Projet de POCA — Role-Play-Game-Over-Internet (RPGOI)

Université Paris Diderot – Master 2

2 octobre 2018

1 Principe du projet

Le projet de POCA vise à vous faire progresser dans votre capacité *à concevoir un système objet extensible.*

Il se déroule en *deux étapes.* Dans la première étape, un sujet vous est fourni. Ce sujet est *volontairement* décrit de façon informelle, sans vous donner d'indication pour le réaliser. Vous devez donc définir vous-même les *requis* (**délivrable 1**) et l'*architecture* (**délivrable 2**) de votre projet.

Une *implémentation* écrite en SCALA de cette première version constitue le **délivrable 3**.

Vous recevrez ensuite un *autre sujet* qui est une extension du premier. Cette extension sera objet du **délivrable 4** (plus de détails seront fournis avec le deuxième sujet) et mettra à l'épreuve votre architecture initiale : un projet bien pensé *n'aura pas besoin de modifier* le code de la première version du projet pour traiter cette extension mais seulement de *rajouter de nouveaux composants.*

2 Sujet

Le projet de cette année consiste en la réalisation d'un système de "jeu de rôle sur table"[4] dématérialisé et augmenté.

D'après Wikipedia, un jeu de rôle sur table est ≪ un jeu de société dans lequel les participants conçoivent ensemble une fiction par l'interprétation de rôles et par la narration, dans le cadre de contraintes de jeu qu'ils s'imposent ≫. On distingue en général deux types de joueur : le maître du jeu et les personnages-joueurs. Une partie de jeu rôle commence par sa préparation : pour le maître de jeu, il s'agit de construire un scénario de jeu, qui va définir la "trame narrative" de la partie ; pour les personnages joueurs, il s'agit de préparer une "feuille de personnage" contenant des informations qui caractérisent leur personnage. Durant la partie de jeu de rôle à proprement parler, les personnages-joueurs communiquent pour décrire leurs actions (désirées) dans le monde. C'est au maître de jeu de valider les actions des joueurs, de décrire les scènes de jeu, et de réagir pour maintenir les joueurs dans la trame narrative qu'il s'est fixé.

Il existe déjà des "tables de jeux de rôle dématéralisées", comme par exemple *Roll20*[3] ou encore *Roliste-am*[2]. Ces plateformes permettent en général d'effectuer logiciellement les actions classiques d'une partie de jeu de rôle : lancer de dé, communication inter-joueur, stockage de la feuille de personnage, écoute de bandes sonores, visualisation d'images, ...En d'autres termes, ces logiciels permettent de reproduire via internet les interactions habituelles qui ont lieu entre les joueurs d'une partie de jeu de rôle.

Dans ce projet, on se propose non seulement de dématérialiser la table de jeu comme le font les logiciels décrits dans le paragraphe précédent mais aussi de profiter des nouvelles technologies pour *augmenter* les possibilités de jeu et la dimension de ce dernier. Quand une partie de jeu de rôle rassemble en général moins d'une dizaine de joueurs, le logiciel que nous souhaitons obtenir devra être capable de prendre en charge un grand nombre de joueurs (plusieurs centaines par exemple). Ainsi, on donnera au maître du jeu la possibilité de préparer des réponses automatiques à certaines actions des joueurs tout en lui permettant de réagir manuellement à des actions ou des situations qu'il n'aurait pas prévu [1].

1. Une fois cette réaction manuelle définie, elle devra alors être intégrée dans l'ensemble des réponses automatiques.

En l'honneur des trente ans du protocole IRC, on impose la contrainte technique suivante : l'interface utilisateur (pour les joueurs comme pour les maîtres de jeu) doit pouvoir être accessible à l'aide d'un client IRC. Toutes les fonctionnalités d'IRC (messages privés, canaux publiques, transfert par DCC, ...) pourront être utilisées pour améliorer l'expérience de l'utilisateur.

3 Travail demandé

3.1 Fonctionnement

Le projet doit être réalisé par groupe de 4 à 5 étudiants dans un dépôt privé sur GitHub. Yann Régis-Gianas créera votre dépôt pour vous si vous lui envoyez un email à `yrg@irif.fr` contenant le nom de votre projet ainsi que la liste des identifiants et des identités des membres de votre groupe. Les enseignants seront systématiquement inclus dans les membres du projet.

Vous devez suivre la méthode de développement *Scrum*[5] et une discipline de branchements et d'intégration bien définie (comme par exemple, *git-flow*[1]). La gestion du projet devra utiliser l'interface proposée par GitHub. Une solution d'intégration continue devra être mis en place à l'aide de Travis-ci.org. Une image Docker du logiciel doit toujours disponible.

Le groupe devra être structuré comme suit :

— Un étudiant doit être nommé *chef du projet*. Il est responsable de s'assurer de l'explicitation et de la répartition des tâches du projet ainsi que de la réalisation de ces dernières dans les délais impartis. Il est aussi responsable de l'évaluation et de l'intégration des *Pull Requests* dans l'arbre de développement. S'il en a le temps, il peut aussi prendre en charge certaines tâches de développement.

— Un étudiant doit être nommé *testeur*. Il est responsable de l'infrastructure d'intégration continue. Il doit s'assurer que la branche de développement principal contient toujours une version publiable du logiciel. S'il en a le temps, il peut aussi prendre en charge certaines tâches de développement.

Chaque groupe se verra affecter un enseignant dont le rôle sera de faire au moins une fois par semaine un retour sur l'état courant du projet. Chacun de ces rapports donnera lieu à une note sur 10 qui complètera pour moitié la note finale obtenue lors de la soutenance du projet. Par ailleurs, toutes les deux semaines, le groupe devra rencontrer son enseignant attitré pour échanger sur l'avancée du projet.

3.2 Délivrables

Note préliminaire : Les documents demandés doivent être rédigés avec soin. Cela signifie qu'ils doivent satisfaire les propriétés attendues par une documentation *technique* : on s'attend ainsi à y trouver une problématisation, une analyse et une argumentation sous la forme d'un discours structuré, clair et rationnel.

Pour l'ensemble des documents à rendre, on trouvera ainsi deux dates : la première date de rendu correspond à la première version du document ; elle sera relue immédiatement par votre enseignant pour vous permettre de produire une seconde version de meilleure qualité, à rendre avant la seconde date.

1. **Analyse des besoins (15 octobre / 28 octobre)**
   Dans un répertoire `deliverables/requirements/` de votre dépôt, vous devez nous soumettre une liste de requis (fonctionnels et non fonctionnels) que vous avez identifié et que votre projet s'engage à respecter. Vous définirez des *User stories* à l'aide du langage *Gherkins*.
2. **Architecture (28 octobre / 15 novembre)**
   Dans un répertoire `deliverables/architecture/` de votre dépôt, vous devez nous soumettre votre analyse du sujet et votre proposition de conception sous la forme d'une architecture décrite par un ou plusieurs diagrammes de classes UML et d'autres diagrammes de votre choix accompagnés bien sûr d'explications. Ce document sera fourni au format PDF et pourra suivre le plan suivant :

   interprétation du sujet Vous expliquerez de façon informelle ce que vous avez compris du sujet et de ces enjeux. Quels sont les problèmes techniques et conceptuels que vous avez exhibés ?

   concepts Dans cette section, vous définirez les concepts utilisés pour modéliser le problème ainsi que les invariants essentiels du système.

description de l'architecture Vous donnerez ici les diagrammes UML décrivant votre architecture et surtout sa justification.

extensions envisagées Vous énumérez ici les généralisations et les extensions que vous avez imaginées et vous expliquerez pourquoi votre architecture permet de les traiter facilement.

3. **Implémentation (25 novembre)**
   Vous devez nous soumettre la première version complète de votre projet. Nous vous communiquerons alors la version 2 du présent sujet.

Références

[1] Gitflow. https://nvie.com/posts/a-successful-git-branching-model/.

[2] Rolisteam. http://www.rolisteam.org/.

[3] Roll20. https://roll20.net/.

[4] Wikipedia - Jeu de rôle sur table. https://fr.wikipedia.org/wiki/Jeu_de_r%C3%B4le_sur_table.

[5] Wikipedia - Scrum. https://fr.wikipedia.org/wiki/Scrum_(d%C3%A9veloppement).