UNIVERSIDADE FERNANDO PESSOA

Porto (26/05/2024)

PROJETO SISTEMAS DISTRIBUIDOS

Manuel Gonçalves – 43087

Bruno Silva – 42824

Rui Pereira – 42619

Conteúdo

[Arquitetura adotada 3](#_Toc167658705)

[Estrutura do Padrão Observer 3](#_Toc167658706)

[Funcionamento do Padrão Observer 3](#_Toc167658707)

[Diagramas 4](#_Toc167658708)

[Implementação realizada 6](#_Toc167658709)

# Arquitetura adotada

O padrão de projeto Observer é um dos padrões comportamentais mais utilizados no desenvolvimento de software, especialmente em sistemas distribuídos. Ele permite que um objeto (chamado de "subject") notifique automaticamente um conjunto de outros objetos (chamados de "observers") sobre qualquer mudança no seu estado.

## Estrutura do Padrão Observer

1. Subject
   1. Irá conter um ArrayList com os Observers e um ArrayList com os Clients;
   2. Possui métodos para adicionar, remover e notificar Observers;
   3. Possui métodos para adicionar e remover Clients;
   4. Irá conter uma variável State que é utilizada para notificar os outros Observers que houve uma alteração no jogo;
2. Observer
   1. O objeto que deseja ser informado sobre mudanças no estado do Subject. Ele define uma interface com o método update, que é chamado pelo Subject quando há uma mudança de estado;
3. State
   1. O state é a class que contem o id e a mensagem. A junção do id e da mensagem atuais criam o estado para ser atualizado em cada observer;

## Funcionamento do Padrão Observer

1. Registo dos observers
   1. Inicialmente quando um user se regista no menu e cria uma sala ou junta-se a uma sala é criado um ObserverImpl para esse user que vai ficar registado no Arraylist de observers do SubjectRI da sala;
2. Mudança de estado
   1. O ClientManager é a class responsável por alterar os estados do jogo e notificar todos os Observers;
3. Atualização dos Observers
   1. Quando o ClientManager atualiza os estados a class SubjectRI vai notificar todos os observers que estiverem registados no ArrayList de Observers, enviando o novo estado atualizado;

# Diagramas

Uma imagem com texto, captura de ecrã, diagrama, Tipo de letra

Descrição gerada automaticamenteUML

Sequencia

Uma imagem com texto, diagrama, Paralelo, captura de ecrã

Descrição gerada automaticamente

Uma imagem com texto, diagrama, Paralelo, file

Descrição gerada automaticamente

# Implementação realizada

1. Requisito 1

Neste requisito apenas não foi implementado o JWT. A autenticação com registo e login foram implementadas a 100%;

1. Requisito 2

Este requisito ficou implementado a 100%;

1. Requisito 3
   1. RMI

Neste requisito a implementação com RMI ficou quase totalmente implementada, apenas faltou fazer o detach() dos jogadores que morrem no jogo;

* 1. RabbitMQ

Neste requisito a implementação com RabbitMQ ainda não foi feita, mas ira ser implementada;

1. Requisito 4

Este requisito não foi implementado nem irá ser implementado

1. Requisito 5

Este requisito foi implementado