Licence 2 - Info 4B

Semestre 4 - Année 2025

Projet de l'UE Principes des Systèmes d'Exploitation

Démonstration la semaine du 7 Avril 2025 • Rendre le rapport et le code au plus tard le 6 Avril 23h59 sur Plubel

Cadre général du projet

Le projet a pour but d'appliquer les principes des systèmes d'exploitation afin de réaliser l'une des 3 applications décrites dans la suite du document : 1) une base de données orientée objet, 2) un jeu des années 80 étendu avec un mode multi-joueurs et un mode réseau, 3) le jeu de Core War qui consiste à simuler des combats de programmes pour conquérir la mémoire.

Contraintes pour la réalisation

Le projet est à réaliser par groupe de deux étudiants au maximum. La programmation se fera en utilisant la plate-forme Java. **Le projet doit être réalisé sur un système d'exploitation GNU/Linux (Debian $\geq$ 12). Il doit être compilable et exécutable sur les machines des salles informatiques de travaux pratiques.**

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), vous pouvez utiliser des tableaux, cependant les collections proposées par Java seront d'une grande utilité. L'emploi des membres de classe déclarés *static* devra être justifié. Il est également très fortement conseillé de respecter les conventions d'écriture de programmes Java qui sont des règles pour faciliter la compréhension et la maintenance du code (https://www.jmdoudoux.fr/java/dej/chap-normes-dev.htm). La lisibilité du code sera évaluée.

Vous devrez faire une présentation de votre projet avant les vacances de printemps (semaine du 7 au 11 avril 2025), produire un document et le code source de votre réalisation (à déposer sur Plubel la semaine précédente).

Pour le document, les consignes **strictes** de présentation sont : au moins 15 pages, police 12pts maximum, marges 2cm maximum. Ce 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), c'est-à-dire les classes les plus importantes et leurs liens ;
- **une description des structures de données** envisagées et retenues ;
- **la spécification des classes principales** et des méthodes essentielles ;
- **l'architecture logicielle détaillée** de votre application, c'est-à-dire, dans le cadre du module Info4B, une conception en couches fonctionnelles à l'image de ce qui est réalisé dans l'architecture d'un système d'exploitation (voir chapitre 1 du cours). Pour chaque couche vous spécifierez les classes qui implémentent les services de la couche fonctionnelle ;
- une description des **algorithmes principaux** ;
- **un jeu de test** montrant que votre programme fonctionne ;
- une section expliquant clairement la répartition des tâches entre les 2 étudiants du groupe.

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/, https://fr.overleaf.com/ ou sur le serveur `iemgit` accessible par VPN en demandant au préalable l'activation d'un compte). Pour le partage de code entre binôme l'outil git et son interface Web Github sont recommandés. Dans le document, il est conseillé 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.

Avant le 6 Avril 2025 23h59 vous devrez avoir déposé sur Plubel, dans l'espace du cours prévu à cet effet : le rapport au format PDF et une archive [1] (tgz, jar ou zip exclusivement) de l'ensemble de code source et de l'exécutable de votre programme.

L'évaluation se fait sur la base du rapport, 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).

Vous devez/pouvez formuler des questions dans le forum de Plubel prévu à cet effet.

Sujet 1 : Base de données orientée objet

L'objectif de ce projet est de fournir un service système de gestion de base de données qui, au travers du mécanisme de sérialisation, permet d'enregistrer des collections d'objets dans des fichiers, mais aussi de fournir au programmeur des fonctions pour rechercher des objets selon des critères numériques ou textuels.

Le projet comporte plusieurs programmes : un serveur, un client texte (en mode terminal), une bibliothèque client. Ces différentes parties doivent permettre à des utilisateurs authentifiés par le serveur et ayant installé un client sur leur poste de d'enregistrer des objets sur le serveur c'est-à-dire de les rendre persistant ou d'en rechercher d'autres et de les manipuler (lire, modifier ou supprimer).

La partie serveur doit permettre l'authentification des utilisateurs et la sauvegarde de collections d'objets. Le serveur tourne en permanence, il gère la concurrence, la sauvegarde des collections, leur indexation et répond aux demandes de consultation. Plusieurs clients doivent pouvoir se connecter au serveur en même temps. Votre serveur devra supporter le mode multi-utilisateurs, gérer la concurrence et les transactions.

Le programme client utilise la bibliothèque de fonctions pour se connecter au serveur, déclarer des collections d'objets à stocker, effectuer des ajouts d'objets, interroger des collections.

La bibliothèque de fonctions est un package Java que vous devez créer et qui est utilisé par le client (sur une autre machine) pour se connecter sur le serveur, effectuer les opérations CRUD (pour Create, Read, Update, Delete).

Références :

— https://fr.wikipedia.org/wiki/CRUD

Sujet 2 : Frogger

Frogger est un jeu des années 80 dont vous pouvez trouver les règles sur plusieurs sites Web, 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 de 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 est le vainqueur.

— passage de tondeuses ou de serpents sur le terre-plein central, présence d'alligators dans la rivière.

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

— classement par utilisateur (parties gagnées, niveaux etc.) ;
— serveur multi-clients et 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.

L'interface graphique n'est pas demandée, vous pouvez utiliser des caractères pour représenter les différents éléments du décor ou les personnages (la bibliothèque JCurses peut être utile dans ce cas). Pour un mode graphique, les bibliothèques Java Swing, JavaFX ou sdljava par exemple peuvent être utilisées.

Références :
— https://fr.wikipedia.org/wiki/Frogger
— https://www.youtube.com/watch?v=nJGqiV-C9Rc
— https://members.loria.fr/VThomas/mediation/JV_IUT_2015/

Sujet 3 : Core War

On se propose d'appliquer les concepts des systèmes d'exploitation afin de réaliser un simulateur *Mars* dont la description se trouve dans l'article de A. K. Dewdney cité dans les références et dont la traduction en français sera déposée sur la plateforme Plubel. Votre simulateur doit être capable d'exécuter simultanément plusieurs programmes écrits en *RedCode*. La taille mémoire de la machine est fixée à l'initialisation du jeu. Le programme doit déclarer un joueur gagnant et classer le ou les autres joueurs.

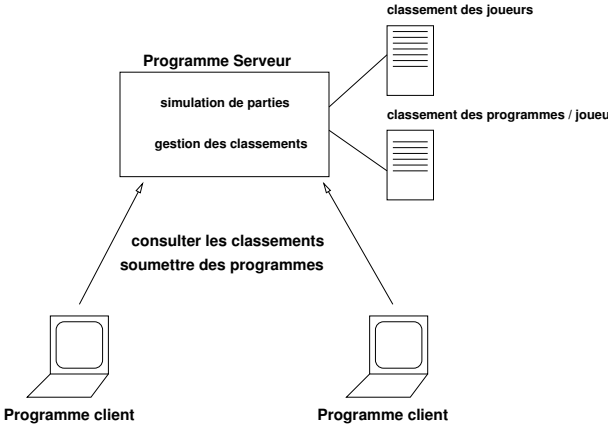

FIGURE 1 – Jeu de Core War en réseau

Les programmes *Mars* sont des fichiers texte que l'on charge en début de partie, qui sont interprétés lors de l'exécution (voir la description de la syntaxe dans les références).

En plus des éléments décrits dans les références, les fonctionnalités principales que vous avez à ajouter sont de deux types :
— par rapport à l'article original de Dewdney vous devez permettre la soumission de programmes par le réseau, c'est-à-dire développer une partie serveur et une partie client. Un même serveur peut héberger plusieurs parties se déroulant en même temps (figure 1). Une partie démarre lorsque chaque joueur participant a envoyé son programme ;
— par rapport à la gestion des joueurs et des programmes : à mesure des parties gagnées un joueur progresse dans le classement (stocké sur la machine qui héberge le programme serveur). On gardera aussi le *hit parade* des noms des programmes gagnants. Avant de lancer une partie, il est possible de consulter les classements.

Si vous souhaitez réaliser d'autres extensions du langage *RedCode*, votre interpréteur devra être capable de supporter deux syntaxes : celle des programmes en *RedCode Base* correspondant à la description du langage dans l'article de A. Dewdney et la syntaxe du *RedCode Étendu* correspondant à vos ajouts. On doit pouvoir sélectionner l'une des syntaxes au lancement des programmes.

Références :

- Article original en anglais : https://www.corewars.org/sciam/
- http://www.koth.org/index.html
- https://fr.wikipedia.org/wiki/Core_War