Trabalho 1 - INE5645

Bernardo De Marco Gonçalves - 22102557

Design da aplicação

A aplicação é composta por N sensores (valor parametrizável via *input*). Cada sensor é uma *thread* que executa um *loop* inifinito, produzindo valores aleatórios e os inserindo no *buffer* do produtor/consumidor.

O *core* da aplicação concentra-se na central de controle (*orchestrator*). Esse componente é uma única *thread* que gerencia os atuadores do veículo, através de um *thread pool* de `4` *threads* e uma tabela de controle. Além disso, o *orchestrator* é o o responsável por consumir os dados produzidos pelos sensores do veículo.

Sempre que o *orchestrator* consumir um valor do *buffer*, ele envia o valor juntamente com uma *task* a ser executada para uma *thread* do *thread pool*. Cada *task* é responsável por determinar o atuador e o nível de atividade, atualizar o registro do atuador e gerar um *log* no terminal.

A atualização e a geração dos *logs* são executados em paralelo (*fork-join*). Caso ocorra algum erro em alguma dessas subtarefas, a *task* é responsável por gerar uma saída informando o erro ocorrido.

Para atualização do atuador, o seu nível de atividade na tabela de atuadores é atualizado, e permanece inalterado por dois ou três segundos. Para não definir a tabela inteira como uma única zona de exclusão mútua, ela foi dividida em seções críticas com uma granularidade pré-determinada de `20`. Ou seja, quando um atuador tiver seu nível de atividade atualizado apenas a sua seção será travada, e não a tabela inteira.

Concomitantemente à atualização da tabela, o *log* é gerado. Sempre que um *log* é impresso, ele será mantido no console por um segundo. Por fim, a *task* do *thread pool*, ao sincronizar o *fork-join*, checa se houve algum erro em alguma tarefa (`20%` de chance de erro em cada uma). Caso algum erro tenha ocorrido, um *log* de erro é gerado e impresso no terminal.

Tecnologias adotadas

- Linguagem de programação C
- POSIX *library*
- C-Thread-Pool

A biblioteca C-Thread-Pool foi utilizada para o uso de *thread pools*. Ela possui uma API clara e de fácil utilização. Além disso, seu código-fonte foi verificado e foi certificado que ela utiliza uma *queue* como *buffer* de *tasks* do *pool*. Com isso, foi possível utilizá-la para cumprir requisitos da aplicação.

Instruções de *setup*

Foi utilizado o `Makefile` para facilitar a compilação da aplicação. O arquivo compilado é o arquivo `app` localizado na pasta `build`.

- Compilar:

  ```
  make compile
  ```

- Rodar:

  ```
  make run
  ```

- Limpar:

  ```
  make clean
  ```