NOVA SCHOOL OF SCIENCE & TECHNOLOGY

Concurrent Programming: Languages and Techniques

2024/2025

Erlang Mini-Project – Task 1

Authors:

70481, Vasco João
70140, Diogo Almeida

Group: 70140_70481

Professor:
Carla Ferreira

Month 10, 2024

1. General Requirements

This system guarantees:

- Concurrency

All the components are run on separate threads, communicating with each other through their PIDs.

- Deadlock-Freedom

Tasks never deadlock as they always have a timeout if they don't get the package or message they are waiting for. Which "resets" the function not letting it reach a deadlock.

- Progress

Packages only go one way. Which means they always end up on the truck. Even in the case of task 2 and 3 where u can't know what package will fill the truck till the truck finds one it can't fit. The package is never lost because even if the truck leaves, the next truck just loads it immediately before even receiving another package.

- Message Passing

As it was mentioned before, the threads communicate using PIDs. Which allows each sender to know who the receiver is beforehand, and the receiver to know who the sender is after receiving the message.

2. Task 1

This task was essentially the implementation of the general requirements. For that purpose, we used the following functions:

- start/0

Spawns threads for feeder/2, conveyor/2 and truck/1 3 times.

- feeder/2

Starts the continuous package generation the feeds the conveyor by calling its loop.

- feederLoop/3

feeder/2 loop, continuously creates packages and sends them to the conveyor via message.

- conveyor/2

Starts the conveyor belt loop.

- beltLoop/2

conveyor/2 loop, each time it receives a package immediately sends it to the truck.

- truck/1

Starts the process of loading packages into the truck by setting a capacity and calling truckLoop/3.

- truckLoop/3

Process of loading packages into the truck, once full it reset the capacity and empties its load as to simulate an arrival of a new truck to replace it.