Programmation répartie

SDR 2022 / Labo 1

Temps à disposition : 6 périodes.
Travail à réaliser par groupe de 2 étudiants.
La présence aux labos est obligatoire.
En cas de copie manifeste entre les rendus de deux labos, tous les étudiants des deux groupes se verront affecter la note de 1.

Labo distribué le vendredi 30 septembre 2022 à 14h55. Au cours cette séance, créez un repo github ***privé*** *qui contiendra le projet et dont vous donnerez l'accès à l'enseignant et à l'assistant (comptes* ***patricklac*** *et* ***raphaelracine*** *sous github) et dont vous leur communiquerez l'adresse par email avec le nom des étudiants du groupe. Vous devrez ensuite pousser votre travail en l'état sur ce repo au moins une fois par semaine, en début de chaque séance de labo.*

A rendre le vendredi 21 octobre 2022 à 14h55. *Un readme indiquant comment utiliser votre programme et précisant ce qui a été réalisé, ce qui fonctionne et ce qui reste à faire, devra être déposé dans votre repository. Un email rappelant vos noms et repository confirmera votre rendu qui sera accessible dans la branche master (ou main).*

Objectifs

- Écrire un premier programme de complexité moyenne en Go (Golang) et se familiariser avec un environnement de programmation Go
- Mettre en œuvre le principe « Communicating Sequential Processes ».
- Réaliser ses premières communications TCP-IP en mode Socket connecté (TCP) en GO.
- Développer une application client/serveur qui servira de base pour mettre ensuite en œuvre, dans le labo suivant, un algorithme sur un réseau réparti.

Limites

- Il n'est pas demandé de gérer la persistance des données au-delà de l'exécution du serveur.
- L'interface utilisateur sera en mode commande.

Contraintes et qualité du code source

Le non-respect des consignes ci-dessous ou un code de mauvaise qualité seront sanctionnés par des points négatifs dans l'évaluation.

Vous devez respecter les bonnes pratiques de conception (séparation du code concernant l'interface utilisateur et celui concernant les communications, packages, objets et interfaces appropriés) et les bonnes pratiques du langage GO.

Il est demandé de créer votre code dans un go module pour permettre la résolution des noms de package de façon indépendante du GOPATH et du dossier d'installation de votre projet. L'utilisation de packages externes est prohibé, sauf autorisation explicite de l'enseignant.

Les noms des variables et des fonctions doivent être suffisamment explicites. Des commentaires seront ajoutés judicieusement (ni trop ni pas assez) dans le code source pour la compréhension des passages difficiles. L'utilitaire « godoc » doit permettre d'afficher une documentation claire de l'utilisation de vos packages et fonctions.

La concurrence d'accès aux données sera gérée selon le principe « Communicating Sequential Processes », au moyen de goroutines et de canaux. La synchronisation par bloc est interdite.

Tests (10 points)

Il est demandé des tests d'intégration automatisés en mode client-serveur, le programme de test se comportant comme le seul client d'un serveur venant d'être lancé. Toutes les requêtes au serveur seront testées avec leurs retours positifs (pas d'erreurs) et négatifs (paramètres incorrects, indisponibilités, …).

D'autre part, on doit pouvoir vérifier en mode manuel qu'il ne présente pas de risques quant à la concurrence d'accès aux données.

Le serveur doit pouvoir être lancé en mode « debug/trace », ce qui aura pour effet de ralentir artificiellement le passage aux points critiques afin de le mettre en situation d'accès concurrent. On activera alors 2 clients selon un mode d'emploi documenté dans le readme et des traces d'exécution très lisibles nous permettront de vérifier la bonne sérialisation des accès concurrents.

Par ailleurs, l'application doit passer le test du « go race » en multi-utilisation.

Application à réaliser

Dans ce premier labo, il s'agit de développer une petite application client/serveur permettant la répartition de bénévoles pour l'organisation de manifestations.

Un organisateur peut créer une manifestation en précisant quels sont les postes à occuper et combien il attend de bénévoles par poste. Les bénévoles peuvent ensuite s'inscrire à un poste dans les limites du nombre de bénévoles attendus pour celui-ci. Lorsque l'organisateur qui a créé la manifestation la termine, il n'est plus possible de s'y inscrire.

Une commande permet de lister les manifestations et plus spécifiquement la répartition des bénévoles pour une manifestation donnée.

Les utilisateurs sont identifiés par leur nom / mot de passe à fournir pour certaines commandes.

Fonctionnalités

1. Initialisation, connexion, utilisation (10 points)

En suivant les directives de votre readme, on doit pouvoir cloner votre repo, lancer le serveur d'application et plusieurs clients. Les déconnexions ou abort des clients seront correctement gérés par le serveur.

Dès son lancement le serveur définit à partir d'un fichier de configuration les utilisateurs (organisateurs ou bénévoles) avec leur nom et mot de passe ainsi que quelques manifestations avec leur répartition de bénévoles afin de faciliter le test de l'application.

Le client ne doit pas être un simple netcat mais doit prendre en charge le formatage des messages échangés avec le serveur et celui des entrées/sorties console. La saisie des mots de passe doit être effectuée sans écho.

A la connexion, le client affiche un message de bienvenue et d'aide à l'utilisation de l'application. Une même connexion pourra être successivement utilisée pour le compte de plusieurs utilisateurs. Il n'y a pas de session : certaines commandes demanderont simplement le nom et mot de passe de l'utilisateur.

Votre readme devra décrire clairement comment utiliser votre application et ce qui fonctionne ou non.

L'application doit fonctionner dans les environnements Windows, Mac et Linux.

2. Création et clôture d'une manifestation (10 points)

Une manifestation est créée en précisant son nom, le nom et mot de passe de l'utilisateur qui le crée (son organisateur), le nom des postes à pourvoir et le nombre de bénévoles attendus par poste. Les manifestations sont numérotées ainsi que les postes pour permettre leur référence dans les autres commandes.

Dès sa création, tous les bénévoles pourront s'y inscrire (point suivant) jusqu'à sa clôture par une commande qui précise pour vérification le nom et mot de passe de son organisateur et le numéro de cette manifestation.

3. Inscription d'un bénévole (10 points)

Un bénévole peut s'inscrire à une manifestation « ouverte » en précisant son nom et mot de passe, le numéro de la manifestation et le numéro du poste qu'il a choisi. S'il s'inscrit ultérieurement à un autre poste de la même manifestation, son ancienne inscription sera effacée.

Si le nombre de bénévole prévu pour un poste est déjà atteint, l'inscription sera refusée.

4. Liste et visualisation de la répartition des manifestations (10 points)

Ces commandes ne demandent pas d'identification de l'utilisateur.

Une commande permettra de lister toutes les manifestations avec leur numéro, leur nom, le nom de leur organisateur et le fait qu'elles soit « ouvertes » ou non. En précisant le numéro d'une manifestation, tous les postes (avec leur numéro, nom et nombre maximum de bénévoles) seront affichés.

Une autre commande permettra d'afficher la répartition des postes pour une manifestation de numéro donné. On affichera un tableau avec en ligne les noms des bénévoles ayant répondu et en colonne les numéros des postes prévus. Les cases correspondant aux inscriptions de bénévoles sur des postes seront marquées. Les autres seront laissées à blanc. En haut de colonne, le nombre de bénévoles attendus sera affiché.