Epsitec\Firebird Databases\AIDER.FIREBIRD) pour avoir un backup à disposition et pouvoir immédiatement revenir dans un état fonctionnel si quelque chose tourne mal.
6. Remplacer la version actuelle de l’application par la nouvelle version
7. Si la mise à jour comprend une modification des schémas de la base de données, lancer l’application sans l’argument -server sur la ligne de commande pour faire la mise-à-jour du schéma de la base de données
8. Lancer l’application
9. Tester que l’application fonctionne correctement
10. Supprimer les fichiers sauvegardés à l’étape 4)
11. Faire un tag de la révision du dépôt SVN vers [tags/aider/aider-[AA].[MM].[JJ](https://svn.opac.ch/svn/cr/tags/aider/aider-%5bANNEE%5d.%5bMOIS%5d.%5bJOUR)] pour garder une trace de cette mise à jour ; le message est du style :  
    - Tag pour le déploiement du 19.06.2013.
12. Envoyer un email avec les modifications aux personnes concernées, à savoir les développeurs de l’application et personnes de contact chez l’EERV qui sont responsables de l’application :
    1. Jean-Michel Sordet (responsable du projet à l’EERV) : *jean-michel.sordet@eerv.ch*
    2. Gérard Jaton (responsable de l’informatique à l’EERV) : *gerard.jaton@eerv.ch*

# Import de données paroissiales

Pour importer un fichier de données dans AIDER, il faut faire deux étapes : préparer le fichier à importer et ensuite l’importer.

La préparation du fichier se fait selon la procédure suivante :

1. Obtenir les fichiers à importer
2. Copier et renommer dans Samples > EERV Région 7 > 7100 (R7, paroisse 10 : numéro repris du fichier livré par l’EERV) > person.xlsx/group.xlsx/supergroup.xlsx/activity.xlsx
3. Compléter ou créer le fichier avec les numéros d’identifications (selon un exemple des fichiers id.xlsx sur SVN) qui fait le lien entre les ID de paroisse et leur nom. Attention au type (P = paroisse, R = région, C = canton).
4. Ouvrir et sauver chaque fichier \*.xlsx à importer avec une version d’Excel pour Windows, pour éviter des problèmes avec des fichiers qui viendraient d’Excel Mac.
5. Faire une des importations de test jusqu’à ce que la procédure aboutisse, en corrigeant à chaque fois le fichier un peu plus.
   1. Appeler TestFullImportJob depuis AiderProgram.TestFullImport.
   2. Éditer les appels à AiderProgram.Test.  
      Il est possible de désactiver la comparaison de la date de naissance/sexe pour augmenter le taux de match.
   3. Dans la méthode Test, commenter l’appel à Import.
   4. Lancer avec -testfullimport.
   5. Corriger les localités incorrectes, par ex. « 1462 Arrissoules » 🡪 « 1463 Rovray ». (<http://www.post.ch/db/owa/pv_plz_pack/pr_main?p_language=fr>).  
      Cette modification se fait dans le fichier Excel.
   6. Ajouter FR/DE, etc. dans la colonne du pays si c’est une localité à l’étranger.
   7. Enlever le commentaire sur l’appel à Import.

Les problèmes communs rencontrés dans les fichiers qui sont corrigeables facilement sont :

* La colonne pays est souvent vide. Il est facile de la compléter en observant la console de l’application qui affiche chaque localité qu’il n’a pas pu corriger automatiquement. Les localités étrangères apparaissent toujours dedans, sauf si une ville étrangère à le même nom et le même numéro postal qu’une localité suisse.
* Corriger les localités qui n’ont pas pu être corrigées automatiquement où qui ont été mal corrigées. On peut les obtenir dans la console de l’application
* On a parfois des valeurs incohérentes ou inattendues dans certaines colonnes, qu’on peut corriger à la main.

Il y a aussi des fois des problèmes plus difficiles ou impossible à corriger :

* Une colonne avec des données importante est vide (c’est arrivé pour les colonnes localités, numéro postal et nom de rue). Dans ce cas, il faut demander un nouveau fichier avec les données manquantes.
* Corruptions dans les numéros de groupes, avec des groupes qui ont deux fois le même numéro. Les cas que j’ai eus étaient corrigeables à la main, mais de manière fastidieuse.
* Mauvais numéros de paroisses. On peut parfois les corriger à la main, mais c’est fastidieux.

Il y a quelques points auxquels faire attention dans la console :

* On imprime à chaque fois les corrections de localités. Il est essentiel que ces corrections soient justes
* On imprime à chaque fois les corrections de rues et ensuite les rues qui n’ont pas pu être corrigées. Un grand nombre de rues qui ne peuvent pas être corrigées peuvent indiquer un problème plus grave
* On affiche comment a été fait le matching des personnes. Il est raisonnable d’avoir un quart des personnes qui ne sont pas matchées. Quand il y a plus, il y a peut-être une raison, mais il est prudent de regarder :
  + Il peut y avoir beaucoup de personnes qui ne sont pas déjà dans AIDER, par exemple une paroisse peut avoir la liste de tous les catholiques qui y habitent, ce qui peut faire beaucoup de monde.
  + Il peut manquer la colonne sexe ou date de naissance. Dans ces cas, l’algorithme a de la peine à trouver les correspondances. Il est possible de ne pas tenir compte de ces deux colonnes, mais le nombre de faux-positifs risque d’augmenter. A faire avec prudence.
  + Les données des familles ne sont pas rentrées dans le fichier. C’est souvent le cas de fichiers régionaux qui ne sont qu’une liste de personnes sans rattachement à des familles. Dans ce cas, on ne peut pas utiliser les informations de familles pour mieux faire le matchage.
* On affiche les personnes de la base de données qui sont matchées par plusieurs personnes du fichier. Cela arrive quand on a des doublons dans le fichier. Il est toujours bien de vérifier qu’il s’agit bien de doublons et pas qu’il y ait trop de faux-positifs.

L’importation du fichier se fait selon la procédure suivante :

1. Modifier le fichier C:\aider\server\Aider.Environment.crconfig pour mettre l’application en lecture seule à son prochain démarrage
2. Fermer l’application
3. Faire une copie du fichier de la base de données (copier depuis Explorer)
4. Relancer l’application
5. Vérifier que l’application fonctionne correctement et en mode lecture seule
6. Télécharger la copie du fichier de la base de données sur sa machine locale
7. Effectuer l’importation des données sur sa machine locale, en utilisant les arguments appropriés sur la ligne de commande pour lancer l’application
8. Copier la base de données de la machine locale sur le serveur
9. Modifier le fichier C:\aider\server\Aider.Environment.crconfig pour permettre l’écriture des données lors du prochain démarrage de l’application
10. Fermer l’application
11. Faire une copie du fichier de base de données du serveur, pour avoir un backup sous la main au cas où
12. Remplacer le fichier de base la base de données du serveur par celle qui vient de la machine locale
13. Relancer l’application
14. Vérifier que tout fonctionne correctement
15. Supprimer le backup fait au point 11)
16. Envoyer un email avec l’information sur l’importation aux personnes concernées. En général, je fais un email aux autres développeurs de l’application et aux personnes de contact chez l’EERV qui sont responsables de l’application :
    1. Jean-Michel Sordet (responsable du projet à l’EERV) : jean-michel.sordet@eerv.ch
    2. Gérard Jaton (responsable de l’informatique à l’EERV) : gerard.jaton@eerv.ch

# Restore d’une base (en local)

Reprendre un des fichiers trouvés sur le serveur dans D:\aider\backup (\*.gbak.gz) et le déposer dans le dossier local C:\ProgramData\Epsitec\Firebird Databases. Exécuter en local AIDER avec l’option -db-restore aider-backup-2013-06-13-04-00-52.gbak.gz (par exemple) pour que le fichier soit décomprimé, ce qui produit un fichier AIDER.FIREBIRD en sortie.

# Job

Pour effectuer un job quelconque sur la base de données, il faut suivre la même procédure que pour l’importation de données paroissiales, en modifiant simplement l’étape 7).

# Mise à jour de Data.Platform

Le projet Data.Platform intègre des données de La Poste suisse (MAT[CH]sort, etc.) et d’autres sources. Au démarrage du serveur une vérification des données MATCH à disposition sur le site de la poste est effectuée. Si un nouveau jeu de données est disponible, le client Match va télécharger les données et les dé zipper sur le serveur dans le dossier :

Users\XXXXXX\AppData\Roaming\Epsitec  
  
Ce cache local des données match contiens différents fichiers :

swisspost.csv la copie des données Match la plus récente

swisspost.meta contient la dernière version utilisée par le client Match

swisspost|house|street|zip .csv des sous-ensembles de données Match produit par l’ETL depuis la copie des données Match.  
  
Ces fichiers sont utilisés pour alimenter les différents dépôts de Data.Platform.   
Ils sont reconstruits à chaque mise à jour Match.  
  
En cas de problème avec la mise à jour on peut essayer d’effacer ces fichiers pour reconstruire le cache au prochain démarrage.

# Exportation « Bonne Nouvelle »

La procédure pour faire une exportation des abonnements Bonne Nouvelle pour l’éditeur du journal est la suivante :

1. Fermer l’application
2. Faire une copie du fichier de la base de données
3. Relancer l’application
4. Télécharger la copie du fichier de la base de données sur sa machine locale
5. S’assurer que Data.Platform a été mis à jour (voir ci-dessus) et copier le fichier MAT[CH]sort le plus récent avec le nom MAT[CH]sort.csv dans le dossier de l’exécutable (typiquement S:\git\core\cresus\Epsitec.Cresus\App.Aider\bin\Debug).tool.data
6. Effectuer la création du fichier d’export sur sa machine locale, en utilisant les arguments appropriés sur la ligne de commande pour lancer l’application :

App.Aider.exe -exportsubscriptions -output:export.txt -error:error.txt

1. Uploader le fichier d’export vers le serveur de l’éditeur, en utilisant les arguments appropriés sur la ligne de commande pour lancer l’application. Il faut obtenir la **date de l’édition** du journal auprès de Vincent Volet ou Paolo Mariani :

App.Aider.exe -uploadsubscriptions -input:export.txt -publicationdate:2013-06-27 -response:response.txt

1. Envoyer un email de confirmation aux personnes concernées, avec les statistiques d’envois, région par région (sauvé dans un fichier export.txt.log).  
   Personnes concernées :
   1. Jean-Michel Sordet (responsable du projet à l’EERV) : [jean-michel.sordet@eerv.ch](mailto:jean-michel.sordet@eerv.ch)
   2. Vicent Volet et Marie-Noëlle Aubert (Bonne Nouvelle, OIC) :  
      [vincent.volet@eerv.ch](mailto:vincent.volet@eerv.ch), [marie-noelle.aubert@eerv.ch](mailto:marie-noelle.aubert@eerv.ch)
   3. Gérard Jaton (responsable de l’informatique à l’EERV) : [gerard.jaton@eerv.ch](mailto:gerard.jaton@eerv.ch)
   4. Paolo Mariani (responsable du BN à l’EERV) : [paolo.mariani@eerv.ch](mailto:paolo.mariani@eerv.ch)
   5. Philippe Martin (responsable du BN chez Tamedia) : [philippe.martin@sr.tamedia.ch](mailto:philippe.martin@sr.tamedia.ch)
2. Avertir le service informatique de Tamedia ([informatique.syscom.reseaux@sr.tamedia.ch](mailto:informatique.syscom.reseaux@sr.tamedia.ch)) que le fichier a été transféré.

# Backups

L’application fait régulièrement des backups qu’elle stocke dans D:\aider\backup. Il n’y a pas de mécanisme automatique de nettoyage des backups. Il faut le faire à la main avant que le disque D ne soit rempli.

# Traitement des données locales

Lorsqu’il faut exécuter un processus de traitement des données locales, j’ai pris l’habitude de faire selon la méthode suivante.

1. Je fais le traitement en lançant l’application depuis Visual Studio avec le débuggeur attaché. De cette manière, on peut plus facilement trouver l’origine du problème s’il y en a un qui se produit. En général, je produis une méthode main sur mesure, qui fait potentiellement plusieurs traitements (comme l’importation de plusieurs fichiers).
2. Pour pouvoir lancer l’application depuis Visual Studio tout en continuant de travailler, il faut procéder de la manière suivante :
   1. Avoir une deuxième copie du repository SVN, par exemple dans S:\Epsitec.Cresus\_WORK. Il faut faire très attention à ce que ce repository ait la révision SVN désirée.
   2. Modifier le fichier Cresus.Core.Library.crconfig pour utiliser un autre fichier de base de données, comme AIDER-WORK.FIREBIRD par exemple.

De cette manière on peut avoir plusieurs instances de Visual Studio qui font plusieurs traitements, tout en continuant à travailler.