Unidade Curricular: Aplicação Bases de Dados

Curso: Engenharia informática

Ano Letivo: 2025/2026

Relatório Smart Traffic Flow Simulation

Trabalho realizado por:

Diogo Forte nº 30014529

Gustavo Caetano nº 30015026

Martim Mello nº30015349

Índice

1.Introdução

Neste projeto tínhamos como objetivo aplicar os conceitos aprendidos na cadeira de Programação Orientada a Objetos e realizar um projeto com o objetivo de criar um simulador de tráfego inteligente feito em java, utilizamos a biblioteca JavaFX para conseguir ver em tempo real o desenvolvimento do nosso projeto.

2.Arquitetura do Sistema

O projeto foi desenvolvido seguindo o modelo MVC, garantindo uma melhor organização no projeto e separando as partes correspondentes ao Model, a View e ao Controller

2.1.Model

No Model temos o ficheiro Vehicle.java que é a classe que gere o comportamento individual de cada carro utilizando um sistema de deteções de colisões baseado em zonas de segurança e verifica a distancia para com outros carros ou semáforos, o movimento imposto por vetores de direção e dependendo do estado do semáforo o veiculo ajusta a sua velocidade. O Intersection.java faz com que o sistema trate um cruzamento como uma unidade lógica ajudando assim a aplicação

2.2.Controller

No Controller encontramos o TrafficStrategy.java onde foi implementada uma interface de estratégia que define o comportamento dos semáforos, isto permite com que o simulador consiga alternar entre uma estratégia fixa e uma adaptativa.

2.3.View

Na View temos o ficheiro SimulationCanvas.java que é responsável por fazer a simulação do projeto no ecrã, algumas das principais funções são a Renderização Gráfica, a Representação Visual, a Animação Fluida e as Transformações Visuais. E temos o ficheiro TrafficApp.java responsável por permitir ao utilizador: iniciar, pausar ou reiniciar a simulação, ajustar a velocidade em tempo real e alterar entre diferentes mapas e estratégias.

Patterns Usados

Os patterns utilizados neste projeto foram Strategy Pattern, State Pattern e o Model-View-Controller.

Strategy Pattern

O Strategy Pattern esta na interface TrafficStrategy.java, serve para permitir que uma interseção alterne entre diferentes logicas, sem alterar o código da interseção. Este pattern confere flexibilidade ao sistema e permite alterar o semáforo em tempo de execução.

State Pattern

O State Pattern serve para as transições de cor do semáforo, sabendo cada estado qual o próximo e quanto tempo deve durar cada um.

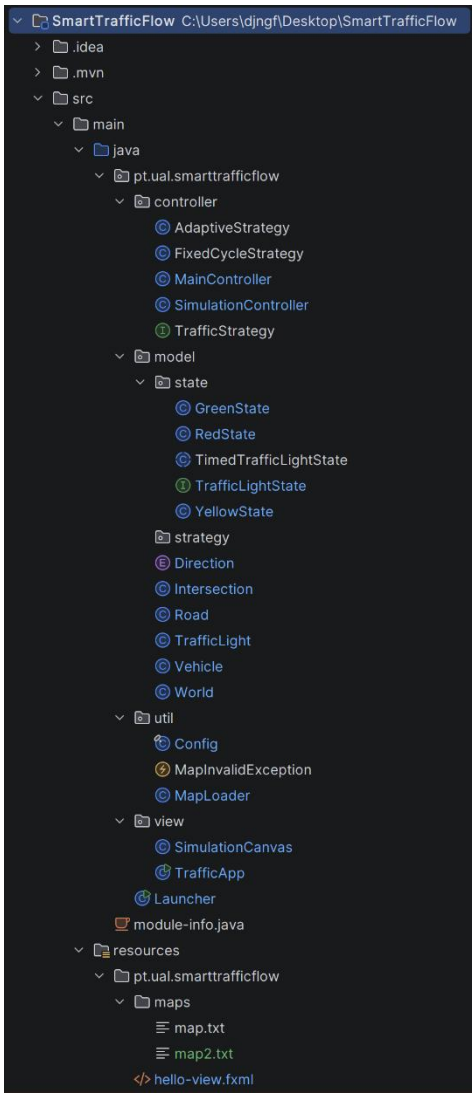
Model-View-Controller

O Model-View-Controller é um pattern de arquitetura simples servindo para organizar o projeto, como a foi falado no Arquitetura do Sistema. No Model estão as classes Vehicle, Intersection e World que servem para guardar os dados e a logica de movimento, na View estão o SimulationCanvas.java e o TrafficApp.java responsáveis pelo desenho da interface e no Controller estão o SimulationController.java e o TrafficStrategy.java que são a ponte entre os dados e o tempo.

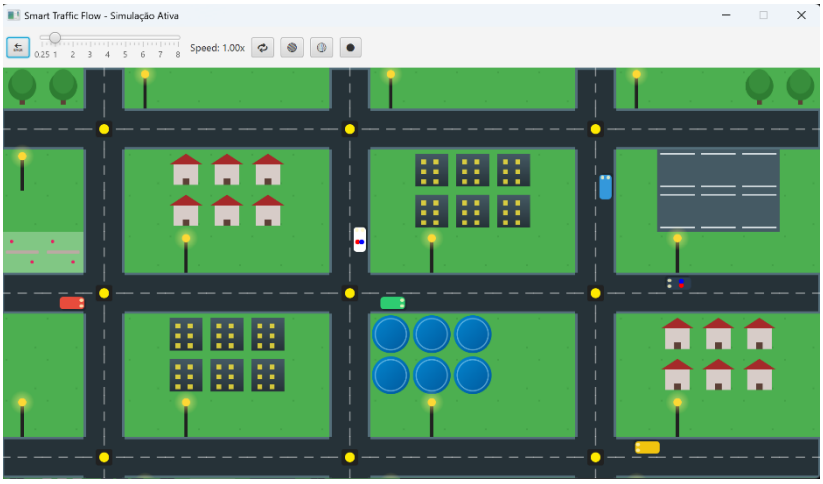
Estratégias de Testes

Os testes do projeto foram realizados através de uma abordagem sistemática, combinando a verificação da lógica de negócio com testes de integração dos padrões de desenho e validação funcional da interface gráfica. Realizamos testes demovimentos e colisões e de integração de padrões.

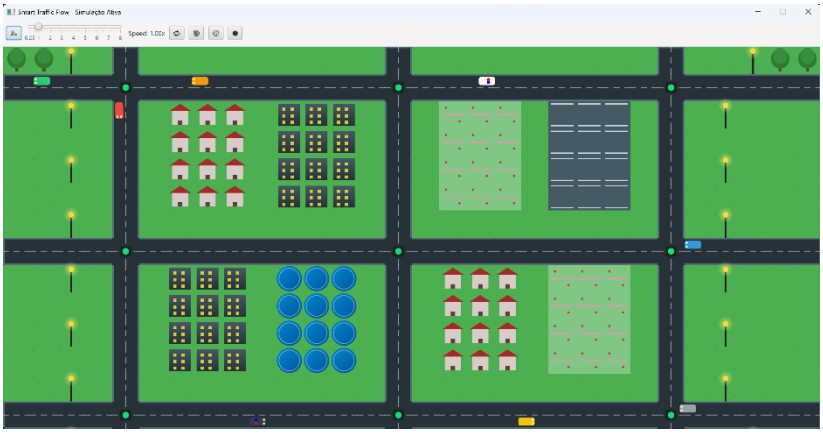
Resultados Principais



MODELO MVC



MAPA 1



MAPA 2

Conclusão

O desenvolvimento deste projeto permitiu-nos aplicar com sucesso vários conceitos fundamentais de Programação Orientada a Objetos, o projeto atingiu todos os objetivos propostos pelo professor e mais alguns. O MVC conseguiu nos ajudar a deixar o trabalho mais organizado e os patterns foram cruciais para a escalabilidade do projeto. Foi um projeto trabalhoso, mas com um resultado muito gratificante