EPFL CS212 : ImgStore – Système de fichiers orienté images — description générale

E. Bugnion & J.-C. Chappelier EPFL

Rev. 2021.01.07 / 1

Table des matières

Projet de Programmation Système – CS212 – 2021 **1**

Description générale du projet principal : **1**

ImgStore – Système de fichier orienté images **1**
Plan de ce document 1
Contexte 1
Buts 2
Organisation 3
Description générale 3

Projet de Programmation Système – CS212 – 2021

Description générale du projet principal :

ImgStore – Système de fichier orienté images

Plan de ce document

1. Contexte du projet
2. Buts du projet
3. Description générale

Contexte

Le but premier de ce projet est de vous faire développer un large programme en C sur une thématique « système ». Le cadre choisi cette année est la construction d'un utilitaire en ligne de commande pour gérer des images dans une base de données de format spécifique, inspiré de celui utilisé par Facebook. Pour votre infor-

mation, le système de Facebook s'appelle « Haystack » et est décrit dans le papier suivant : https://www.usenix.org/event/osdi10/tech/full_papers/Beaver.pdf). Vous **ne** devez **pas** lire ce papier dans le cadre du cours (c'est juste pour information) car, bien évidemment, nous allons implémenter une version simplifiée de ce système. Tous les concepts de base requis pour ce projet sont introduits ici de façon simple en ne supposant qu'une connaissance « utilisateur » standard d'un système informatique.

Les réseaux sociaux doivent gérer des centaines de millions d'images. Les systèmes de fichiers usuels (tel que celui utilisé sur votre disque dur par exemple) ont des problèmes d'efficacité avec de tels nombres de fichiers. De plus, ils ne sont pas conçus pour gérer le fait que l'on veut avoir chacune de ces images en plusieurs résolutions, par exemple toute petite (icône), moyenne pour un « *preview* » rapide et en taille normale (résolution d'origine).

Dans l'approche « Haystack », plusieurs images se trouvent dans un même fichier. De plus, différentes résolutions de la même image sont stockées automatiquement. Ce fichier unique contient à la fois les données (les images) et les métadonnées (les informations concernant chaque image). L'idée clé est que le serveur d'images a une copie de ces métadonnées en mémoire, afin de permettre un accès très rapide à une photo spécifique, et ce dans la bonne résolution.

Cette approche a un certain nombre d'avantages : tout d'abord, elle réduit le nombre de fichiers gérés par le système d'exploitation ; d'autre part, elle permet d'implémenter de manière élégante deux aspects importants de la gestion d'une base d'images :

1. la gestion automatique des différentes résolutions d'image, dans notre cas les trois résolutions supportées ;
2. la possibilité de ne pas dupliquer des images identiques soumises sous des noms différents (p.ex. par des utilisateurs différents chez Facebook) ; c'est une optimisation extrêmement utile dans tout réseau social.

Cette « déduplification » (« *deduplication* » en anglais) se fait à l'aide d'une « fonction hash » qui résume un contenu binaire (dans notre cas une image) en une signature beaucoup plus courte. Nous utiliserons ici la fonction « SHA-256 » qui résume tout contenu binaire en 256 bits, avec comme propriété cryptographique intéressante que la fonction est résistante aux collisions : pour une image donnée, il est pratiquement impossible de créer une autre image qui aurait la même signature. Dans le cadre de notre projet, nous considérons que deux images ayant la même signature SHA-256 sont nécessairement identiques. Même si cela peut paraître surprenant, de nombreux systèmes informatiques de production sont basés sur ce principe.

Buts

Vous allez construire un serveur d'images ad-hoc, dans une version inspirée et simplifiée de Haystack. Durant les premières semaines, il s'agira d'abord

d'implémenter les fonctions de base du système, à savoir :

— lister les informations (métadonnées, liste des images) ;
— rajouter une nouvelle image ;
— effacer une image ;
— extraire une image donnée, dans une résolution spécifique au choix : « *original* », « *small* » ou « *thumbnail* ».

Dans cette première phase, les fonctions seront exposées via un utilitaire de ligne de commande. Durant les dernières semaines du semestre, vous construirez un véritable serveur web qui exposera la même fonctionnalité avec la plus grande performance possible via le protocole HTTP.

Organisation

Durant les 10 semaines de ce projet, vous allez devoir implémenter, graduellement morceau par morceau les composants clés mentionnés ci-dessus et décrits plus bas, puis détaillés dans les sujets hebdomadaires.

Vous allez aussi devoir développer des tests complémentaires utiles pour observer et analyser le fonctionnement du système. Ces tests seront développés sous forme d'exécutables indépendants du cœur principal.

Afin de faciliter au mieux l'organisation de votre travail (dans le groupe et dans le temps), nous vous conseillons de consulter la page de barème du cours (et la lire en entier !!).

Description générale

Nous décrivons ici de façon générale les principaux concepts et structures de données que ce projet nécessitera. Leurs détails d'implémentation seront précisés plus tard lorsque nécessaire dans chaque sujet hebdomadaire correspondant.

Vous allez donc utiliser un format spécifique – appelons le « `imgStore` » (*Image Storage*) – pour représenter un « système de fichiers d'image » (ou « base de données d'images »). Un fichier de type `imgStore` (représentant donc un tel système) comprend trois parties distinctes :

— un `header`, de taille fixe, qui rassemble les éléments de configurations du système ; le contenu du `header` est créé lors de la création du `imgStore` ;
— le tableau des métadonnées ; il s'agit d'un tableau dont la taille est spécifiée par le champ `max_files` du `header` ; chaque entrée du tableau décrit toutes les métadonnées d'une seule image, et en particulier la position de l'image (sous ses différentes résolutions) dans le fichier ;
— les images elles-mêmes ; chaque image est stockée sur une partie contiguë du fichier, les unes après les autres.

Ce format sera utilisé par 2 outils que vous aurez à développer :

1. un utilitaire sur la ligne de commande (c.-à-d. un interpréteur de commandes, simple), permettant de manipuler de tels « système de fichiers d'image » : créer et lister le contenu de tels systèmes, ajouter, lire et supprimer des images qui s'y trouvent ;
2. un server Web permettant les mêmes commandes mais au travers d'un navigateur Web.