Implementação de Jogo Multiplayer Usando RPC em Go

Introdução

Este documento descreve a especificação para modificar um jogo de jogador único escrito em Go para um ambiente multiplayer utilizando RPC. O objetivo é permitir que múltiplos jogadores se conectem a um servidor central e compartilhem suas posições e ações com os demais jogadores. A comunicação sempre será iniciada pelos clientes, que irão periodicamente buscar atualizações do estado do jogo no servidor.

O trabalho deve ser desenvolvido a partir do código-fonte do Trabalho 1 da disciplina ou a partir do código original disponibilizado pelo professor no Moodle. Não serão aceitos trabalhos nos quais a implementação foi realizada do zero, a menos que seja previamente autorizado pelo professor.

Arquitetura do Sistema

Servidor de Jogo:

- O servidor é responsável por gerenciar uma sessão de jogo e manter o estado atual do jogo (ex: lista de jogadores, posição atual de cada jogadores, número de vidas, etc.).
- O servidor NÃO deve manter uma cópia do mapa do jogo e a lógica de movimentação de personagens e de funcionamento do jogo NÃO deve ser movida para o servidor.
- O servidor não deve conter interface gráfica.
- As requisições recebidas pelos clientes e as respostas retornadas devem ser impressas no terminal para permitir depuração durante do jogo.

Cliente do Jogo:

- O cliente possui a interface onde o jogador interage com o jogo e toda a lógica de movimentação do personagem e de funcionamento do jogo.
- Ele se conecta ao servidor para obter o estado atual do jogo (ex: lista de jogadores e a posição de cada um) e envia comandos de movimento e interação ao servidor.
- Deve possuir uma thread (goroutine) dedicada para buscar periodicamente atualizações do estado do jogo no servidor e atualizar o seu estado local.

Requisitos de Comunicação e Consistência

- Toda comunicação é iniciada pelos clientes. O servidor apenas responde.

- Todas as chamadas de procedimento remoto devem ter tratamento de erro com reexecução automática em caso de falha.
- É implementar garantia de execução única (exactly-once) dos comandos enviados que modificam o estado do servidor:
    - Cada comando deve incluir um sequenceNumber.
    - O servidor deve manter o controle de comandos processados por cliente para evitar reexecução em caso de retransmissão.

Entrega

O trabalho pode ser realizado em grupos de até 4 integrantes previamente cadastrado no Moodle. Para permitir que todas as apresentações sejam realizadas em duas aulas, o número máximo de grupos foi limitado (ver Moodle). A entrega deverá ser feita como um arquivo ".zip" contendo os arquivos fonte desenvolvidos, bem como informações para sua compilação e execução. É necessário escrever um pequeno relatório explicando a implementação do servidor RPC, as modificações no cliente e a estratégia utilizada para implementar e validar a garantia de execução única. Atente para os prazos de entregas e apresentação definidos no Moodle.

Critérios de Avaliação do Trabalho

Critério	Descrição	Peso
Funcionalidade do jogo multiplayer	O jogo funciona corretamente com múltiplos jogadores interagindo de forma estável e consistente?	2,0
Comunicação RPC entre cliente e servidor	A comunicação foi implementada corretamente usando RPC, com chamadas remotas claras e funcionais seguindo especificação? **Atenção:** Não serão aceitas soluções que movam toda a lógica do jogo para servidor.	2,0
Concorrência e sincronização (mutex)	Tata corretamente a concorrência nos acessos ao estado do jogo (cliente e servidor), utilizando mutex ou técnicas equivalentes?	1,0
Execução Única dos Comandos	Cada comando possui sequenceNumber e o servidor gerencia corretamente os comandos para evitar reexecuções (exactly-once).	2,0
Tratamento de erros e reenvio de comandos	O cliente lida com falhas e reenvia comandos com o mesmo sequenceNumber em caso de falha?	1,0
Domínio sobre a implementação	O grupo demonstra conhecimento técnico e consegue explicar suas decisões e funcionamento do sistema?	2,0