Sistemas e Algoritmos Distribuídos: Threads

Trabalho 2

André Henrique Ludwig[1]
Vitor André Gehrke[2]

[1] andre.ludwig@edu.udesc.br
[2] vitor.gehrke@edu.udesc.br

UDESC
UNIVERSIDADE
DO ESTADO DE
SANTA CATARINA

Solução Proposta

- Java
- JavaFX

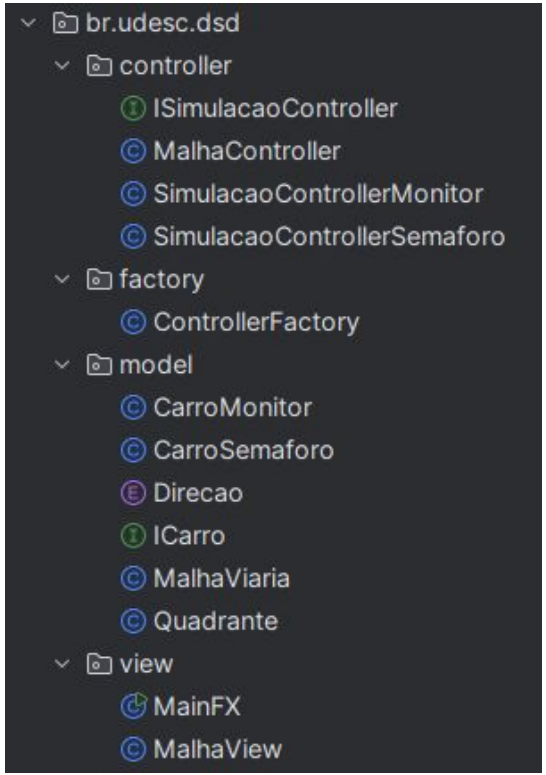
- Semáforos
- Monitores

- Seletor de malhas

UDESC
UNIVERSIDADE
DO ESTADO DE
SANTA CATARINA

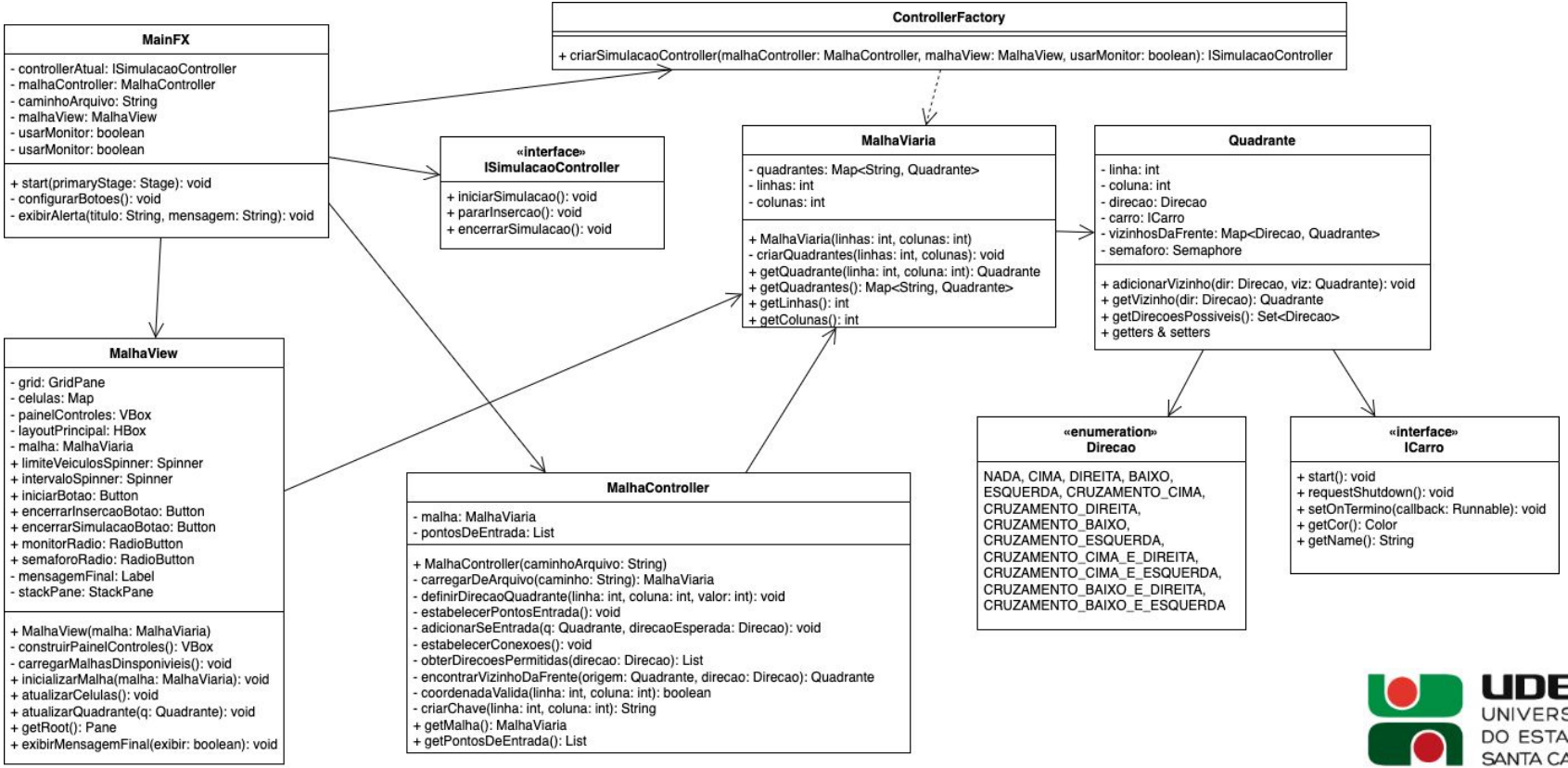
Estrutura da aplicação

- Estruturado nos pacotes controller, factory, model, view
- Padrões utilizados: Factory, Strategy
- Programação para Interfaces



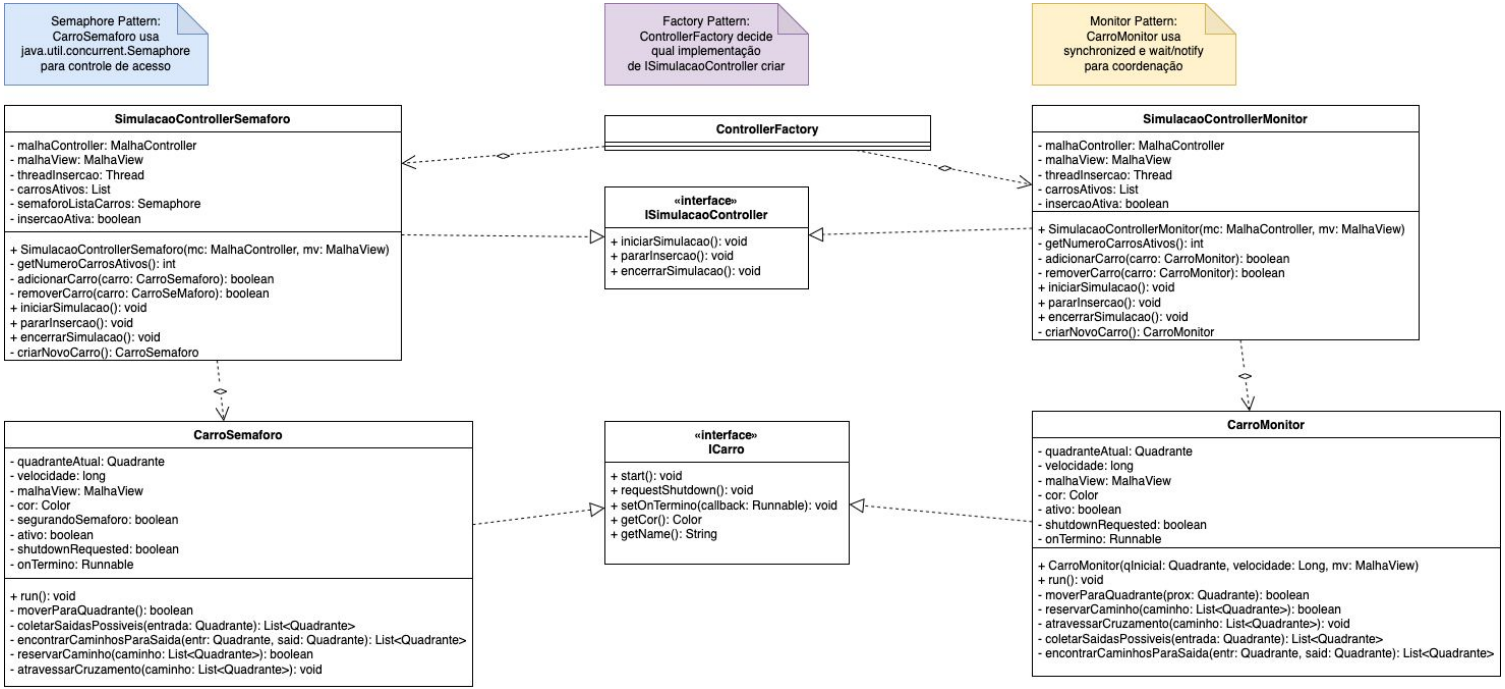
UDESC
UNIVERSIDADE
DO ESTADO DE
SANTA CATARINA

Diagrama de Classes - Visão Geral



UDESC
UNIVERSIDADE
DO ESTADO DE
SANTA CATARINA

Controller Factory e Implementações das Interfaces



UDESC
UNIVERSIDADE
DO ESTADO DE
SANTA CATARINA

Diferenças entre implementações

Característica	Implementação Semáforo	Implementação Monitor
Mecanismo	`Semaphore` por `Quadrante`	`synchronized` + `wait`/`notifyAll` por `Quadrante`
Estratégia nos cruzamentos	Reserva caminho **inteiro** (`tryAcquire`)	Move **passo a passo**; `wait()` se bloqueado
Aquisição	Tenta `tryAcquire`; se falhar, libera	Bloqueia com `synchronized`; `wait()` se ocupado (libera lock); Ordenação de locks
Prós	• **Alta Segurança** (Evitando deadlock interno) • Previsibilidade	• Potencial **maior vazão** • Fluxo "adaptativo" em cruzamentos
Contras	• **Menor vazão** • Risco de Livelocks	• Risco de **Deadlock** *teórico* (lock/`wait`) • "Reserva" • Menos previsível
Foco	**Segurança** e Prevenção	**Maior fluxo**

UDESC
UNIVERSIDADE
DO ESTADO DE
SANTA CATARINA

MainFX

```java
public class MainFX extends Application {  AndreHLudwig +1
    private ISimulacaoController controllerAtual;  11 usages
    private MalhaController malhaController;  3 usages
    private String caminhoArquivo;  2 usages
    private boolean usarMonitor = false;  3 usages
    private MalhaView malhaView;  11 usages

    @Override  AndreHLudwig +1
    public void start(Stage primaryStage) {
        malhaView = new MalhaView(null);
        configurarBotoes();

        Scene scene = new Scene(malhaView.getRoot());
        primaryStage.setScene(scene);
        primaryStage.setTitle("Simulador de Tráfego");
        primaryStage.show();
    }

    private void configurarBotoes() {...}


    private void exibirAlerta(String titulo, String mensagem) {...}

    public static void main(String[] args) {  AndreHLudwig
        launch(args);
    }
}
```

UDESC
UNIVERSIDADE
DO ESTADO DE
SANTA CATARINA

SimulacaoControllerSemaforo

```java
public class SimulacaoControllerSemaforo implements ISimulacaoController {  1 usage  Vitor André Gehrke +1

    private final MalhaController malhaController;  3 usages
    private final MalhaView malhaView;  9 usages

    private Thread threadInsercao;  7 usages
    private final List<CarroSemaforo> carrosAtivos = new ArrayList<>();  8 usages
    private final Semaphore semaforoListaCarros = new Semaphore( permits: 1,  fair: true);  12 usages
    private volatile boolean insercaoAtiva = false;  5 usages

    public SimulacaoControllerSemaforo(MalhaController malhaController, MalhaView malhaView) {...}

    private int getNumeroCarrosAtivos() {...}

    private boolean adicionarCarro(CarroSemaforo carro) {...}

    private boolean removerCarro(CarroSemaforo carro) {...}

    private List<CarroSemaforo> getCarrosAtivos() {...}

    private boolean carrosAtivosVazio() {...}

    private void limparCarrosAtivos() {...}

    public void iniciarSimulacao() {...}

    public void pararInsercao() {...}

    public void encerrarSimulacao() {...}

    private CarroSemaforo criarNovoCarro() {...}
}
```

UDESC
UNIVERSIDADE
DO ESTADO DE
SANTA CATARINA

SimulacaoControllerMonitor

```java
public class SimulacaoControllerMonitor implements ISimulacaoController {  1 usage  AndreHLudwig

    private final MalhaController malhaController;  3 usages
    private final MalhaView malhaView;  9 usages

    private Thread threadInsercao;  7 usages
    private final List<CarroMonitor> carrosAtivos = new ArrayList<>();  8 usages
    private volatile boolean insercaoAtiva = false;  5 usages

    public SimulacaoControllerMonitor(MalhaController malhaController, MalhaView malhaView) {...}

    private synchronized int getNumeroCarrosAtivos() { return carrosAtivos.size(); }

    private synchronized boolean adicionarCarro(CarroMonitor carro) {...}

    private synchronized boolean removerCarro(CarroMonitor carro) {...}

    private synchronized List<CarroMonitor> getCarrosAtivos() { return new ArrayList<>(carrosAtivos); }

    private synchronized boolean carrosAtivosVazio() { return carrosAtivos.isEmpty(); }

    private synchronized void limparCarrosAtivos() { carrosAtivos.clear(); }

    public void iniciarSimulacao() {...}

    public void pararInsercao() {...}

    public void encerrarSimulacao() {...}

    private CarroMonitor criarNovoCarro() {...}
}
```

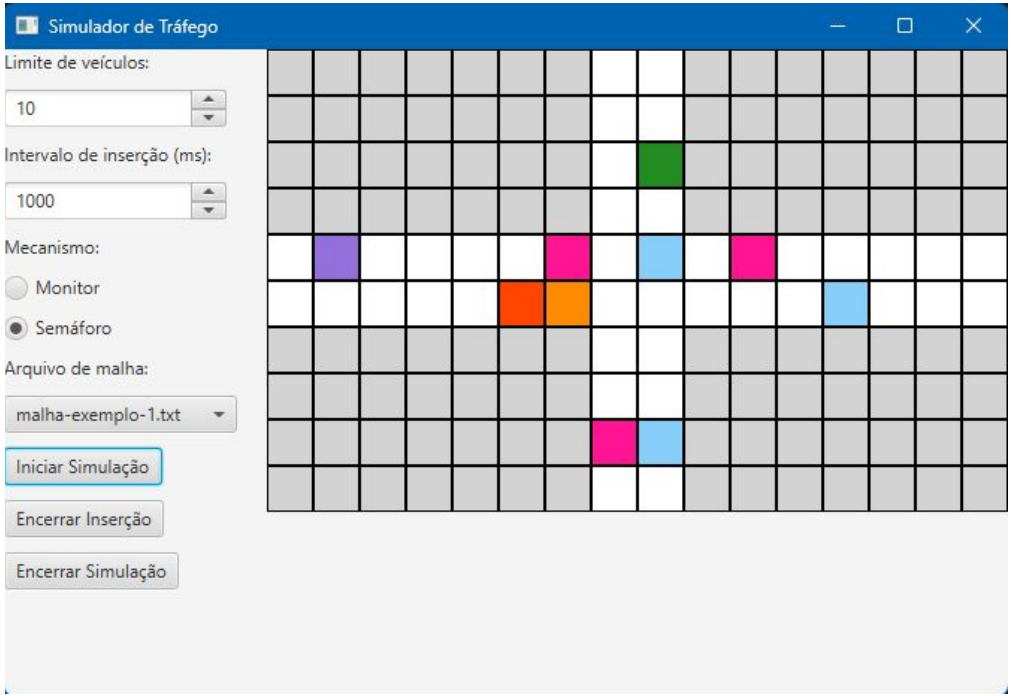
UDESC
UNIVERSIDADE
DO ESTADO DE
SANTA CATARINA

ControllerFactory

```java
public class ControllerFactory {  2 usages  Vitor André Gehrke +1
    public static ISimulacaoController criarSimulacaoController(  1 usage  Vitor André Gehrke +1
            MalhaController malhaController,
            MalhaView malhaView,
            boolean usarMonitor) {

        if (usarMonitor) {
            return new SimulacaoControllerMonitor(malhaController, malhaView);
        } else {
            return new SimulacaoControllerSemaforo(malhaController, malhaView);
        }
    }
}
```

Interface JavaFX



Dificuldades encontradas?
Aprendizados?
Melhorias futuras?

UDESC
UNIVERSIDADE
DO ESTADO DE
SANTA CATARINA

Obrigado!

https://github.com/gehrkev/T2_65DSD

UDESC
UNIVERSIDADE
DO ESTADO DE
SANTA CATARINA