# Licence 2 - Info 42

**Semestre 4 - Année 2018**

*Projet Principes des Systèmes d'Exploitation* • à rendre la semaine du 23 Avril

### Cadre général du projet

Le projet à pour but d'appliquer les concepts des systèmes d'exploitation afin de réaliser l'une des applications décrites dans la suite du document : 1) un outil de capture et d'analyse de tweets utilisant entre autre l'algorithme HITS, 2) un jeu des années 80 étendu avec un mode multi-joueurs et un mode réseau.

### Contraintes pour la réalisation

Le projet est à réaliser par groupe de deux étudiants. La programmation se fera en utilisant la plate-forme Java. Il n'est pas obligatoire de développer une interface graphique. Votre programme devra être conçu de façon modulaire (packages, interfaces, classes respectant une encapsulation stricte, classes utilitaires, constantes, fichiers de configuration, etc.) et il devra refléter la conception en couches d'un système d'exploitation (les couches et leurs fonctionnalités sont à détailler dans le rapport). Les structures de données que vous utiliserez ne sont pas nécessairement dynamiques (structure à base de références). L'emploi des membres de classe déclarés *static* devra être justifié.

Vous devez faire une présentation de votre projet deux semaines avant la semaine des examens (semaine du 23 avril). Ce jour là vous remettrez un rapport papier. Pour ce document, les consignes **strictes** de présentation sont : au moins 15 pages, police 12pts maximum, marges 2cm maximum. Le rapport doit comporter au moins les éléments suivants :

- **une analyse fonctionnelle** du sujet, précisant, entre autre, toutes les règles de fonctionnement de votre application et un découpage du problème en sous-problèmes (pour aboutir aux grandes lignes de l'architecture logicielle) ;
- **une description des structures de données** envisagées et retenues ;
- **la spécification des classes principales** et des méthodes essentielles ;
- **l'architecture logicielle** de votre application, c'est-à-dire, dans le cadre du module Info42, une conception en couches fonctionnelles à l'image de ce qui est réalisé dans l'architecture d'un système d'exploitation. Pour chaque couche vous spécifierez les classes qui implémentent les services de la couche fonctionnelle.
- une description des **algorithmes principaux** ;
- **un jeu de tests** montrant que votre programme fonctionne.

Il est conseillé de rédiger le rapport en utilisant LaTeX car il permet d'inclure facilement des extraits de code source avec une mise en évidence des éléments syntaxiques. On trouve des applications permettant de rédiger collaborativement des documents LaTeX (`https://fr.sharelatex.com/`). Il est recommandé d'utiliser des schémas pour illustrer vos propos. Des diagrammes UML comme le diagramme de classes ou de séquences peuvent être ajoutés à votre rapport.

Une archive [1] (tgz, jar ou zip) de l'ensemble de code source et de l'exécutable de votre programme devra être produite à la date du jour de la démonstration, puis envoyée par mail, ou déposée dans vos espace personnel (`public_html` de votre répertoire).

L'évaluation se fait sur la base du rapport papier, de la qualité technique de la réalisation, du déroulement de la démonstration. Lors de la démonstration vous devrez avoir préparé un scénario et des jeux de tests afin de présenter les fonctionnalités de votre application (vous disposerez de 10 minutes).

*Le descriptif de ce projet est volontairement synthétique. Vous devez/pouvez déposer des questions dans le forum de Plubel prévu à cet effet.*

## Sujet 1 : Capture, Stockage et Analyse de tweets

L'objectif est de développer un outil de capture, de stockage et d'analyse de tweets qui peut être appliqué dans différents domaines comme par exemple pour suivre des avis d'utilisateurs sur des produits (marketing), détecter des utilisateurs influents, des *fake-news*, construire un média thématique spécialisé, etc.

L'analyse des données recueilles sera effectuée avec l'algorithme (HITS) qui permet de déterminer et de classer les utilisateurs faisant autorité (*authorities*), c'est-à-dire ceux qui publient un contenu souvent repris et les utilisateurs qui diffusent les contenus (*hubs*).

Votre outil sera composé de plusieurs éléments (programmes) :

1. Les autres formats d'archive ne sont pas autorisés

— Un module *crawler* qui se connecte à Twitter et télécharge les nouveaux messages publiés ou bien des messages correspondant à des critères spécifiques (mots clés par exemple).
— Un entrepôt de tweets qui contient les tweets brut au format JSON qui seront repris par un module d'indexation.
— Un module d'indexation qui doit être multi-thread et ou multi-machines et qui extrait depuis l'entrepôt, pour chaque tweet, l'émetteur, la date, le contenu, les références à des URL externes, les hashtags, les citations d'autres utilisateur, les retweets. Ces éléments extraits seront stockés dans des hashtables et rendus persistant dans des fichiers.
— Un ensemble d'algorithmes qui peuvent être utilisés par un module d'interrogation pour rechercher et classer les tweets par utilisateur, hashtag, etc. Dans notre cas il s'agit principalement de l'algorithme HITS.

Votre réalisation devra faire apparaître clairement ces différents éléments. Pour l'algorithme HITS, vous devrez implanter une version itérative (la version utilisant valeurs propres et vecteurs propres n'est pas demandée).

La programmation réseau ne peut pas se limiter à la capture des tweets , votre système doit pouvoir fonctionner en équilibrant le charge sur plusieurs machines, par exemple la partie indexation et l'algorithme HITS devront être exécutés sur une autre machine qui viendra consulter les hashtables.

Vous devrez fournir des jeux de données (par exemple un fichier contenant plus de 100 000 tweets), les résultats produits et un scénario d'exécution afin de montrer la pertinence des résultats.

**Remarques :** pour la collecte de tweets, utiliser la bibliothèque Twitter4j `http://twitter4j.org/en/index.html`.

**Références sur l'algorithme HITS :**
— `https://fr.wikipedia.org/wiki/Algorithme_HITS` (suivre le lien externe)
— `https://en.wikipedia.org/wiki/HITS_algorithm` (page en anglais de Wikipedia assez détaillée, contenant des exemples d'algorithmes)
— `https://www.youtube.com/watch?v=zydSN8C1Et4` (avec le principe de l'algorithme expliqué pas à pas)

# Sujet 2 : Frogger

Frogger est un jeu des années 80 dont vous pouvez trouver les règles sur plusieurs sites, vous pourrez essayer également des versions émulées du jeu.

L'objectif est de réaliser le jeu en utilisant la plateforme Java et les principes des systèmes d'exploitation vus en cours. Les fonctionnalités à ajouter par rapport au jeu original sont les suivantes :
— mode collaboratif en réseau avec 2 variantes :
  1. objectif sauver le plus de grenouilles par équipe de $n$ joueurs ;
  2. sauver le plus de grenouilles avec un ou plusieurs participants perturbateurs possédant une grenouille carnivore dont l'objectif est de dévorer les grenouilles des participants l'équipe.
— mode compétition en réseau avec deux variantes :
  1. le premier ayant acheminé $n$ grenouilles est déclaré gagnant ;
  2. au bout de $m$ minutes celui qui a acheminé le plus de grenouilles et le vainqueur.
— passage de tondeuse ou de serpent sur le terplain central, présence d'alligator dans la rivière.
— classement par utilisateur (parties gagnées, niveaux etc.) ;
— serveur multi-parties : un joueur sélectionne un type de partie et donne son nom. Dans le cas d'une partie réseau, si au bout d'une minute le serveur n'a pas trouvé d'autre participant il bascule vers une partie normale (sans réseau ni variante). De plus, en fonction du classement du joueur, le serveur peut adapter la difficulté du jeu c'est-à-dire la rapidité de défilement et la densité des obstacles.

**Références :**
— `https://en.wikipedia.org/wiki/Frogger`
— Vidéo du jeu original `https://www.youtube.com/watch?v=l9fO-YuWPSk`
— Emulateur en ligne `http://froggerclassic.appspot.com/`
— La librairie JCurses portage en Java de la librairie curses d'Unix. Très utile pour simuler le jeu en console : `https://sourceforge.net/projects/javacurses/`