Universidade Federal De Pelotas

Ciências Da Computação

Redes de Computadores

CRIAÇÃO DE PROTOCOLO DE APLICAÇÃO

Diego Aquino

Gabriel Oliveira

Gustavo Reginatto

1. Introdução:

Este relatório é referente ao desenvolvimento de um jogo de combate de turno inspirado nas regras de Pedra, Papel e Tesoura e o sistema de fraqueza de tipos do Pokémon, com foco na definição do protocolo usado na camada de aplicação, discutindo o objetivo e funcionalidade das características implementadas.

O jogo funciona com um sistema de turno no qual cada jogador deve escolher uma ação específica para tentar obter o melhor resultado possível utilizando os recursos disponíveis. Cada jogador deve selecionar uma de três habilidades e logo atribuir um de cinco elementos à habilidade. Os elementos definem a eficácia que a habilidade terá contra a combinação de habilidade e elemento escolhida pelo outro jogador.

O objetivo é zerar os pontos de vida do adversário antes que ele zere os seus, enquanto tenta balancear o gasto de mana ofensivo e a recuperação de mana que exige passar o turno sem atacar.

2. Características do Protocolo

- Arquitetura: Cliente-Servidor, tendo um servidor central que gerencia o estado das partidas e a lógica do jogo, podendo múltiplos clientes se conectarem a ele;
- Modelo de Conexão: Com estado, o servidor mantém informações persistentes sobre as partidas, como pontos de vida, mana e o último elemento utilizado;
- Persistência: Persistente, a conexão TCP entre o cliente e o servidor é mantida ativa durante a partida inteira, garantindo comunicação contínua;
- Modo de Comunicação: Push, o servidor envia atualizações do estado do jogo e as notificações de turno para os clientes assim que os eventos ocorrem, sem que os clientes precisem solicitá-las;
- Controle: Na banda, as mensagens de controle (início/fim de jogo, controle de turno) e os dados da aplicação (jogadas, dano) são transmitidos pelo mesmo canal de comunicação TCP.

3. Tipos de mensagens enviadas pelo cliente

- JOIN_GAME: Enviada pelo cliente para solicitar a entrada em uma nova partida.
- PLAY_MOVE: Enviada pelo cliente durante seu turno, contendo a ação escolhida (elemento e habilidade).

4. Tipos de mensagens enviadas pelo servidor

- LOGIN_SUCCESS: Enviada após um JOIN_GAME bem sucedido, confirmando a entrada do jogador na fila de espera para uma partida.
- LOGIN_FAIL: Informa que a tentativa de JOIN_GAME falhou, por exemplo um nome de jogador repetido ou inválido.
- GAME_START: Notifica os jogadores que uma partida está iniciado, enviando o estado inicial.
- GAME_UPDATE: Após cada jogada, envia o estado atualizado do jogo, com o HP e a mana atuais por exemplo.
- YOUR TURN: Informa o cliente que é sua vez de jogar.
- GAME_END: Anuncia o final da partida e informa o vencedor.
- ERROR: Envia uma mensagem de erro genérica, seja por habilidade inválida, falta de mana ou outra invalidez.

5. Formato das mensagens e campos de cabeçalho

A estrutura geral:

```
{
  “type”: “TIPO_DA_MENSAGEM”,
  “game_id”: “IdentificadorDaPartida”,
  “payload”: { … }
}
```

Campos do *payload* por tipo de mensagem, com exemplos:

- JOIN_GAME

```
{
  “player_id”: “nome_do_jogador”
}
```

- PLAY_MOVE

```
{
  “element”: “fire”,
  “ability_id”: 1
}
```

- GAME_START

```
{
  “players”: {
    “jogador1”: { “hp”: 100, “mana”: 50 },
    “jogador2”: { “hp”: 100, “mana”: 50 }
  }
  “msg”: “Jogo iniciado!”
```

}

- GAME_UPDATE

{
“msg”: “Jogador1 usou Ataque Forte causando 30 de dano em jogador2!”,
“players”: {
“jogador1”: { “hp”: 100, “mana”: 35},
“jogador2”: { “hp”: 70, “mana”: 50 }
}
}

- YOUR_TURN

{
“msg”: “Sua vez de jogar!”
}

- GAME_END

{
“msg”: “Jogador1 venceu o jogo!”
“winner”: “jogador1”
}

- ERROR

{
“msg”: “Não é sua vez.”
}

6. Especificação dos Campos

Campo	Tipo	Valores Possíveis
type	string	JOIN_GAME, PLAY_MOVE, GAME_START, GAME_UPDATE, YOUR_TURN, GAME_END, ERROR, etc.
player_id	string	Identificador único do jogador (ex: "nome_do_jogador").
game_id	string	ID único da partida, gerado pelo servidor (ex: "Game1").
element	string	"fire", "water", "plant", "electric", "earth".
ability_id	integer/string	1, 2, 3 ou "pass".
hp	integer	Pontos de vida do jogador (0 a 100).

mana	integer	Pontos de mana do jogador (0 a 50).

6. Considerações Finais

O protocolo desenvolvido fornece uma estrutura simples, mas eficiente, para o gerenciamento de partidas de um jogo de turno. A escolha do modelo cliente-servidor com conexão persistente via TCP garante confiabilidade e consistência durante as partidas.

Esse trabalho demonstrou, na prática, como conceitos de rede de computadores como protocolos de aplicação podem ser aplicados em um ambiente interativo.