 Département de génie logiciel et des TI	PROJET LOG791/ GTI791	DOCUMENT NO.	DATE AAAA-MM-JJ	VERSION X.Y
	TITRE Titre du projet - Proposition		PAGE 1	PAGES 8


PROPOSITION
Projet spécial
Département de génie logiciel et des TI

Systèmes hautement scalable en Rust

Auteurs
Robyn Girardeau
GIRR25029403

Professeur superviseur
Fabio Petrillo

Date
3 mars 2024

 Département de génie logiciel et des TI	PROJET LOG791/ GTI791	DOCUMENT NO.	DATE AAAA-MM-JJ	VERSION X.Y
	TITRE Titre du projet - Proposition		PAGE 2	PAGES 8

Suivi des changements

*A – Ajouté M – Modifié S – Supprimé

NUMÉRO DE VERSION	DATE aaaa/mm/jj	NUMÉRO DE FIGURE, TABLE OU SECTION	A* M S	BRÈVE DESCRIPTION DU CHANGEMENT	NUMÉRO DE DEMANDE CHANGEMENT
1	2024/03/06	1	A	Première itération	
1	2024/03/06	2	A	Première itération	
1	2024/03/06	3	A	Première itération	
1	2024/03/06	4	A	Première itération	
1	2024/03/06	6	A	Première itération	
2	2024/03/08	1	M	Ajout du contexte	
2	2024/03/08	2	M	Améliorations légères	
2	2024/03/08	Équipe et assignement des tâches	S	Équipe solo	
2	2024/03/08	5	A	Complétion	
2	2024/03/08	7	A	Complétion	
2	2024/03/08	Annexe A	A	Complétion	



 Département de génie logiciel et des TI	PROJET LOG791/ GTI791	DOCUMENT NO.	DATE AAAA-MM-JJ	VERSION X.Y
	TITRE Titre du projet - Proposition	PAGE 3		PAGES 8

TABLE DES MATIÈRES

1.	Problématique et contexte	4
2.	Objectifs du projet	4
3.	Méthodologie	5
4.	Livrables et planification	5
4.1	Description des artefacts	5
4.2	Planification	5
5.	Risques	6
6.	Techniques et outils	6
7.	Références	7
	ANNEXE A : Plan de travail	8

 Département de génie logiciel et des TI	PROJET LOG791/ GTI791	DOCUMENT NO.	DATE AAAA-MM-JJ	VERSION X.Y
	TITRE Titre du projet - Proposition		PAGE 4	PAGES 8

1. PROBLÉMATIQUE ET CONTEXTE

Rust est un langage de programmation assez récent (2006) qui prend de plus en plus d'ampleur, surtout dans le domaine de programmation système. Il est maintenant utilisé dans les systèmes d'opération Windows et Linux.

Aujourd'hui, les serveurs de jeux massivement multijoueur sont souvent programmés en C/C++, Java ou C#. Rust entre aujourd'hui dans le champ des possibilités grâce à sa gestion de la mémoire sûre et sans « garbage collector ». Il amène le niveau de performance des langages systèmes comme C++ tout en ayant une approche plus moderne et haut niveau comme C# et Java. La communauté Rust grandissante partage également beaucoup de bibliothèques de qualité via Cargo, le gestionnaire de paquets de Rust.

L'enjeu consiste donc à explorer les possibilités de développement de serveur de jeu massivement multijoueur avec Rust, en incluant l'apprentissage du langage.


2. OBJECTIFS DU PROJET

Le projet a pour potentiel but d'être utilisé comme sujet pédagogique par les futurs étudiants en génie logiciel. Le projet se doit donc d'être développé selon les règles de l'art pour présenter les bonnes pratiques en architecture, test, documentation et intégration continue.

Le premier objectif est d'évaluer la plateforme Rust dans le contexte des services hautement scalable et fiable pour jeu massivement multijoueur. Le second est de développer un serveur hautement scalable et fiable pour jeu massivement multijoueur.

La première partie du serveur concerne un serveur de « matchmaking ». Les tests mesureront la performance du serveur lorsqu'il est dupliqué sur plusieurs microservices. C'est-à-dire, la stabilité du serveur, la gestion des pannes, le temps de latence, le nombre maximum de clients supportés... Il faudra donc aussi développer un client de test. L'objectif est de s'approcher à 1 million de clients simultanés. Il faut toutefois pouvoir aussi assumer l'exécution de ces clients.

S'il reste du temps au projet, la partie 2 du serveur concerne un serveur de jeu, possiblement de tir.

 Département de génie logiciel et des TI	PROJET LOG791/ GTI791	DOCUMENT NO.	DATE AAAA-MM-JJ	VERSION X.Y
	TITRE Titre du projet - Proposition		PAGE 5	PAGES 8

3. MÉTHODOLOGIE

Le projet est approché en plusieurs étapes.

1. Apprentissage de Rust
2. Analyse du problème
3. Recherche des bibliothèques adéquates
4. Conception architecturale
5. Prototypage d'une première itération
6. Mesure des performances.
7. Développement du produit final.
8. Mesure des performances.
9. Rapport et conclusion.


4. LIVRABLES ET PLANIFICATION

4.1 Description des artefacts

Nom de l'artefact	Description
Code	Le code de tout le projet est ouvert et hébergé sur Github.
Documentation	La documentation est incluse en format wiki sur Github ainsi que sous forme de commentaires dans le code.
Dockerfiles	Les fichiers Docker sont inclus pour améliorer le déploiement et unifier l'environnement d'exécution.
Tests unitaires	Des tests sont inclus pour assurer le bon fonctionnement des projets.
Tests de performance	Des tests de performance sont inclus pour mesurer le nombre de clients supportés.
Intégration continue	Le projet inclut des fichiers d'actions Github pour automatiser les tests sur requête de tirage.
Rapport	Le rapport final contient toutes les informations du projet, les démarches, les possibilités étudiées ainsi que les raisons qui ont mené aux choix finaux.

4.2 Planification

Voir Annexe A.

 Département de génie logiciel et des TI	PROJET LOG791/ GTI791	DOCUMENT NO.	DATE AAAA-MM-JJ	VERSION X.Y
	TITRE Titre du projet - Proposition		PAGE 6	PAGES 8

5. RISQUES

Risque	Impact	Probabilité	Mitigation / atténuation
Difficulté à prendre Rust en main	Moyen	Haute	Commencer l'apprentissage avant le début de la session pour avoir plus de temps pour travailler sur le projet.
Surfaire	Moyen	Haute	Il est très possible de développer trop d'outils et de fonctionnalités pour au final manquer de temps. C'est pourquoi le projet commence seulement par le serveur de matchmaking.

6. TECHNIQUES ET OUTILS

Outils de traçabilité et documentation:

- OneNote
- Google Doc
- Github Issues
- Github commits
- Github pull requests

Outils de développement:


- Visual Studio Code

Outils de communication:

- Discord
- Courriel
- Zoom

Outils de génération et déploiement:

- Docker
- Github Actions

 Département de génie logiciel et des TI	PROJET LOG791/ GTI791	DOCUMENT NO.	DATE AAAA-MM-JJ	VERSION X.Y
	TITRE Titre du projet - Proposition		PAGE 7	PAGES 8

7. RÉFÉRENCES

Rust

<https://doc.rust-lang.org/rustdoc/what-is-rustdoc.html>
<https://fasterthanli.me/articles/a-half-hour-to-learn-rust>
<https://stevedonovan.github.io/rust-gentle-intro/readme.html>

Crates (caisses, librairies)

Bases de données

<https://www.mongodb.com/docs/drivers/rust/current/>
<https://docs.rs/redis/latest/redis/>

ECS

https://docs.rs/bevy_ecs/latest/bevy_ecs/
<https://github.com/amethyst/legion>

Injection de dépendances

<https://github.com/Nashenas88/coi-actix-web>
<https://crates.io/crates/nject>
<https://github.com/snuk182/indep>

Http

<https://actix.rs>
<https://github.com/tokio-rs/axum>
<https://hyper.rs>
<https://github.com/cloudflare/pingora>

Swagger


<https://github.com/OpenAPITools/openapi-generator>

Exécution asynchrone et TCP

<https://tokio.rs>
<https://github.com/rayon-rs/rayon>

Analyse et génération de code

<https://crates.io/crates/syn>
<https://docs.rs/reflect/latest/reflect/>
<https://github.com/danielhenrymantilla/inheritance-rs>

 Département de génie logiciel et des TI	PROJET LOG791/ GTI791	DOCUMENT NO.	DATE AAAA-MM-JJ	VERSION X.Y
	TITRE Titre du projet - Proposition		PAGE 8	PAGES 8

ANNEXE A : PLAN DE TRAVAIL

#	Commence	Termine	Efforts estimés*	Tâches/Jalon	Livrable(s)/Artéfacts	Responsable(s)
1				Recherche de projet		Robyn Girardeau
2				Remise de la fiche de renseignements	Fiche de renseignements	Fabio Petrillo
2.1	28 février	28 février	2 heures	Rencontre – professeur superviseur		Fabio Petrillo, Robyn Girardeau
2.2	3 mars	13 mars	10 heures	Remise de la proposition de projet	Proposition de projet	Robyn Girardeau
3.	28 février	17 août	20 heures	Apprentissage de Rust		Robyn Girardeau
4.	6 mai	28 juin	50 heures	Première itération: prototypage	Documentation	Robyn Girardeau
4.1	28 février	28 juin		Exploration des bibliothèques disponibles	Documentation	Robyn Girardeau
4.2	6 mai	28 juin		Conception d'une première architecture	Documentation	Robyn Girardeau
4.3	6 mai	28 juin		Implémentation du prototype	Code, Dockerfiles, Tests unitaires, Tests de performance, Intégration continue..	Robyn Girardeau
4.4	24 juin	24 juin		Rencontre – professeur superviseur		Robyn Girardeau
4.5	24 juin	28 juin		Remise du rapport d'étape	Rapport d'étape	Robyn Girardeau
5.	1 juillet	17 août	50 heures	Deuxième itération: développement final	Documentation	Robyn Girardeau
5.1	1 juillet	7 août		Implémentation du serveur	Code, Dockerfiles	Robyn Girardeau
5.2	1 juillet	7 août		Implémentation du client	Code, Dockerfiles.	Robyn Girardeau
5.3	1 juillet	17 août		Implémentation des tests	Tests unitaires, Tests de performance, Intégration continue.	Robyn Girardeau
5.4	29 juillet	29 juillet	2 heures	Rencontre – professeur superviseur		Fabio Petrillo, Robyn Girardeau
6	29 juillet	7 août	10 heures	Présentation	Présentation	Robyn Girardeau
6.1	29 juillet	17 août	20 heures	Remise du travail	Rapport	Robyn Girardeau

* En heures. Doit totaliser au moins 135 heures par personne