Ici vous trouverez des informations sur le code du plugin 4vet.

Vous pouvez trouver ici un exemple d'un plugin installé, de plus dans cette page on peut trouver la documentation du plugin SDK qu'on utilisera pour 4vet : http://www.codeproject.com/Articles/797118/Implementing-a-WADO-Server-using-Orthanc

> The Orthanc Plugin SDK is defined in a C header that is documented with Doxygen. This article will exploit this SDK to create the basic WADO server.

C'est ainsi qu'on trouve ceci :

> This C/C++ SDK allows external developers to create plugins that can be loaded into Orthanc to extend its functionality. Each Orthanc plugin must expose 4 public functions with the following signatures:
>
> 1. `int32_t OrthancPluginInitialize(const OrthancPluginContext* context)`: This function is invoked by Orthanc when it loads the plugin on startup. The plugin must store the context pointer so that it can use the plugin services of Orthanc. It must also register all its callbacks using OrthancPluginRegisterRestCallback().
> 2. `void OrthancPluginFinalize()`: This function is invoked by Orthanc during its shutdown. The plugin must free all its memory.
> 3. `const char* OrthancPluginGetName()`: The plugin must return a short string to identify itself.
> 4. `const char* OrthancPluginGetVersion()` The plugin must return a string containing its version number.
>
> The name and the version of a plugin is only used to prevent it from being loaded twice.

Cela nous montre 4 fonctions à utiliser dans notre plugin si l'on veut que cela fonctionne. J'ai alors trouvé que ces fonctions se trouvent dans le fichier : Plugin.cpp et débutent à la ligne 270.

Peut-on récupérer les tags spécifiques de cette façon (Cpp) ? :

```
(GetTag(center, 0x0028, 0x1050 /*DICOM_TAG_WINDOW_CENTER*/) &&
 GetTag(width, 0x0028, 0x1051 /*DICOM_TAG_WINDOW_WIDTH*/))
```

Image proventant de ParsedDicomImage.cpp (ligne 458)

Ensuite ce que je sais c'est que le dossier "Orthanc" provenant du dossier WebViewer, est un extrait du code source d'Orthanc afin d'obtenir une base de données ainsi que des fichiers importants pour le fonctionnement du plugin comme OrthancCPlugin.cpp. C'est pour cela que je pense que Sébastien n'a rien modifié à l'intérieur est à seulement ajouté des fonctionnalités à son plugin.
C'est ainsi qu'en faisant une recherche, avec le mot de recherche sqlite, j'ai trouvé qu'il n'y existait que 3 fichiers :

- CacheManager.cpp
- CacheManager.h
- Plugin.cpp

Cependant le fichier CacheManager.cpp me parait être celui qui fait le plus de requêtes SQL, cependant dans le fichier Plugin.cpp, on y exécute une connection à la base de données :

```
private:
  Orthanc::FilesystemStorage  storage_;
  Orthanc::SQLite::Connection  db_;

  std::auto_ptr<OrthancPlugins::CacheManager>  cache_;
  std::auto_ptr<OrthancPlugins::CacheScheduler>  scheduler_;
```

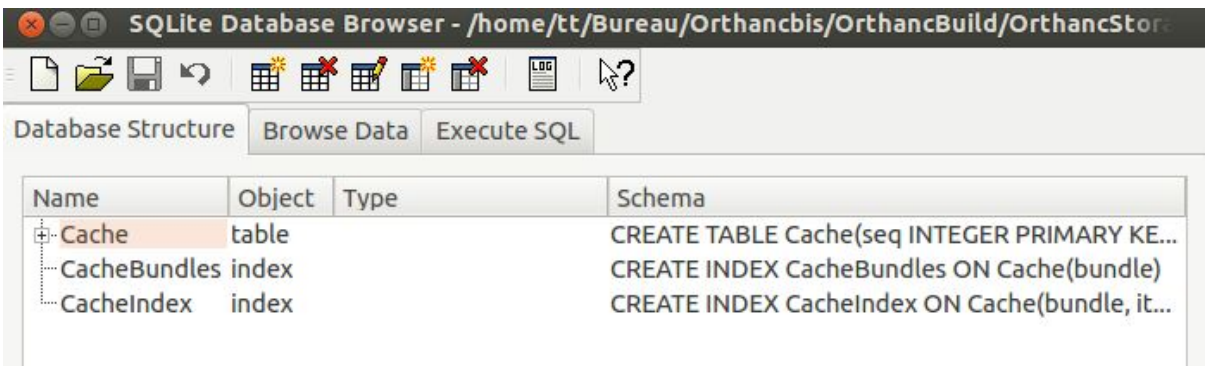
Je pense qu'il se connecte au cache.

Les données sont stockées dans la base d'Orthanc, "OrthancStorage" qui se trouve dans le dossier build d'Orthanc lors de sa compilation.

```
314       // By default, the cache of the Web viewer is located inside the
315       // "StorageDirectory" of Orthanc
316       boost::filesystem::path cachePath = GetStringValue(configuration, "StorageDirectory", ".");
317       cachePath /= "WebViewerCache";
318       int cacheSize = 100;  // By default, a cache of 100 MB is used
319
```

Pour accéder à ce fichier avec linux, vous pouvez télécharger SQliteDatabaseBrowser qui est simple à utiliser. Une fois installé, il vous suffit de sélectionner le fichier .db qui contient les données du cache de WebViewer : ~/OrthancBuild/OrthancStorage/WebViewerCache/

Vous devriez obtenir ceci :



C'est dans le fichier CacheManager.cpp que sont créées ces tables :

```
296  void CacheManager::Open()
297  {
298    if (!pimpl_->db_.DoesTableExist("Cache"))
299    {
300      pimpl_->db_.Execute("CREATE TABLE Cache(seq INTEGER PRIMARY KEY, bundle INTEGER, item TEXT, fileUuid TEXT, fileSize INT);");
301      pimpl_->db_.Execute("CREATE INDEX CacheBundles ON Cache(bundle);");
302      pimpl_->db_.Execute("CREATE INDEX CacheIndex ON Cache(bundle, item);");
303    }
```