



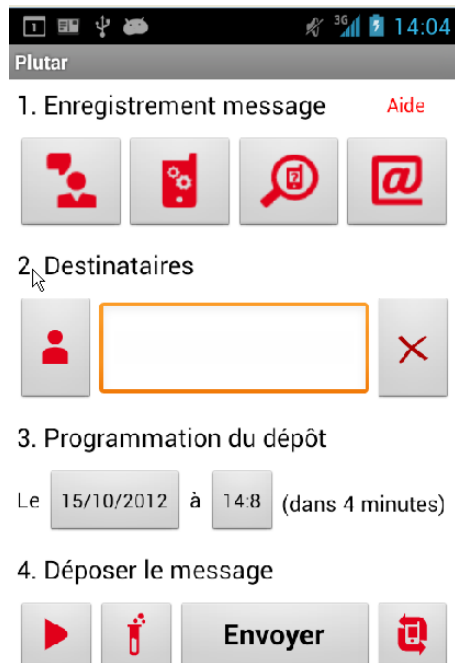
Démonstrateur de  
l'API SFR "dépôt de message"

L'application Plutar est une application Open Source Android et Java (serveur) faisant la démonstration de l'[API de dépôt direct de messages](#) sur boîte vocale SFR.

L'idée a été présentée une première fois en décembre 2010 lors d'un Hackathon Android organisé par BeMyApp. Une première version a été développée par une équipe réunissant :

- Dorian Herlory
- Aurdrey Martel
- Hervé Hoareau
- XXX

Cette version permettait notamment l'envoi différé de SMS, de mail ou de MMS. Ici on ne reprend que le dépôt de message sur boîte vocale. L'architecture de cette version reste la même mais le code a été entièrement réécrit.



Le principe consiste à enregistrer un message depuis son téléphone, lui attacher une liste de destinataires et programmer une date de dépôt. Le message est envoyé à un serveur qui déclenche une [demande de consentement](#) au près des destinataires (autorisation d'accès à sa boîte vocale pour déposer un message) puis conserve le message jusqu'à la date de dépôt.

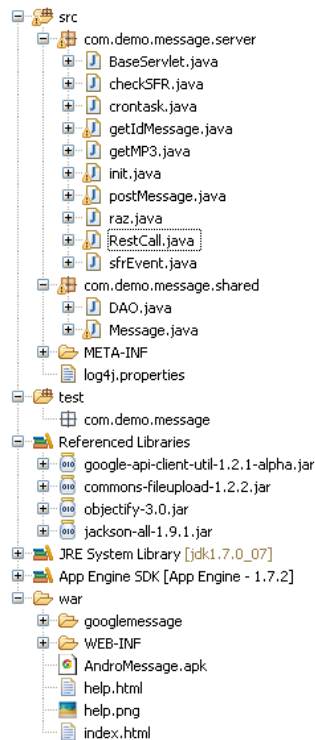
Du point de vue de l'architecture, cette application se décompose en :

- une interface client, sous forme d'une application native Android,
- un serveur java, hébergé par le [Google App Engine](#), utilisant les API SFR.

## Le code de l'application

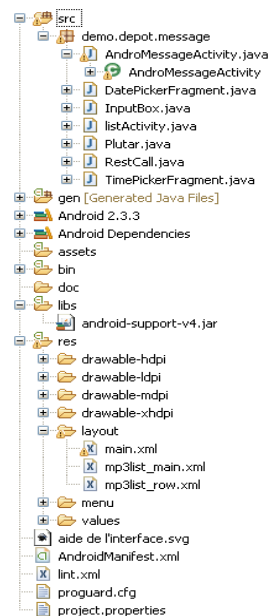
Le serveur et le client ont été développés sous Eclipse avec les environnements Google App Engine et Android installés. Le code est disponible sous [licence GNU](#), récupérable et modifiable sur le [dépôt GitHub](#) de SFR API. Il illustre plusieurs concepts fréquents dans le développement d'applications ayant recours à des web services externes :

Au niveau du serveur Java :



- l'usage de l'API SFR de dépôt de message (*class* *postMessage*)
- la gestion des consentements de dépôt (*class* *sfrEvent*) et l'[attribution de numéros courts SFR](#)
- l'usage des API SFR en mode REST depuis un environnement java (*class* *RestCall*)
- l'exécution de tâches programmées sur le Google App Engine (GAE) (*classe* *crontask* et [QueueFactory](#) dans la *classe* *PostMessage*)
- l'usage de la base de données du GAE au travers du framework [Objectify](#) (*class* [DAO](#))
- la manipulation de données au format JSON (librairie [jackson](#))

Au niveau du client Android :



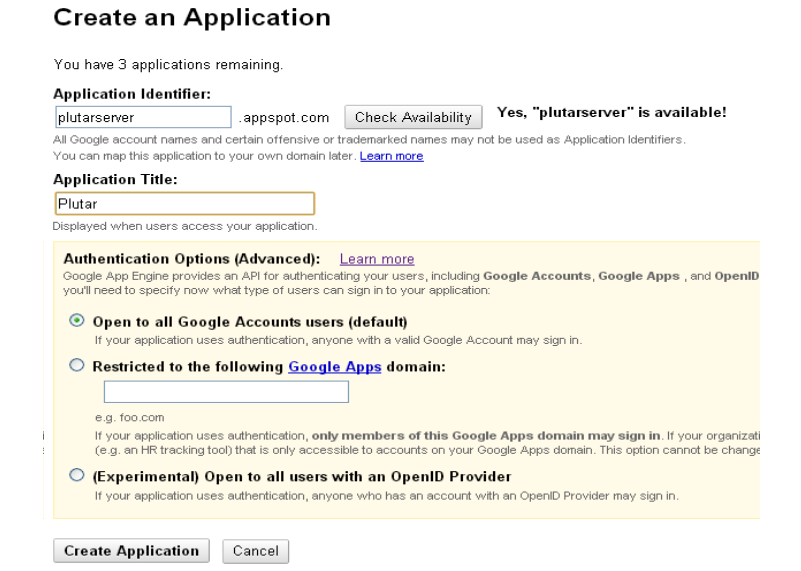
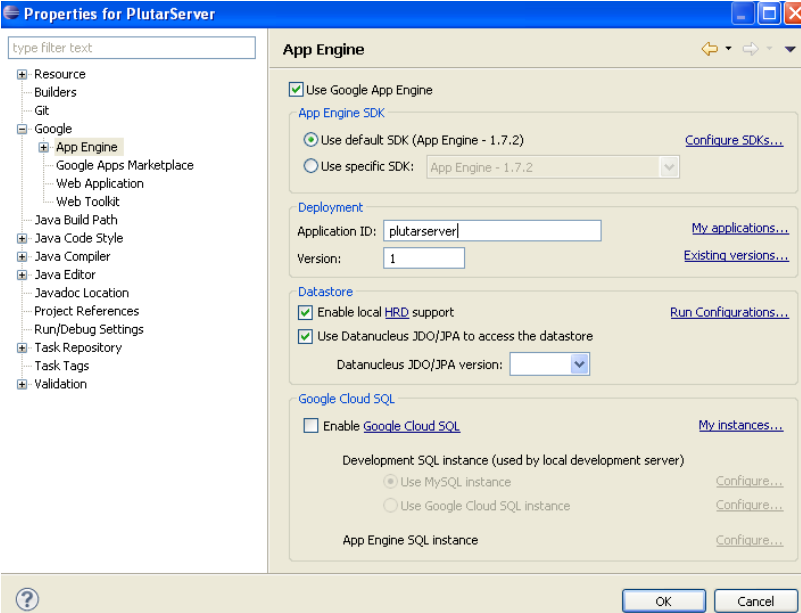
- l'appel d'API en mode REST (*class* *RestCall*)
- l'upload de fichier binaire vers un serveur GAE avec encodage préalable en base64
- l'usage du moteur de [synthèse vocale Android](#)
- l'usage du micro pour créer un fichier son
- la saisie de date & heure via le SDK Android (*class* *DatePickerFragment* et *TimePickerFragment*)

## Récupération du code

Ce code est avant tout un exemple de ce qu'il est possible de faire avec l'API de dépôt de message. L'application n'est pas à un niveau de robustesse permettant d'envisager directement

une exploitation commerciale. Pour autant le code est réutilisable et modifiable selon les termes de la licence GNU.

L'installation du serveur et du client se fait en quelques étapes :

<p>Récupération du code serveur sur GitHub :</p> <p><a href="https://github.com/sfrapi/Plutar-serveur">https://github.com/sfrapi/Plutar-serveur</a></p>	
<p>Créer une application sur le Google App Engine.</p> <p>voir le tutorial : “<a href="#">Ma première application sous GAE</a>”</p>	
<p>Reporter l'identifiant GAE dans les propriétés du serveur.</p>	

Paramétrer l'adresse du serveur dans la constante `SERVER_MESSAGE` de la classe `BaseServlet`

```
//Cette classe permet de partager des méthodes entre plusieurs classes filles
public class BaseServlet extends HttpServlet {
    private static final long serialVersionUID = 1464654657498786L;
    protected static DAO dao=new DAO();
    private static final String SERVER_MESSAGE="http://plutarserver.appspot.com/api";
```

S'inscrire sur SFR API pour récupérer un token permettant d'utiliser les API SFR :

<https://api.sfr.fr/user/register>

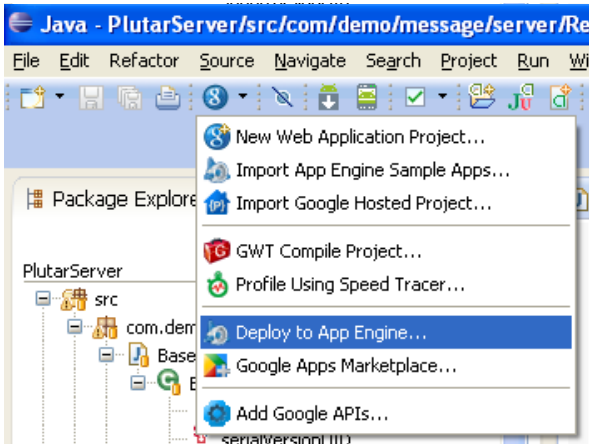
Inscrire le nom du projet comme alias du service utilisé par SFR API :

<https://api.sfr.fr/my-services>

Insertion du token SFR API dans le code du serveur :  
Variable `tokenSFRAPI`

```
public class RestCall {

    //Inscrire dans la variable tokenSFRAPI, le token de votre service récupérer
    //sur http://api.sfr.fr/
    private String tokenSFRAPI=URLEncoder.encode("<ici votre token SFR API>");
    private String ident="token="+tokenSFRAPI+"&reponseType=json";
```

dans la classe RestCall	
Publier le serveur sur le Google App Engine	

Une fois le serveur en ligne, celui-ci peut être utilisé par le client Web (fichier index.html) ou le client Android.

De la même façon le client Android doit être paramétré avant d'être exécuté sur le mobile.

<p>Récupération du code client sur GitHub :</p> <p><a href="https://github.com/sfrapi/Plutar-client-Android">https://github.com/sfrapi/Plutar-client-Android</a></p>	<div><div><div><div><div><div></div><div>Latest commit to the v0.1 branch</div></div></div><div><div><div>Version 0.1 test</div><div><div><div><div></div><div>hhoareau</div><div>authored 2 days ago</div></div><div><div></div><div>commit db-948f13</div></div></div></div></div></div></div><div><div>Plutar-client-Android /</div><table><thead><tr><th>name</th><th>age</th><th>message</th><th>history</th></tr></thead><tbody><tr><td><div><div></div>PlutarClient</div></td><td>2 days ago</td><td>Version 0.1 test [hhoareau]</td><td></td></tr></tbody></table></div></div></div>	name	age	message	history	<div><div></div>PlutarClient</div>	2 days ago	Version 0.1 test [hhoareau]	
name	age	message	history						
<div><div></div>PlutarClient</div>	2 days ago	Version 0.1 test [hhoareau]							
<p>Paramétrer l'adresse du serveur dans la variable MESSAGE_SERVER de la classe <i>PlutarActivity</i></p>	<pre>public class PlutarActivity extends FragmentActivity {      //La constante MESSAGE_SERVER doit contenir l'adresse du serveur     //par exemple, pour google app engine : http://nom_appli_GAE.appspot.com     private static final String MESSAGE_SERVER = "http://&lt;nom_appli&gt;.appspot.com"; }</pre>								

## Remarques

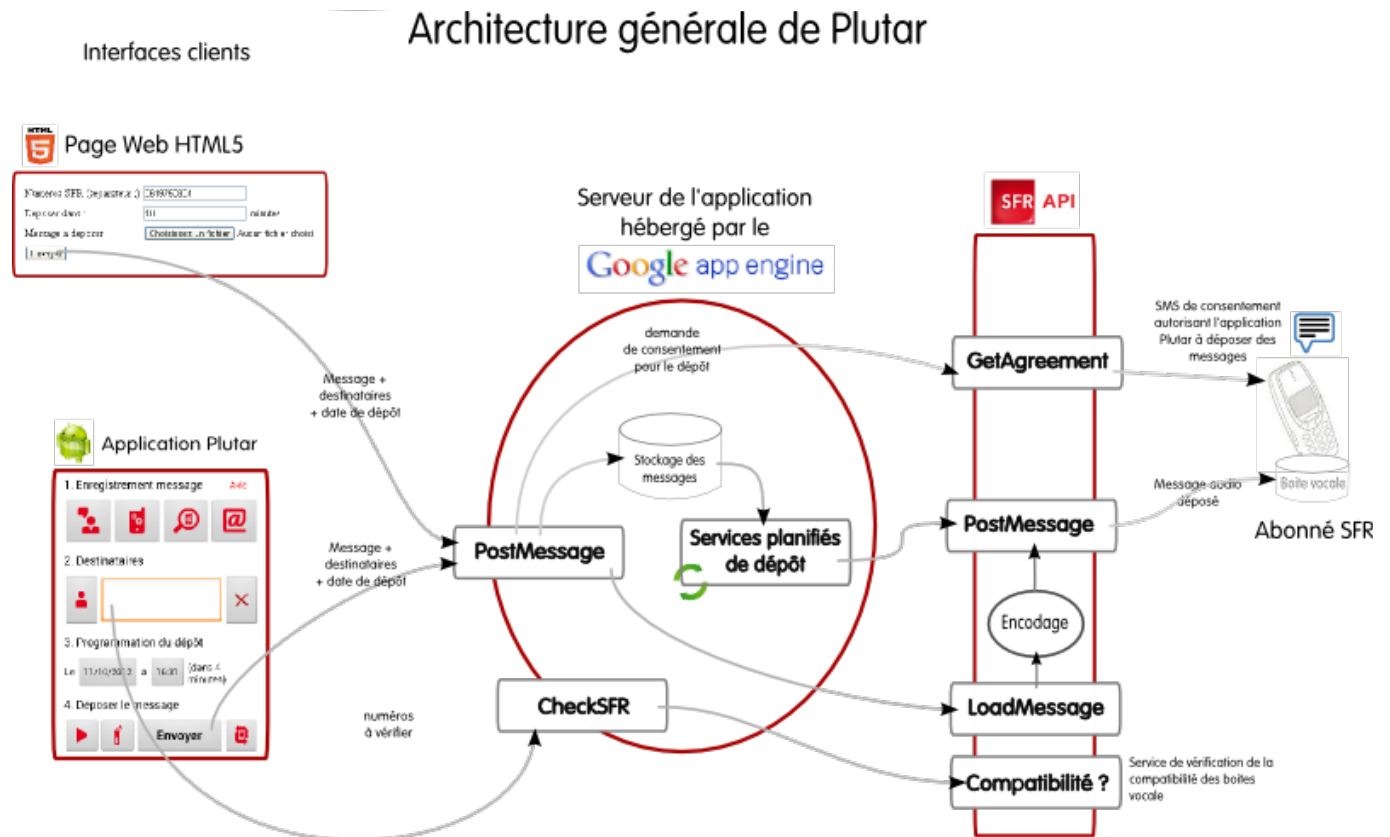
### Gratuité

SFR API, comme Google App Engine fonctionnent gratuitement en dessous d'un certain volume d'usage :

- [Tarifs du Google App Engine](#)
- [Tarifs SFR API](#)

Cela signifie que le service est entièrement utilisable gratuitement (après ouverture d'un compte Google App Engine et SFR API).

## Architecture générale de la solution



## Evolutions

De nombreuses évolutions pourraient être mise en oeuvre sur cette application,

### Évolutions techniques

- sécuriser l'accès aux web services du serveur,
- optimiser l'envoi du fichier
- améliorer la gestion des exceptions au niveau serveur et client

### Évolutions fonctionnelles des fonctions existantes ou ajout de nouvelles fonctionnalités

- Amélioration des fichiers sons (réduction du bruit par exemple, échos, reverb)
- Mixage de plusieurs fichier (pour par exemple pouvoir déposer un message avec fond sonore)

- présentation des noms des destinataires à la place des numéros de mobiles
- réintégration des messages différés sous d'autres formes : texte, sms, mail, mms (idem première version de Plutar)

Hervé Hoareau  
Responsable Développement SFR API