Titre: Déploiement des applications: partie cliente
Version: 1.1
Dernière modification: 2008/11/03 17:25
Auteurs: Aurélien Minet <aurelien dot minet at cocktail dot org>, Hugues Villesuzanne <hugues dot villesuzanne at cocktail dot org>
Statut: version finale
Remplace: Installation postes clients

Remarques:

- si le contexte est le PGI Cocktail, il n'a rien de spécifique à Cocktail (mis à part Zap) sur le plan technologique
- connaître Java (JVM, Class, exceptions mais pas besoin de savoir développer)
- avoir un minimum de connaissances sur Oracle (Net) et WebObjects.
- au niveau de la typographie: le code sql, les lignes de commandes, le contenu des fichiers sont en italique

Déploiement des applications: partie cliente

Objectif:

- installer le nécessaire en fonction des applications sur le poste client
- obtenir les connaissances nécessaires pour faire fonctionner les divers applications et résoudre les problèmes liés à la configuration du poste client

Introduction:

Ce document vise à fournir les informations nécessaires pour la mise en œuvre de l'ensemble des applications du PGI Cocktail sur les postes clients.
C'est d'abord une "fiche recette", pour l'équipe de gestion du parc informatique mais aussi une documentation qui aborde la résolution de problèmes, elle est donc également à destination des informaticiens qui déploient les applications surtout que même si le document traite principalement du poste client, la partie serveur n'est pas occultée pour autant.

1. Les applications

Deux types d'applications:

- les lourdes, basées sur un dialogue direct entre le client et la base de données.
- les légères, basées sur une architecture 3 tiers (client / serveur d'applications / serveur de base données). Il y a séparation entre la présentation, le traitement et le stockage des données: "L'intelligence" de l'application réside sur serveur d'application, la partie client ne fait que de la présentation.

Les applications légères existent sous 2 formes:

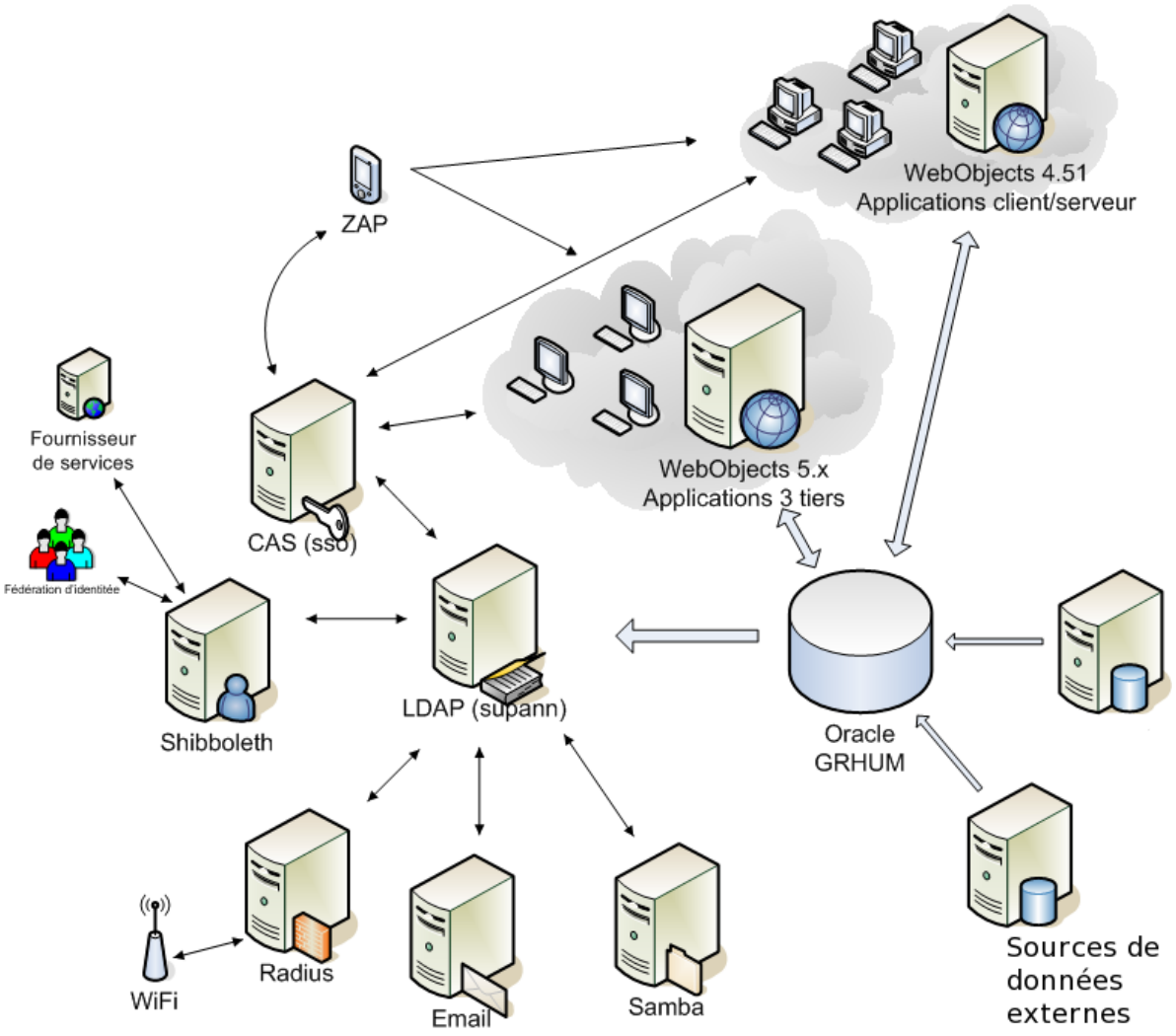
- Application légère web (connexion avec un navigateur web comme Mozilla Konqueror)
- Application légère java en mode :
    - "Java clients" (lacement avec l'exécutable java)
    - "Java Web Start" (lancement avec l'exécutable javaws qui lance java)

Les applications Java Client / Java Web Start sont identiques au final, il n'y a que le mode de chargement qui change. Une application Java-Client a besoin de classes java notamment celles de WebObjects (elles sont regroupées dans ClientSTD.jar) alors qu'une application Java Web Start a besoin d'un fichier jnlp (Java Network Lauch Program) qui indique les ressources nécessaires (fichiers jar, xml...) au lancement de l'application devant être téléchargées.

L'architecture du PGI Cocktail:

Elle est centrée sur GRHUM qui est le schéma central d'une base de données Cocktail.
Cette base est alimentée par des applications (Profil, Annuaire, Mangue...) mais elle peut l'être également par des bases externes (Apogee...) via un "connecteur".
Elle peut produire un annuaire LDAP (à la norme supann) qui peut être utilisé par divers services. Notamment pour l'authentification avec Central Authentification Service (CAS) qui permet de faire du SSO.

Schéma d'une architecture type:



A noter que l'application ZAP (application pur java, disponible qu'en Java Web Start) est plus que recommandée car elle règle les problèmes de raccourcis et permet une authentification unique (SSO) grâce au serveur CAS. En effet elle joue le rôle de "proxy-cas" (en cours d'évolution). Dans le cadre de l'utilisation d'une PKI elle peut aussi effectuer l'authentification via un token USB du type rainbow ikey et code pin.
Il faut ajouter que l'application ZAP n'est pas limitée à Cocktail.

2. Installation:

2.1 Applications lourdes:

Elles dépendent de :

- Apple WebObjects 4.51 (java 1.1 embarqué)
- Oracle Client 10g (8i minimum)
- Adobe Acrobat (Reader, affichage des impressions)

Elles fonctionnent sous Microsoft Windows NT (>=4) (Vista non testé)

Installer WebOjects 4.5.1:

- "from scratch":
  A partir des CD de WebObjects, installer la version 4.51, télécharger chez Apple l'update 4 et l'installer, mettre a jour le JDK présent dans C:\Apple\Library\JDK avec le jdk 1.1.8_009, installer PaletteFrameWork45, ajouté le utilitairecri.zip dans le classpath...ce qui est laborieux.
- depuis le package fournit (install_patched_4.5.1.zip):
  - dézipper dans C: , il doit y avoir avoir C:\Apple, c:\add et 2 raccourcis dans C:\Documents and Settings\All Users\Menu Démarrer\Programmes\Démarrage
  - "définir l'environnement":
    - double-cliquer sur les 3 fichiers .reg pour ajouter les 2 daemons Apple au comme service et ajouter les variables d'environnement Apple.
    - dans les propriétés système : avancé : Variables d'environnement il faut modifier les variables système CLASSPATH et PATH en ajoutant (à la suite de l'existant) le contenu des 2 fichiers .txt correspondants.
      (si vous n'avez pas CLASSPATH il faut donc la créer)
  - enfin il faut rebooter ...
  - Après le reboot et ouverture de session vous devez avoir :
    - pbs.exe et WindowServer.exe exécutés sous votre session
    - nmserver.exe et machd.exe exécutés en tant que system

Lorsque l'on utilise un server TSE / Citrix, étant donné que plusieurs pbs/WindowServer seront exécutés en même temps il est nécessaire que chaque utilisateur ait une variable d'environnement NSSESSIONNAME de définie à une valeur unique (son login, ce qui implique une seule session). Dans c:\add il y a logon.vbs. C'est un exemple, le script regarde si la variable existe, si ce n'est pas le cas elle est créée et la session est fermée.

Installer Oracle Client:

Choisir d'installer la version 10g/11g (8i minimum) dans c:\Oracle et installer le minimum (Sqlnet Sqlplus, éventuellement le driver ODBC), sélectionner comme protocole TCP/IP et pour la résolution de nom tnsname.
Modifier le path pour que les répertoires de Windows (%systemroot% ....) soient les premiers, mettre ceux d'oracle en dernier.
Modifier au besoin les fichiers sqlnet.ora tnsname.ora dans C:\oracle\oraXX\network\ADMIN où utiliser NetCA
Attention donner les droits de lecture et d'accès aux fichiers et à ce répertoire aux utilisateurs
(Il est possible de centraliser le configuration dans OID ou dans un serveur LDAP à partir des clients 9i)

Problèmes:

- pbs et windowserver ne sont pas lancés. pbs plante parfois ce qui bloque le lancement des applications, le tuer et le relancer.
- jar manquant ou inaccessible car chemin dans le CLASSPATH est faux.
- jar en trop (ClientSTD.jar ne doit pas être dans le CLASSPATH) ou jar fournissant une version supplémentaire d'une classe (java ne sait pas choisir celle à utiliser)
- la variable NEXT_ROOT n'est pas positionnée sur C:\Apple
- mauvaise version de Java.
- la variable PATH est incorrecte, elle doit commencer par les chemins windows, puis ceux des application en avant dernier ceux d'Apple et ensuite ceux d'oracle.
- en cas de fenêtre d'erreur Oracle avec login/password/service faire annuler pour obtenir le message d'erreur (ORA-xxxx) qui après une recherche dans google informe sur le problème.

Remarques:

- pbs n'est pas ce qu'il y a de plus stable, ne pas hésiter à le relancer (il y a une option dans Zap pour cela).
- Ne pas hésiter sous Windows à aller dans l'observateur d'évènements afin de regarder les éventuels messages d'erreurs (ceux sur les fontes sont à ignorer).
- On peut utiliser l'application Shell.app (dans C:\Apple\Demos) pour avoir des logs plus verbeux.
  Taper dans le prompt "*defaults write NSGlobalDomain EOAdaptorDebugEnabled YES*" puis aller dans le répertoire de l'application et l'exécuter./App.exe pour avoir les logs de l'EOAdaptor , mais attention il faut penser à désactiver le debug"defaults write NSGlobalDomain EOAdaptorDebugEnabled NO"

2.2 Applications légères:

Pour les applications web il faut juste un navigateur http correct. Pour les applications web AJAX le navigateur de référence est Firefox 2 (tests en cours pour Firefox 3).

Pour les applications légères Java, il y a donc besoin de java (la version 1.4.2 reste celle de référence). Il y a 2 méthodes pour lancer l'application:

- lancement via ClientSTD.jar (Lors de l'installation de WebObjects 4.51 ce jar a été copier dans C:\Apple\Jar) pour faire du Java Client (non recommandé pour le déploiement)
- lacement par Java Web Start qui pour résumé est comme le lancement via ClientSTD.jar sauf qu'il est dynamique car les jar nécessaires (ou autres ressources) sont téléchargés au lancement.

A noter que la détection par Java Web Start des JVM disponibles si elles sont enregistrées est possible (voir Automatic Detection of JREs http://java.sun.com/j2se/1.4.2/docs/guide/jws/relnotes.html)

Installer Java:

Télécharger le j2re 1.4.2 sur le site de sun: http://java.sun.com
Dans le panneau de configuration on a donc un icone Java plug-in qui permet de configurer le "comportement" de java (mais pas de Java Web Start). Pour identifier les problèmes java il faut cocher l'affichage de la console et de la boite de dialogue d'exceptions.
On dispose de plusieurs exécutables dans le répertoire bin, notamment java.exe qui interprète les binaires java(.exe) et javaw(.exe) qui fait la même chose que java mais de manière silencieuse (non verbeuse).
Dans le répertoire javaws, il y a javaws.exe qui lance les application Java Web Start grâce aux fichiers jnlp. Exécuter javaws pour configurer des options plus spécifiques. Aller dans le menu puis dans fichier et choisir préférences.

- Dans l'onglet raccourcis, cocher "Ne jamais intégrer une application java sur le bureau"
- Dans l'onglet avancé, cocher "Afficher la console Java" et changer le "dossier des applications" qui est en fait le répertoire de cache, mettre c:\temp\sun (ou tout autre répertoire avec un chemin court) et donner le contrôle total a tout le monde.

Remarques:

- Attention certains des paramètres n'ont une qu'une portée limitée: l'utilisateur courant et non tout le système (et donc les autres utilisateurs). Pour plus d'informations lire J2SE deployment guide et Java Web Start developers guide .
- Pour vérifier la version de java, ouvrir un shell (cmd.exe sous windows) et taper java -version.

Java Client:

Le lancement s'effectue avec une commande basée sur le modèle :
java –cp chemin_du_client/ClientSTD.jar com.webobjects.eoapplication.client.EOClientApplicationSupport -applicationURL http://nom_du_serveur/cgi-bin/WebObjects/App.woa
Il faut donc connaître l'URL où est disponible l'application (ou éventuellement le serveur où se trouve l'application et le port pour une connexion directe).

- Pour windows, il suffit de créer un raccourci vers javaw (pour ne pas être en mode verbeux) et passer les arguments comme dans le modèle avec le chemin vers ClientSTD.jar, l'URL de l'application. Comme répertoire d'exécution choisir le répertoire où est ClientSTD.jar
- Sous Unix (BSD*, GNU Linux/Hurd ...), faire un script shell contenant la ligne de commande, éventuellement changer le path. (inutile de préciser qu'il faut avoir un display X11)
- Sous MacOSX il est possible de «packager» les applications Java pour en faire un composant "double clickable". Les paramètres de lancement sont contenus dans le fichier Client_MacOSX.app/Contents/Info.plist. Il suffit de paramétrer l'entrée *<string>-applicationURL http://nom_du_serveur/cgi-bin/WebObjects/App.woa</string>* avec la bonne URL (on peut aussi utiliser server:port).

Problèmes:

- les firewalls empêchant l'accès au port de l'application sur le serveur ou filtrant les échanges avec le serveur Apache.
  (Norton Internet Security peut bloquer certains échanges binaires des applications WebObjects)
- l'indisponibilité de ClientSTD.jar.
- mauvaise version de java.

Pour trouver le problème, il faut regarder l'exception java en lançant l'application depuis une console avec java et non javaw.

Java Web Start:

Cette méthode simplifie le déploiement étant donné qu'il ne faut sur l'ordinateur que java.
En effet il suffit d'aller sur une page web où il y a un lien vers un .jnlp (dans le navigateur le type mime x-java-jnlp-file et les fichier avec extension .jnlp sont associés à javaws lorsque java est bien installé). Le fichier est téléchargé, passé à javaws qui le lit et télécharge les ressources nécessaires (jar, images ...).
(pour plus d'informations sur cette technologie voir Java Web Start, son guide ou le JSR 56)

Problèmes:

- généralement liés au téléchargement des jar:
  Sous Windows on ne peut ouvrir un fichier dont le chemin (lui inclus) fait plus de 256 caractères, il faut donc changer le répertoire "dossier des applications" qui sert de cache à Java Web Start par un autre dont le chemin est plus court (plutôt choisir un répertoire près de la racine et hors du profile de l'utilisateur comme C:\temp, attention aux droits pour les sessions suivantes).
- Un jar (comme CLientSTD.jar) dans le CLASSPATH peut avoir une class qui fait concurrence à l'une de celles qui se trouvent dans les jar téléchargés.
- les firewalls empêchant l'accès au port de l'application sur le serveur ou filtrant les échanges avec le serveur Apache.

Si les jar sont bien téléchargés, qu'il n'y a pas de problèmes de communication avec le serveur et que l'application ne fonctionne pas il faut regarder l'exception dans la fenêtre de dialogue (voir dans préférences avancées de Java Web Start), elle fournira une information sur le problème.

Remarques:

- Java Web Start demande à créer des raccourcis, il ne faut pas le laisser faire car il le fait "mal": le raccourcis utilise un jnlp téléchargé et stocké en cache, ce qui n'est pas forcement adapté lorsque l'on a un profil itinérant... Il faut faire des raccourcis vers javaws.exe avec en argument l'URL du fichier .jnlp de l'application (l'url du jnlp peut aussi être bookmarkée dans le navigateur).
- Dans le cas où les profils des utilisateurs sont centralisé sur un serveur avec le répertoire de cache, ce dernier va prendre de la place pour rien: il va y avoir x fois les même jar. Donc dans la politique de déploiement de Java Web Start prévoir de purgé le cache, de mettre une taille maximal, de le mettre hors du profile....
- Les jar des applications sont signés il faut donc faire confiance aux certificats, de même pour leur téléchargement en https (avoir un certificat reconnu au niveau des jar est plus couteux que celui présenté au niveau https, les certificats SCS du CRU peuvent être utilisé pour signer des jar mais ne seront pas reconnus car ce sont des certificats serveur comme l'indique le sigle SCS)
- Pour signer les jar il faut avoir un keystore avec un certificat:
  - création du certificat:
    keytool -genkey -keystore my.ks -alias monuniv -validity 3650
    (à noter qu'il existe un outils graphique Keytool IUI)

    - signature d'un jar: *jarsigner -keystore monkeystore -storepass passdukeystore -keypass passdesclefs /..../htdocs/WebObjects/Java/monfichier.jar nomducertificat*
    - verification: *jarsigner -certs -verbose -verify ...jar*
- les applications pré-remplissent le login avec celui de la session sur l'OS, pour cela il faut dans le jnlp:
  <security>
  <all-permissions/>
  </security>
  sinon il peut y avoir une erreur. (le login doit être alpha-numérique)
- Toujours regarder les logs (ceux du serveur également), généralement les messages sont assez explicites pour comprendre ce qui ne va pas.
- Lors de problèmes sur une application WebObjects 5.x :
    - vérifier que toutes les ressources Web sont disponibles et à jour
    - vider le cache java sur le poste client ou relancer le navigateur web peut aider
    - vérifier que l'application tourne, s'il n'y pas création du fichier de log correspondant à l'instance il faut lancer l'application en ligne de commande (avec plus ou moins d'arguments)
    - modification du SpawnOfWotaskd.sh pour avoir les sorties sdtout et stderr.
      remplacer *$@ 1>/dev/null 2>&1 &* par *sh $@ 2>&1 1>> $NEXT_ROOT/Local/logs/webobjects.log &*
      tout n'est pas loggué (il doit y avoir le "generated classpath") car WebObjects prend en charge la création d'un fichiers spécifique pour instance (s'il n'est pas créé alors que dans webobjects.log il y a le classpath généré, il y a de grandes chances pour qu'il manque un jar)
    - coté serveur on peut démarrer une instance en activant le debug avec l'option: *-WODebuggingEnabled YES* (il y a une checkbox dans JavaMonitor)
      on peut spécifier: *-NSDebugLevel 3* (defaut est 1), *-NSDebugGroups -1* (-1 = tout les groupes, groupes importants 18:locks, 13:IO, 17:SQL), *-EOAdaptorDebugEnabled true* (correspond au DebugGroup 17 soit SQL generation)
      (voir la JavaDoc NSLog pour plus d'informations sur les niveaux de debug)
    - certaines applications ont dans leur fichier de configuration une option pour activer leur propre debug (basé sur NSLog ou pas)
- les applications légères java ne peuvent passer par un proxy sauf de manière transparente.

Conclusion

Le fonctionnement des applications sur le poste clients ne doit pas être souci, les problèmes liées au lancement des applications ont été listés et ne sont pas nombreux.
Java Web Start est recommandé car il simplifie le déploiement d'applications avec interface Java (WebObjects 5.x).
Ne pas oublier que la résolution de problèmes peut passer par l'obtention puis l'analyse de fichiers de logs.