Spécifications

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

14/04/2017

Module *Projet*

Table des matières

Introduction

Ce document va décrire l'ensemble des exigences fonctionnelles que doit satisfaire notre produit final, c'est-à-dire les différentes fonctionnalités que notre application va fournir. Cette description va prendre en compte les caractéristiques des outils de développements choisis.

Notre outil, Python, est un langage de programmation hybride. On utilisera d'une part la programmation fonctionnelle pour les calculs, et d'autre part la programmation objet pour le développement des interfaces graphiques. Dans Python, le type de données n'est connu qu'à l'exécution (typage dynamique), par conséquent, ces types ne seront pas indiqués dans les signatures des fonctions et les structures des classes. Ils seront néanmoins définis dans des paragraphes explicatifs.

Pour les parties qui s'appuient sur une interaction avec l'utilisateur, notre démarche de description des fonctionnalités va essentiellement prendre en compte l'*expérience utilisateur*[1]. Cette description sera donc axée sur la qualification du résultat et du ressenti de l'utilisateur lors de la manipulation de l'interface fournie (une illustration à l'aide croquis), plutôt que sur les points techniques de l'application (fonctions et classes).

Les fonctionnalités de notre application seront présentées selon les modules de l'organigramme établi dans le cahier des charges. Ces modules eux-mêmes seront regroupés en packages. Ce document va donc décrire, pour chaque package de l'organigramme, les fonctionnalités de ses modules : d'abord ceux du package de chargement des données, ensuite ceux du package d'analyse descriptive des données et enfin ceux du package de l'interface web.

1 Package Chargement des données

Ce package est composé de deux modules qui ont pour fonction de traiter le fichier de données fourni : une vérification de son format et une analyse de son contenu. On pourra aussi parler de API puisque ce package peut être éventuellement livré en sortie.

1.1 Module Vérification format fichier

Ce module va vérifier le format du fichier de données fourni en entrée en 3 points. Il aura donc 3 fonctionnalités :

1. http ://uxdesign.com/ux-defined

1. Fonctionnalité de vérification de l'ouverture du fichier :

```
verifOuverture(fichierCSV)
```

Paramètres :

`fichierCSV` : `TextIoWrapper` - représente le fichier CSV fourni.

Retour : variable de type booléen.
Description : cette fonction prend en entrée le fichier CSV ouvert. Elle vérifie que le paramètre `fichierCSV` contient bien des informations représentant un fichier quelconque, et renvoie un booléen vrai si c'est le cas, faux sinon.

2. Fonctionnalité de vérification de l'extension du fichier ouvert :

```
verifExtenstion(fichierCSV)
```

Paramètres :

`fichierCSV` : `TextIoWrapper` - représente le fichier CSV fourni.

Retour : variable de type booléen.
Description : cette fonction prend en entrée le fichier CSV ouvert. Elle vérifie que l'information décrivant l'extension du fichier dans le paramètre `fichierCSV` correspond bien au format CSV, et renvoie un booléen vrai si c'est le cas, faux sinon.

3. Fonctionnalité de vérification de l'accessibilité en lecture du fichier :

```
verifLecture(fichierCSV)
```

Paramètres :

`fichierCSV` : `TextIoWrapper` - représente le fichier CSV fourni.

Retour : variable de type booléen.
Description : cette fonction prend en entrée le fichier CSV ouvert. Elle vérifie que le paramètre `fichierCSV` représentant le fichier possède bien la propriété d'accès en lecture. Un test de lecture sera aussi effectué sur le fichier. La fonction renvoie un booléen vrai si l'accès en lecture est permis, faux sinon.

1.2 Module Analyse contenu fichier

Ce module va analyser le contenu du fichier fourni en lisant, d'une part, les données de ce fichier une à une et d'autre part, en repérant les données erronées. Il aura donc 2 fonctionnalités :

1. Fonctionnalité de lecture du contenu du fichier CSV :

```
lecture(fichierCSV)
```

Paramètres :

`fichierCSV` : `TextIoWrapper` - représente le fichier CSV fourni.

Retour : structure contenant les données du fichier, le nombre de lignes et le nombre de colonnes (connus à partir de la taille de la structure).

Description : cette fonction prend en entrée le fichier CSV ouvert. Elle crée une structure pour y sauvegarder le contenu du fichier représenté par le paramètre `fichierCSV`. On lit ligne par ligne des caractères du fichier. A chaque fois qu'on détecte un caractère de séparation (une virgule, un point-virgule ou une tabulation), on stocke les caractères lus (la donnée) dans la structure.

2 Package Analyse descriptive des données

Ce package va être livré au client pour une intégration externe. On pourra donc parler de API.

3 Package Interface web

4 Glossaire des types

Dans ce glossaire, on va préciser la définition en Python des types des données qu'on a utilisé dans la description des fonctionnalités.

— `TextIoWrapper` : classe représentant les flux de texte bufferisés (entrées/sorties de texte avec une sauvegarde dans une mémoire tampon). Elle permet donc de représenter les fichiers de texte brut en particulier. Elle définit des méthodes de manipulation de ces flux.

Conclusion

Bilan : rappel ce qu'on a fait et ce que ça a apporté

Ouvertures :

- Mise en place des tests d'acceptations (comment on va mesurer, il faut définir une mesure et après la calculer)
- Difficultés

Dernière phrase positive.