Notice explicative d'application WebServiceAdminSystem

Dans le schéma ci-dessous (*figure 1*) nous trouvons l'architecture de l'application développée :

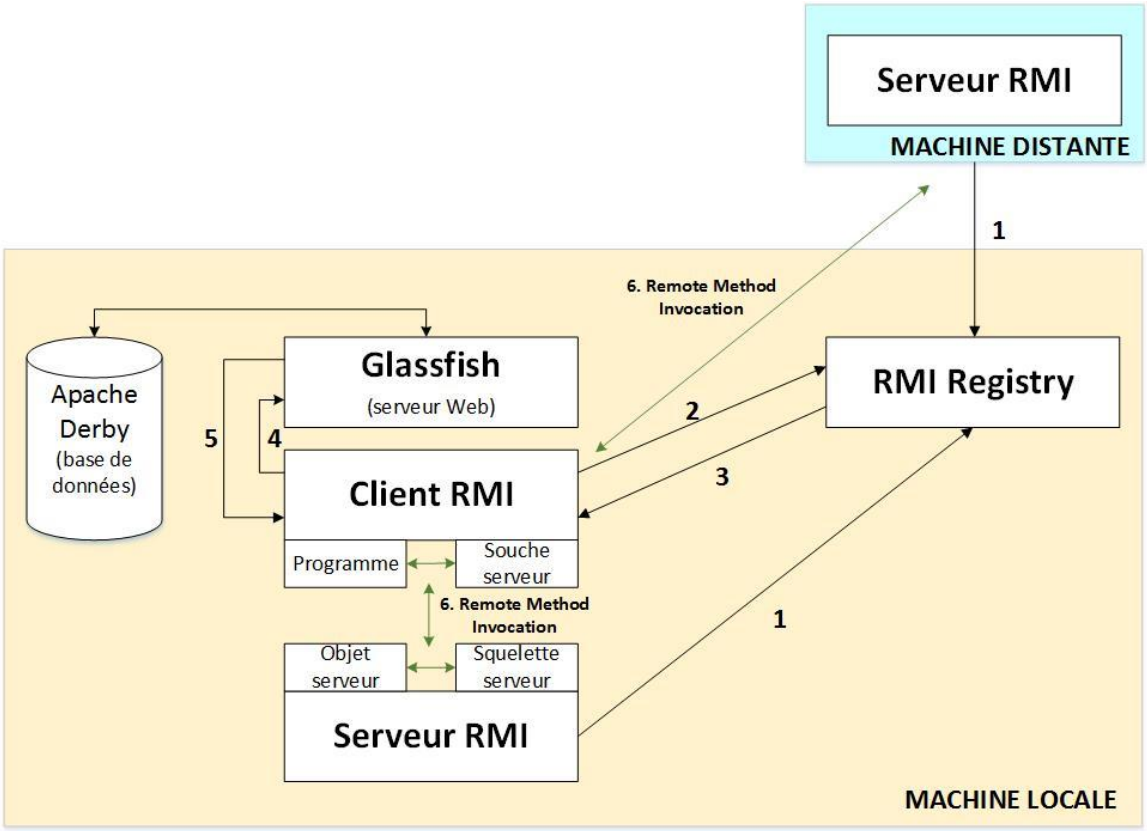

Figure 1 : Eléments du WebServiceAdminSystem

En termes de fonctionnement les évènements se passent dans la séquence suivante :

1. Le serveur RMI fait un enregistrement auprès du serveur *"RMI registry"* - `(re)bind "RMIServer"`
2. Le client recherche les informations concernant le serveur RMI - `lookup "RMIServer"`
3. Une instance de la "souche" d'objet distant est envoyée au client
4. Le client demande le serveur Web pour la classe "souche" d'objet distant
5. Le serveur Web renvoie la classe "souche" d'objet distant

Un servlet existe au sein d'un conteneur Web. Le conteneur permet de passer des méthodes "*get*" ou "*post*" au servlet qui ensuite les traites dans son code.

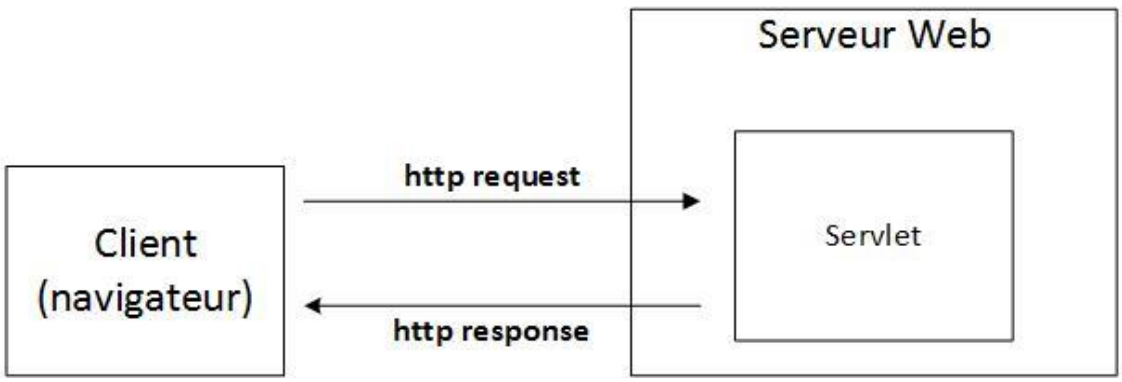

Figure 2 : HTTP entre client et serveur

Le serveur Web doit être capable de "mapper" la requête d'un utilisateur à une ressource existante. Pour cela, il va s'appuyer sur le fichier descriptif `web.xml` ce qui lui permettra de voir où se trouve la ressource demandée. Dans l'exemple ci-dessous (*figure 3*) l'utilisateur essaie de trouver la page `admin-welcome.jsp`. Le serveur voit que cette page (ressource) est "mappée" au servlet `ServletTreatLogin` qui se trouve dans le paquetage `admin`.

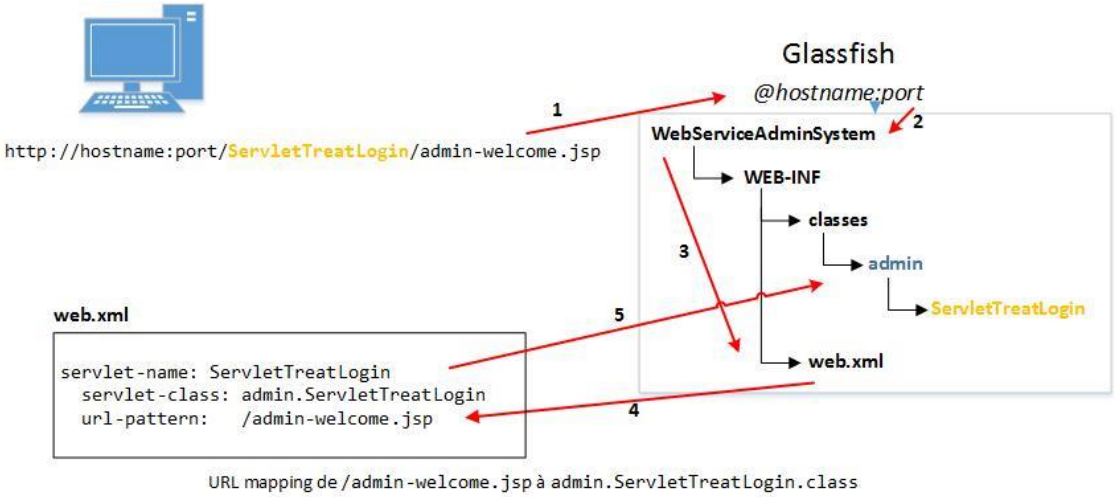

Figure 3 : Fonctionnement d'un servlet

Un outil de persistance de données a été mis en place grâce à la création d'une base de données spécifique à ce projet. Nous retrouvons le schéma de cette base de données ci-dessous :

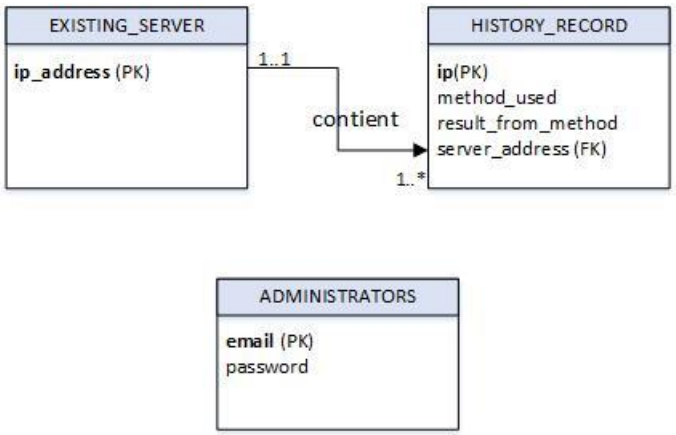

Figure 4 : Schéma BDD

Améliorations futures :

1. Vérifier si le serveur RMI est lancé (« up ») en essayant de faire un « ping »
2. Vérifier si le RMIregistry est lancé en essayant un « telnet » sur le port 1099
3. Dans la page « Historique » on peut afficher :
 - nom d'utilisateur qui avait lancé la commande
 - la date de lancement de la commande
4. On peut mettre en place la possibilité de lancer plusieurs commandes sur plusieurs serveurs en même en ajoutant une case à cocher
5. On peut mettre en place la possibilité de supprimer plusieurs processus se trouvant dans la page « Historique »

Solutions essayées :

- Possibilité d'un problème provenant depuis le chemin où le Server.jar se trouve (**pas fonctionnel**)
- Possibilité de manque de droits (**pas fonctionnel**) :

```
System.setProperty("java.security.policy", "./security.policy");
System.setSecurityManager(new RMISecurityManager());

./security.policy
   grant {     permission java.security.AllPermission "", ""; };
```

- Possibilité d'externaliser le Client et le serveur Web **(pas fonctionnel)** :
  Machine 1 contenant le Server.jar - on le lance depuis Netbeans
  Machine 2 contenant Glassfish et ClientRMI

- Mettre à jour la version de JDK **(pas fonctionnel)**