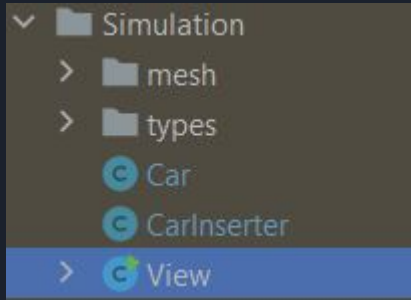




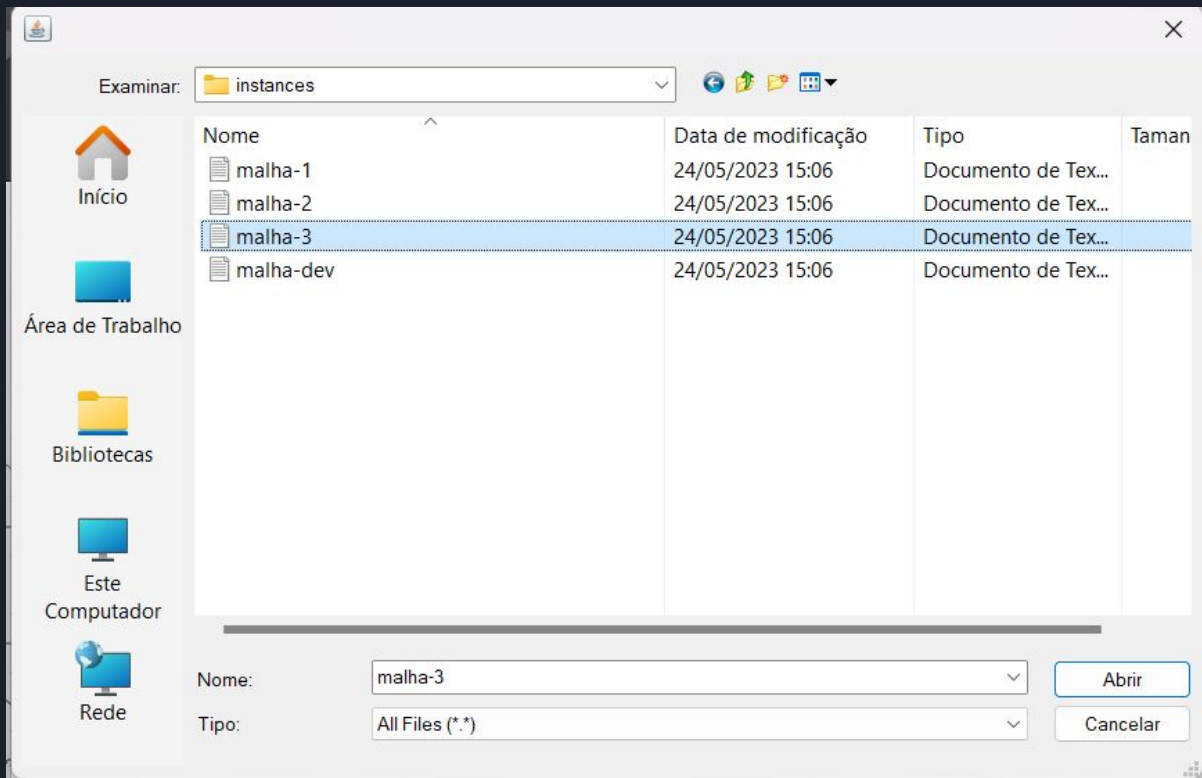
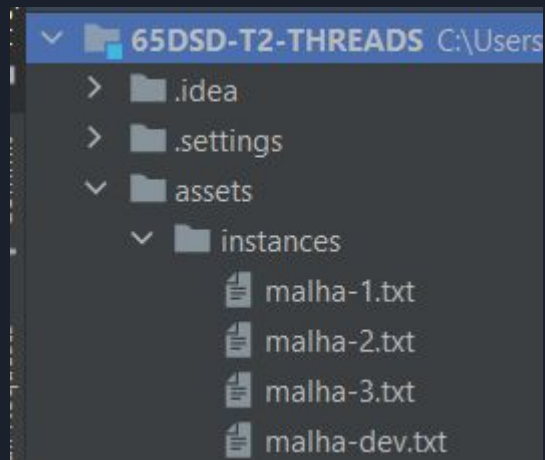
# T2 - THREADS

Acadêmicos: Augusto Rustick e Denis Zickuhr

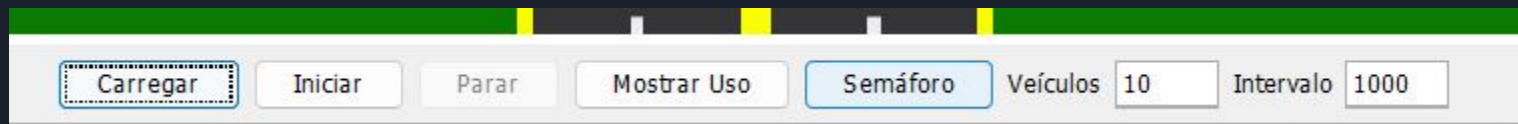
# Executar o projeto



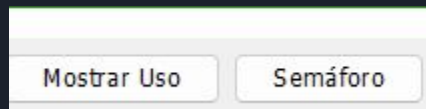
# Escolha de malhas



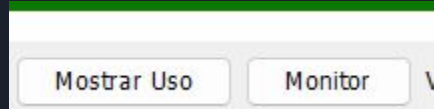
# Componentes da tela



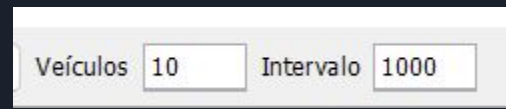
A horizontal control bar with a green progress indicator at the top. Below it, a series of buttons: 'Carregar' (highlighted with a dashed border), 'Iniciar', 'Parar', 'Mostrar Uso', 'Semáforo' (highlighted with a solid blue border), 'Veículos' (with a text input field containing '10'), and 'Intervalo' (with a text input field containing '1000').



A panel containing two buttons: 'Mostrar Uso' and 'Semáforo'.

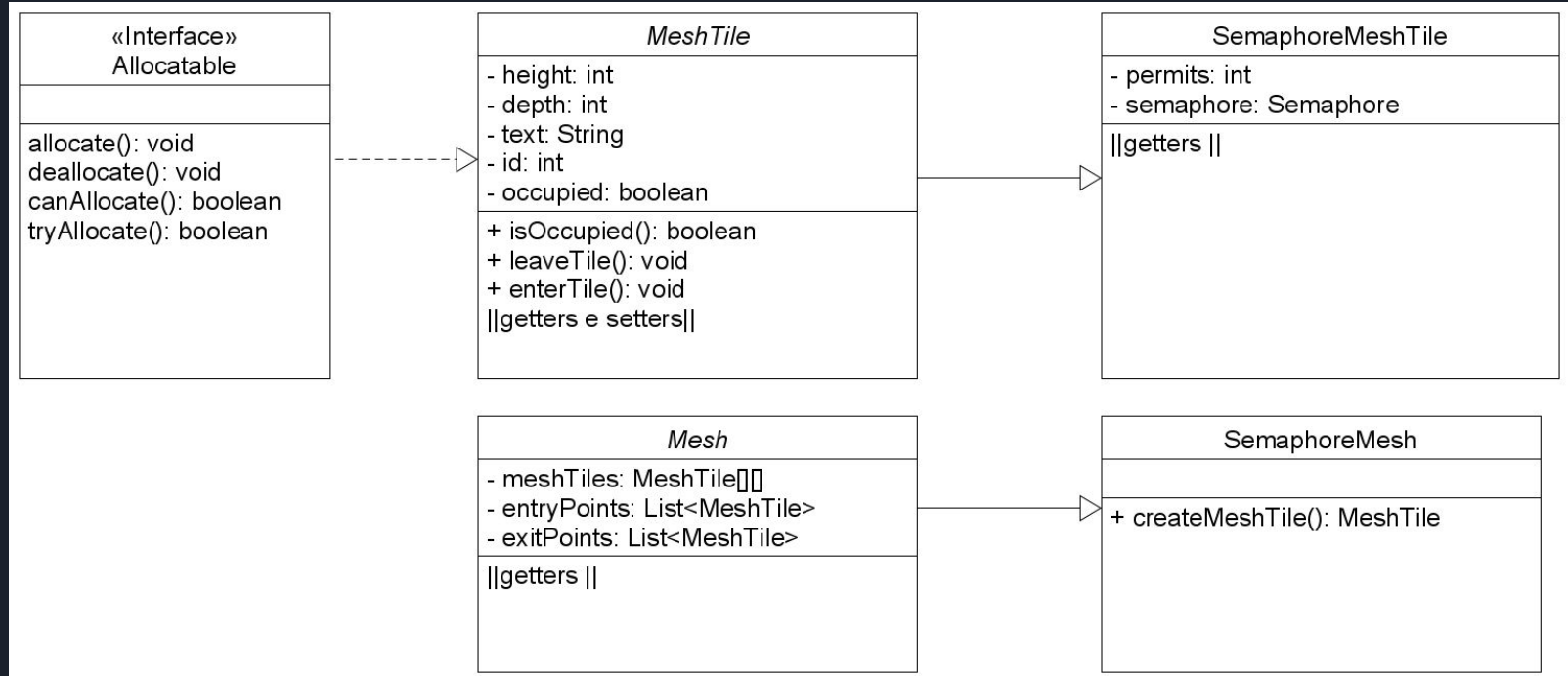


A panel containing two buttons: 'Mostrar Uso' and 'Monitor'.

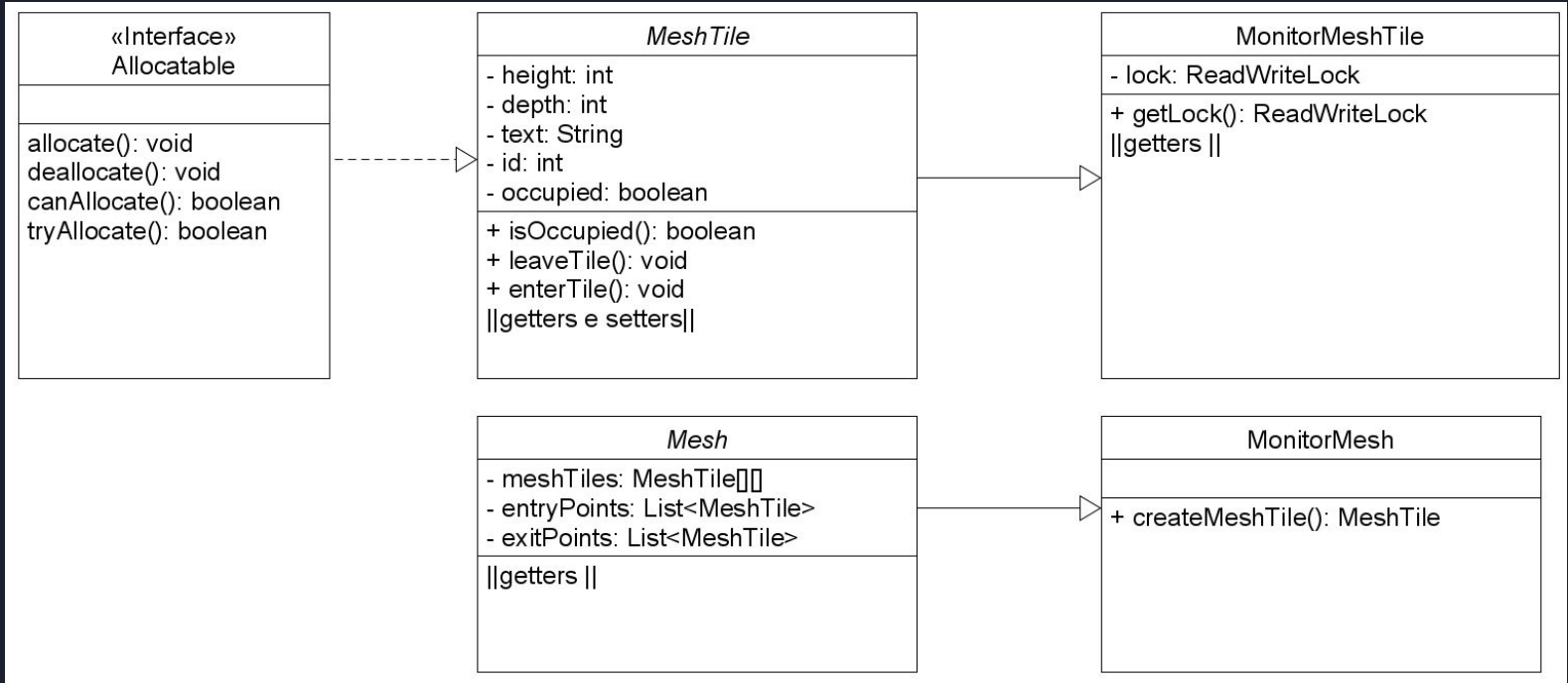


A panel containing two input fields: 'Veículos' with the value '10' and 'Intervalo' with the value '1000'.

# Diagrama de semáforos



# Diagrama de monitores





# Problemas enfrentados

- Organização do projeto
- Atualização dos frames (Desenhar tela a cada movimento)
- Manipulação da sobre camada, para a exibição de uso do semáforo, na implementação com monitores
- Alocação/Devolução dos 'Tiles' para os Cruzamentos



# Soluções encontradas

- Adoção de Classes abstratas para a Malha e os pontos da malha, além da adoção de uma interface Allocatable, que tem implementação forçada na classe abstrata de *MeshTile*.
- Atualização periódica da tela, solução funcional, porém má performática.
- Adoção de um monitor duplo, que suporta leitura de vários consumidores (ReadWriteLock)
- Uso de passos simulados, utilizando um laço de repetição, que só se encerra ao percorrer todo o passo simulado necessário, e em caso de falha, devolve todo o passo simulado, devolvendo os Locks, ou Semáforos em caso de concorrência.





Agora, vamos ao documento e ao fonte...