Compte Rendu

Sonny Klotz - Jean-Didier Pailleux - Malek Zemni

Interface de chargement, de contrôle
et d'analyse statistique des données
pour la constitution d'un graphe de flux

27/05/2017

UNIVERSITÉ DE VERSAILLES ST-QUENTIN-EN-YVELINES
université PARIS-SACLAY

Module *Projet*

Table des matières

Introduction

Ce document est le compte-rendu final de notre travail qui s'inscrit dans le cadre du module *Projet* de la licence informatique de l' ***UVSQ***. Le sujet de ce projet a été proposé par l'entreprise ***DCbrain***.

Notre client, ***DCbrain***, développe des outils qui permettent de visualiser le comportement des **réseaux physiques** afin de prédire et trouver les problèmes de ces réseaux dans le but de les optimiser. Ces réseaux physiques sont principalement les réseaux industriels, de fluide ou de distribution, tels les réseaux électriques. Des données sont collectées à partir de ces réseaux puis analysées grâce aux technologies du **Big Data**. Le but est d'en extraire une valeur ajoutée.

Notre tâche correspond au travail indispensable d'analyse préliminaire de ces données, afin de faciliter leur exploitation. Ce projet a donc pour objectif de fournir aux utilisateurs une application web qui permettra de charger les données collectées, les visualiser et enfin les analyser.

Dans une première partie de ce document, on présentera l'architecture de notre application, illustrée par un organigramme qui a été préalablement établi. Cet organigramme représente un découpage de l'application en modules avec les différentes informations qui circulent entre ces modules.
Dans une deuxième partie, on parlera des différents langages de programmation choisis pour réaliser l'application et des contraintes qui ont justifié ces choix.
Ensuite, dans une troisième partie, on traitera la partie technique de notre projet, c'est à dire, l'application, son fonctionnement et ses problèmes.
Finalement, dans les deux dernières parties, on établira un bilan quant à l'organisation interne au sein du groupe et un bilan comparatif des coûts présumés et des coûts finaux de notre produit.

1 Architecture de l'application

1.1 Organigramme et données échangées

Cet organigramme a été préalablement établi dans le cahier des charges du produit. Il représente la décomposition en modules de l'application, ainsi que les informations qui circulent entre ces modules.
Remarque : on pourra noter quelques légers arrangements dans la représentation des informations transmises entre les modules de l'organigramme. Ces arrangements sont principalement effectués pour des fins d'optimisation et seront explicitement justifiés.

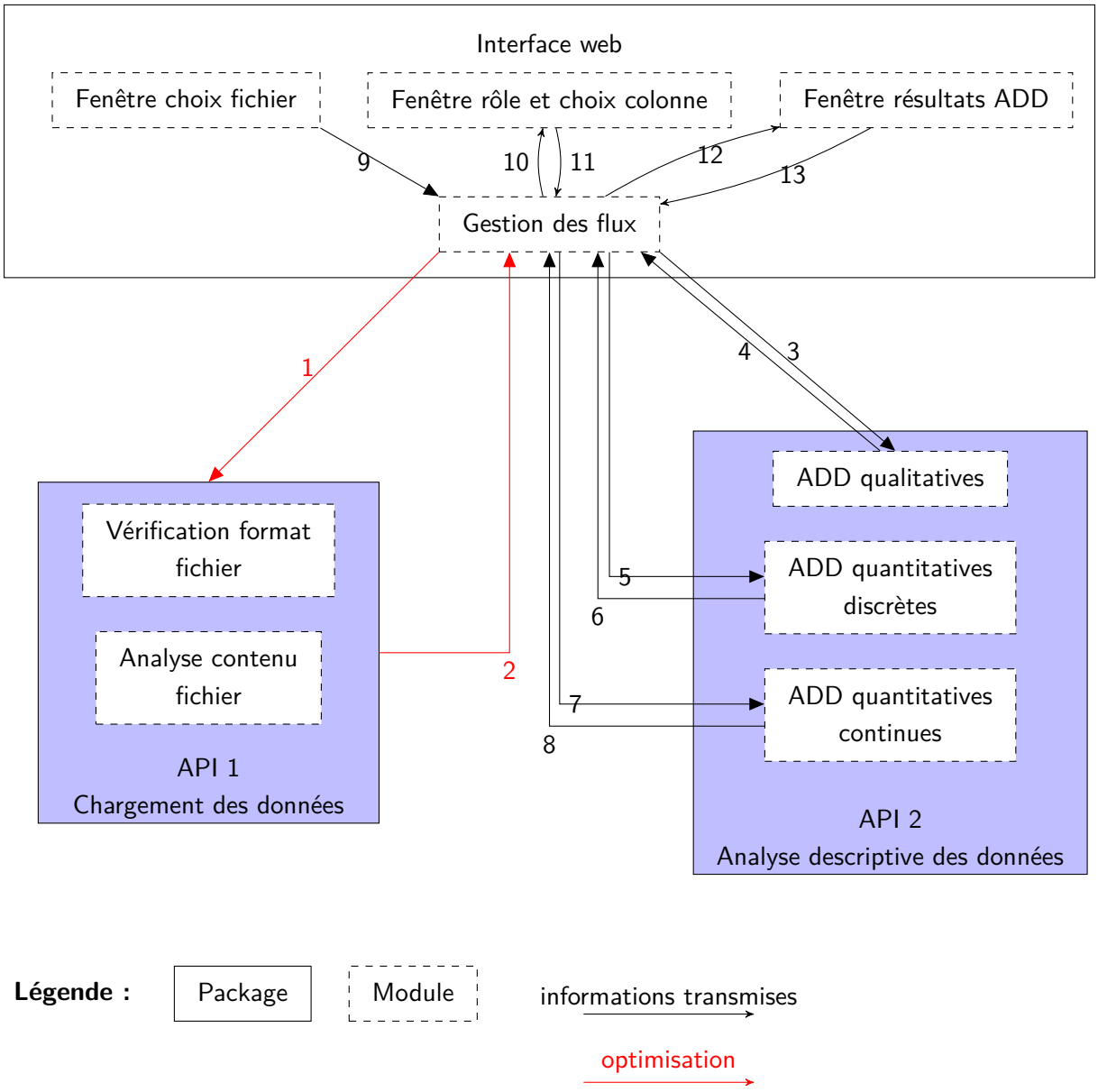

FIGURE 1 – Organigramme des différents modules du logiciel

Notes :

(1) Chemin du fichier CSV importé : lancement des fonctionnalités du module de chargement de manière indépendante : la vérification de son format et l'analyse de son contenu

(2) Le résultat du chargement du fichier CSV importé :

- soit un message d'erreur signalant un problème lié au fichier
- soit deux structures, l'une contenant les données du fichier, et l'autre contenant une description du nom, du type, et des erreurs de chaque colonne de données du fichier

Justification de l'optimisation : lors de la phase d'enchaînement entre les différents modules de l'ap-

plication, on a remarqué qu'il était plus judicieux d'établir une communication à une seule passe avec le package de chargement des données, au lieu de faire le va et vient sur ses différents modules. De cette manière, ce package devient indépendant et assure ainsi une facilité lors du débogage et de la maintenance.
(3) Ensemble de données de type qualitatif
(4) Erreur ou effectifs, effectifs cumulés, fréquences, fréquences cumulées, diagramme en secteur, histogramme
(5) Ensemble de données de type quantitatif discret
(6) Erreur ou indicateurs de tendance central, de dispersion et de forme, les anomalies, la distribution des données, un diagramme à moustaches
(7) Ensemble de données de type quantitatif continu
(8) Même données que (6)
(9) Chemin du fichier CSV importé
(10) Informations du (2) et ensemble de données contenu dans le fichier CSV
(11) Signal de validation du choix de la colonne, et noms des colonnes
(12) Envoi des résultats d'analyses de (4), (6) et (8)
(13) Signal de contrôle : demande d'exportation des résultats de l'ADD, analyse d'une autre colonne, ou importation d'un autre fichier

1.2 Format du fichier CSV

Le format du fichier a été établi par le client ***DCbrain***. Son contenu est assez structuré et est décrit par des colonnes aux types prédéfinies :

timestamp	**parent**	**enfant**	**mesure 1**	**mesure 2**
January 1st 2017, 15 :00 :00.000	102	95	26644.235	176.253

Remarque : le graphe de flux utilisé par ***DCbrain*** pour analyser les réseaux de ses clients est orienté, d'où l'utilisation des nœuds *parent* - *enfant* numérotés. Ceux-ci permettent d'identifier une connexion précise. Les colonnes *mesure* représentent des données mesurées sur une connexion à un temps donné. Le nombre de ces colonne n'est pas limité et en pratique il ne dépassera pas la dizaine.

1.3 Fonctionnalités des modules

Package Chargement des données

1. Module Vérification format fichier :
   Vérification de l'existence du fichier, de l'extension, de la lisibilité du contenu (texte brut ou formaté) et l'accessibilité en lecture.

2. Module Analyse contenu fichier :
   Ce module comprend deux fonctionnalités principales :
   — Lecture du contenu du fichier CSV : on détermine le délimiteur de données du fichier, puis on lit ses données ligne par ligne. Ces données sont stockées dans une (**première structure**.
   — Description des données de chaque colonne du fichier CSV : on stocke les noms des colonnes fournis dans le fichier, ensuite on détermine les types de données attendus à partir de ces noms, et finalement, on parcourt la structure contenant les données du fichier en comparant le type de ces données au type attendu. Si ces types ne correspondent pas, on indique une description de l'erreur.
   Ces trois informations (nom, type et erreurs) sont stockées dans une (**deuxième structure**.
   Ce module va donc fournir les deux structures décrites ci-dessus.

Package Analyse descriptive de données

1. Module ADD qualitatives :
   — Calcul des effectifs, effectifs cumulés, fréquences et fréquences cumulées des données
   — Préparation des représentations graphiques : diagramme en secteur et histogramme
2. Module ADD quantitatives discrètes :
   — Statistiques : moyenne, quantiles, variance, écart-type, coefficients de symétrie et d'aplatissement de Fisher
   — Anomalies de Tukey : toute donnée hors de l'intervalle [Q1 - 1.5*IQ ; Q3 + 1.5*IQ], où Q1 et Q3 sont les quartiles, et IQ l'écart inter-quartiles.
   — Préparation des représentations graphiques : fonction de distribution, fonction de distribution cumulative, boîte à moustaches
3. Module ADD quantitatives continues :
   — Discrétisation de l'étendue (découpage en classe d'intervalles)
   — Calcul des quantiles pour le cas de données continues
   — Préparation des représentations graphiques : fonction de distribution cumulative

Package Interface web

1. Module Gestion des flux :
   — Gestion des branchements : exécution normale ou arrêts pour cause d'erreur.
   — Interface entres les différentes fonctionnalités : communique les données nécessaires entre les modules.
2. Module Fenêtre choix fichier :
   — Récupération un fichier CSV en renseignant son chemin en parcourant l'arborescence de fichiers, ou de la manière d'un Drag & Drop.

3. Module Fenêtre rôle et choix colonne :
   — Affichage des données du fichier CSV : noms de colonnes, leurs valeurs et du nombres de lignes du fichier.
   — Affichage des données erronés (description de l'erreur).
   — Affichage d'un échantillon de données du fichier à l'aide de filtres.
   — Sélection et envoi d'une colonne de mesures pour lancer l'analyse sur celle-ci.
4. Module Fenêtre résultats ADD :
   — Affichage des résultats d'analyse descriptive : informations statistiques de l'API 2 et représentations graphiques pour visualiser les données dans leur ensemble.
   — Une fonctionnalité de retour en arrière permet de sélectionner une nouvelle colonne sans relancer l'applet.

2 Outils et langages de programmation

2.1 Contraintes

Lors de la réalisation de l'application, des contraintes ont du être respectées. Ces contraintes ont été imposées par notre client ***DCbrain*** lors de la remise du sujet du projet. On peut en énumérer trois, qui sont les suivantes :

1. Le produit doit fournir une application web : notre produit sera une application fonctionnant sur un navigateur web, appelée ***applet***.
2. Le produit doit être développé avec un langage de programmation compatible avec l'analyse de données : le langage de programmation choisi doit inclure des bibliothèques qui permettent d'analyser les données (taches de data mining et de machine Learning).
3. Le produit doit fournir une API (ensemble de packages) d'analyse de données : le code source de l'application doit intégrer des API d'analyse descriptive de données qui pourront être réutilisés par le client.

Ces contraintes se sont avérées déterminantes pour les choix techniques que l'on a fait, et en l'occurrence, le choix des langages de programmation.

2.2 Choix des outils et des langages

Les contraintes imposées par le client ainsi que les exigences définies nous ont permis de fixer nos choix sur plusieurs langages de programmation qui vont interagir ensemble :

— **Python :** d'une part, ce langage est adapté pour l'analyse descriptives de données, puisqu'il est doté de nombreux modules permettant d'effectuer du calcul scientifique sur des données, donc l'analyse de données. D'autre part, ***Python*** est compatible avec le développement d'applications web grâce aux bibliothèques web qui permettent cela, dont ***Flask*** qu'on a utilisé. En effet,

Flask est une bibliothèque qui s'efforce d'être la plus simple possible pour une prise en main, à l'opposé d'autres bibliothèques. C'est donc plus abordable pour une première expérience sur le développement d'applications web.

— **Langages de programmation web :** l'application devant être une applet intégrée dans une page web pour s'exécuter, le recours à des langages de balisage comme ***HTML*** pour la présentation et ***CSS*** pour la mise en forme de ces pages web était nécessaire.
Ensuite, pour permettre à l'application de faire communiquer les pages web avec un serveur web sans occasionner le rechargement de la page, nous avons eu recours à ***Ajax***. Il s'agit d'un concept de programmation web reposant sur plusieurs technologies comme le ***Javascript***, qui lui se charge d'établir la connexion entre la page web et serveur. ***Javascript*** est également employé pour rendre les pages web interactives. ***Jquery*** est quant à lui utilisé pour faire le lien entre ***HTML*** et ***Ajax***.

— **Autres outils :** pour la génération de la documentation des fonctionnalités du programme, on a utilisé le logiciel ***Sphinx*** qui se charge de générer une *docstring* à partir des commentaires présents dans le code python.

3 Bilan technique du projet

3.1 Comportement de l'application

Notre produit final, c'est à dire l'application, se comporte comme prévu : l'application est fonctionnelle, la liaison entre ses différents modules réussi bien et les différentes fonctionnalités fournissent le résultat attendu. Voici un descriptif du comportement typique de notre application après lancement :

— Importation du fichier à analyser par un parcourant d'arborescence ou par un Drag & Drop.

— Chargement des données du fichier et visualisation des valeurs, avec possibilité d'application de filtres.

— Résultats de l'analyse d'un colonne choisie du fichier avec des graphes et des informations statistiques.

Tests : les principales fonctionnalités de notre application ont été testées avec plusieurs cas de tests différents. Ces tests ont traité plusieurs cas d'utilisation de ces fonctionnalités et nous ont permis de trouver quelques erreurs et ainsi les corriger. La décomposition en différents modules de l'application nous a permis de tester ces modules indépendamment, sans qu'il n'y ait d'impact sur les autres modules lorsqu'une erreur se produit.

Verrous techniques (levés) : langages de prog nouveaux, points délicats de flask/application web sur l'approche et tout ce qui va avec

3.2 Problèmes rencontrés

Problèmes résolus :

gestion des routes avec flask, application de flitres sur les données et pagination, traitement de fichiers mal structurés, Objet TextIOWrapper non sérialisable,

Problèmes non résolus :

- perfs tab echantillonage (données trop grandes) - sonny graphs

Points divergent :

Justification des points qui divergent par rapport aux specs et organigramme de départ. - FileWithSGF/ FileWithDragDrop se charge d'upload et nom de retourner un chemin. - Pas d'écriture dans des json mais dans des js ? ?

4 Organisation interne du groupe

Ci-dessous se trouve le tableau de la répartition des modules pour chaque membre du groupe établie avant le commencement du développement :

Module	Malek	Sonny	Jean-Didier	Total
Gestion des flux			x	1
Fenêtre choix fenêtre			x	1
Fenêtre rôle et choix colonne	x			1
Fenêtre résultats ADD		x		1
ADD qualitatives			x	1
ADD quantitatives discrètes		x		1
ADD quantitative continues		x		1
Vérification format fichier	x			1
Analyse contenu fichier	x			1

1 page On expliquera plus tard comment ça s'est déroulé concrètement par rapport à ce qui était prévu.

5 Coût

Tableau indiquant l'estimation et le coût final en nombre de lignes de code, pour chaque module :

Module	Nombre de lignes	Justification	Coût final
Gestion des flux	15	Mise en forme du main et appel de l'application	
Fenêtre choix fichier	10 + 20	Fonctions pour : Drag& Drop + Système de fichiers	
Fenêtre rôle et choix colonne	5 + 20 + 10 + 10 + 20	Communication avec le module application + Affichage de l'interface+ lecteurs des valeurs + affichage des valeurs	
Fenêtre résultats ADD	10 + 3* 30	Envoie d'informations au module application + Construction des graphe pour l'ADD pour les 3 types d'analyse	
ADD qualitatives	20 + 20*3	Application des formules pour les calculs de fréquences et d'effectifs + calcul des valeurs pour la construction de 3 graphes	
ADD quantitatives discrètes	60 + 20*2	Application des formules attaché à l'analyse quantitative discret + calcul des valeurs pour la construction de 2 graphes	
ADD quantitatives continu	20 + 10 + 10 + 5 + 20*2	Parcours + choix précision classe d'intervalle + écriture + communication avec les modules+ calcul des valeurs pour la construction de 2 graphes	
Vérification format fichier	30	Ouverture fichier + vérification si ouverture en lecture + présence de texte formaté ou non	
Analyse contenu fichier	20 + 5 + 25 + 10	Recopie et vérification + initialisation de la structure+ Parcours du fichiers avec condition + Fonction pour donner nom et type de colonne	
Coût Total	**565**	**Estimation totale du coût**	

Conclusion

1 demi-page, plus tard