
Relatório Técnico – Avaliação do Projeto "SimuladorDeLixo"
Curso: Engenharia de Software
Período: 3º Período
Disciplina: Estrutura de Dados
Tipo de Atividade: Análise Simulada de Projeto Desenvolvido por Luis Guilherme de Morais Abreu e João Matheus Brito de Araújo Sousa

---

1. Objetivo do Sistema

O projeto **"SimuladorDeLixo"** é uma aplicação desenvolvida em Java com o objetivo de simular o processo de coleta e transporte de resíduos sólidos urbanos, utilizando caminhões de pequeno e grande porte. A simulação contempla o fluxo completo, desde as zonas de coleta até as áreas de transferência e o destino final dos resíduos, representando de forma simplificada como funciona o sistema de gerenciamento de lixo em ambientes urbanos.

---

2. Estrutura do Código

O projeto está bem organizado, com separação clara de responsabilidades em diferentes pacotes, o que facilita a manutenção e o entendimento:

- **EstruturasDeDados:** Contém implementações próprias de estruturas como listas e filas, fundamentais para a lógica de simulação.
- **Mecanismo:** Gerencia o funcionamento da simulação, coordenando os eventos e operações por meio da classe principal Simulador.java.
- **Modelo:** Define as entidades principais do sistema, como Zona, CaminhaoPequeno, CaminhaoGrande, EstacaoTransferencia, entre outras.
- **Utilitarios:** Inclui classes auxiliares, como App.java(inicialização da simulação) e Relatorio.java(provável responsável pela exibição dos resultados).

---

3. Funcionamento Geral

A simulação segue um modelo de eventos discretos, em que cada evento representa uma ação no tempo. Os caminhões pequenos atuam coletando o lixo nas zonas designadas e transportam até as estações de transferência. A partir daí, os caminhões grandes assumem a função, levando os resíduos até o destino final.

O controle dos eventos e do tempo é feito pela classe Simulador.java, que mantém uma fila de eventos organizada cronologicamente. À medida que o tempo avança, os eventos são processados e atualizam o estado do sistema (como disponibilidade dos caminhões e capacidade das zonas).

Esse fluxo garante que o comportamento da simulação siga uma ordem lógica e previsível, refletindo adequadamente situações comuns na logística de coleta urbana.

---

4. Avaliação Técnica

Pontos positivos:

- Boa organização modular.
- Classes bem nomeadas, facilitando a leitura do código.
- Aplicação correta de conceitos de simulação, como o uso de filas e eventos.

5. Pontos que podem ser melhorados:

- A aplicação não possui interface gráfica, o que pode dificultar a compreensão dos resultados por parte de usuários não técnicos.
- Há pouca documentação (comentários) no código, o que compromete a compreensão por outros desenvolvedores.
- A exibição dos resultados e relatórios poderia ser mais clara e visualmente acessível.

Esse fluxo garante que o comportamento da simulação siga uma ordem lógica e previsível, refletindo adequadamente situações comuns na logística de coleta urbana.

---

6. Heurística e Lógica de Funcionamento

A heurística do código segue um modelo direto e orientado a estados, com base no seguinte ciclo:

1. **Inicialização de entidades**: zonas, caminhões e estações são carregados e conectados.
2. **Simulação em tempo real**: um Timer Java é usado para avançar o tempo a cada segundo, simulando minutos no sistema.
3. **Eventos acionados pelo tempo**: a cada “tick”, verifica-se o estado das zonas, caminhões e estações para decidir se uma ação será tomada (coleta, transporte, descarregamento).
4. **Atualização e persistência**: o estado é atualizado a cada ciclo e os resultados são registrados em um relatório.

Não há uma fila de prioridade explícita no estilo clássico de eventos (como em simuladores baseados em heap). Em vez disso, a simulação avança minuto a minuto, verificando as condições para execução das ações — um modelo mais simples, porém funcional.

---

7. Papel das Entidades

Entidades:

. Zonas**: geram lixo a cada minuto, com capacidade de acúmulo.**

. **Caminhões Pequenos**: coletam lixo das zonas e entregam em estações.

. **Estações de Transferência**: armazenam lixo temporariamente e são ponto de partida para os caminhões grandes.

. **Caminhões Grandes**: levam o lixo das estações até o destino final.

---

8. Simulação Orientada a Eventos – Adaptação no Código

Apesar de não usar uma FilaPrioridade<Evento> típica, o projeto aplica **eventos temporais discretos** como mecanismo de controle de simulação:

- O **tempo avança de forma contínua (1 minuto por tick)** via TimerTask.
- A cada minuto, eventos implícitos são avaliados:
    - Se uma zona acumulou lixo suficiente.
    - Se há caminhão pequeno disponível.
    - Se uma estação está livre.
    - Se o caminhão grande pode levar o lixo ao aterro.

Ou seja, os eventos são disparados **por condições de estado**, não por uma fila explícita. Isso reduz a complexidade, embora também limite o controle fino sobre a ordem e dependência entre eventos.

---

9. Possibilidade de Melhoria – Aleatorização da Ordem de Saída

Uma sugestão funcional seria implementar a **aleatorização da ordem de saída dos caminhões pequenos** para as zonas de coleta. Isso pode ser feito utilizando o método Collections.shuffle()em uma lista de caminhões, antes da simulação iniciar.

Essa alteração proporcionaria uma simulação mais próxima de cenários reais, em que a ordem de operação dos caminhões não é necessariamente fixa, além de permitir uma maior variedade nos testes.

---

10. Análise das Estruturas de Dados

O sistema utiliza **estruturas customizadas**, como:

- Lista<T>: implementação própria de lista ligada genérica.
- Fila<T>: fila básica usada nas zonas para representar acúmulo de lixo.

- Essas estruturas foram desenvolvidas sem uso de ArrayList, conforme exigência acadêmica, e são adequadas para o propósito do projeto.

---

11. Conclusão

O projeto **"SimuladorDeLixo"** representa uma aplicação sólida dos princípios de orientação a objetos e de simulação discreta. A estrutura do código está organizada e os papéis das classes estão bem definidos.

Apesar de haver espaço para melhorias, especialmente na interface e na documentação, o funcionamento central do sistema demonstra compreensão técnica e domínio dos conteúdos da disciplina. Trata-se de um trabalho bem conduzido e adequado ao nível atual do curso.