Rust Sujet projet

Projet : Essaim de Robots pour l'Exploration et l'Étude Astrobiologique (EREEA)

Sujet

Ce projet sera à réaliser en groupe de 2 à 4 étudiants.
La note sera attribuée au groupe. Si un élève ne participe pas, à moins qu'il n'y ait un arrangement entre vous, il est important de m'en informer et si possible, pas à la dernière minute.

Développez un essaim de robots autonomes spécialisés pour l'exploration spatiale et la recherche astrobiologique. Ces robots collaboreront pour mener des missions d'exploration et de recherche sur des corps célestes (planètes, lunes, astéroïdes) afin de recueillir des données sur la géologie, la chimie, et les potentiels signes de vie.

Le projet est à rendre sur GitHub. Merci de m'envoyer un email avec les noms des participants et un lien utilisable à cette adresse : swann.herrera@intervenants.efrei.net
Si vous souhaitez faire un projet privé, merci de m'inviter sur GitHub (SwannHERRERA).

Objectif pédagogique

L'objectif technique du projet est de vous faire pratiquer Rust et les différents styles de programmation qu'il propose, tout en mettant l'accent sur l'aspect performance, notamment sur l'utilisation de modèles concurrents.

Composants du projet

Map

Sur une carte en 2D avec des obstacles, les bordures peuvent être des obstacles ou vous pouvez créer une map "sphérique" (en bonus).
La carte et plus généralement la simulation peuvent être modélisées dans le terminal, ou, en bonus, en utilisant d'autres outils (moteur de jeux vidéo comme Bevy, communication avec un système d'affichage tiers, via une FFI, un SDK ou le réseau).

Les obstacles de la carte doivent être générés à partir d'une fonction de noise.

Toutes les générations doivent être reproductibles à partir d'une seed aléatoire.

Libre à vous d'implémenter différents types de maps, d'une taille configurable...

La map doit contenir 3 ressources : l'énergie, les minerais et les lieux d'intérêt scientifique. L'énergie et les minerais sont consommables, ce qui signifie que chaque unité récoltée doit disparaître de la carte (une seule fois).

Robots

- **Conception Modulaire :** Chaque robot aura une conception modulaire, permettant des configurations variées selon les besoins spécifiques de la mission.
- **Spécialisation :** Différents modules seront spécialisés, par exemple : analyse chimique, forage, imagerie haute résolution, etc.

Apprentissage en Essaim

- **Reconnaissance de Terrain :** Les robots seront équipés de capteurs pour analyser et cartographier les terrains, identifiant les zones d'intérêt scientifique. Ils collecteront des échantillons pour analyse (qui ne se fait pas directement sur l'exoplanète).
- **Transmission des données sur la planète :** Les robots ne seront pas informés des découvertes des autres robots avant leur retour à la station. Une fois à la station, les informations des robots sont partagées avec la station (à la manière de git), avec gestion des conflits. Le robot connaît la taille de la carte.
- **Comportement des robots :** Certains robots peuvent être de type explorateur et chercher à compléter la carte de la station, d'autres vont collecter l'énergie, les minerais, ou aller sur les lieux d'intérêt scientifique.
- **(Bonus) Transmission des données avec la Terre :** La station doit partager des informations avec la Terre, ce qui peut être représenté par l'interface graphique. L'idée est d'avoir une coupure logicielle entre la simulation et l'affichage (réseau/communication entre les threads ou les processus).
- **Comportement de la station :** La station collecte les informations scientifiques pour les transmettre à la Terre, ainsi que les minerais et l'énergie pour créer plus de robots (vous choisirez le modèle de décision pour la création d'un robot).

Évaluation

L'évaluation du projet se basera sur plusieurs critères :

- **Qualité du Code :** Clarté, organisation, et utilisation des bonnes pratiques en Rust.
- **Qualité des tests :** Intérêt du test, documentation, type de test implémenté.
- **Fonctionnalités :** Implémentation correcte et efficace des fonctionnalités requises.
- **Concurrence :** Gestion efficace et sécurisée de la concurrence.
- **Créativité et Innovation :** Approche originale dans la conception de la simulation.
- **Documentation :** Concise, incluant des ADR, un README, un changelog.
- **Effort de travail :** Oui, c'est purement arbitraire, mais basé sur l'étude du répertoire git.

Tous les choix d'architecture devront être rédigés dans des ADR. Je vous conseille d'utiliser le template d'EdgeX.

Ressource potentiellement utile

- ECS
- Event sourcing
- Noise
- Model de concurrence
  - Programmation par acteur
  - Worker pools
  - shared memory
  - Async-IO (faut que je trouve des ressources)