POLITÉCNICO
DE PORTALEGRE

Escola Superior
de Tecnologia,
Gestão e Design

Department of Technologies

Operating Systems

IPC Mechanisms in Linux - 2024/2025

Eng. Informática

Delivery date: 18/01/2025 – **Teacher** : Valentim Realinho

Restaurant Ordering System Simulation

Objectives

The main objective of this work is to develop practical skills in the use of IPC mechanisms in Linux to create systems that simulate complex interactions between *threads* , namely:

1. Implement a system that uses message queues, shared memory and semaphores.
2. Coordinate multiple *threads* to simulate real-world scenarios.
3. Demonstrate an understanding of synchronization and efficient communication between *threads* .

Description

You must implement a program that simulates the operation of a restaurant, which must have the following components:

1. **Clients (*threads*):** Request dishes from the menu by sending orders to the central server.
2. **Cooks (*threads*):** Prepare the dishes requested by customers.
3. **Central Server (*threads*):** Manages the orders received, passes them to the cooks and delivers the ready dishes to the customers.

System Details

- The client sends requests to the central server using **message queues** .
- The server stores orders in a **shared memory** so that the cooks can access them.
- You should use **semaphores** to manage access to shared memory, ensuring that only one cook handles a request at a time (you can use the semaphore code provided in the classes - `semaphore.c` and `semaphore.h`).
- Each customer should receive feedback (via appropriate `printf`) indicating the status of the order (requested, in preparation, ready).
- The simulation must include delays (`sleep`) to represent the preparation time of the dishes.

Requirements

- The restaurant menu must contain at least 5 dishes, each with a different preparation time.
- There must be at least 2 threads of cooks and the arrival of customers must be simulated with a variable interval between arrivals. The restaurant never closes (customers can always be arriving) and the restaurant always has the number of cooks working.
- The central server must be robust, handling multiple customer orders and distributing tasks among the cooks efficiently.
- You should use detailed logs of operations, including orders received, dishes being prepared, dishes delivered, and the status of the order queue.

Evaluation Criteria

1. **Functionality (40%):** The system adequately responds to the described requirements and functions correctly.
2. **IPC Usage (20%):** Efficient and appropriate implementation of message queues, shared memory and semaphores.
3. **Code Structure (5%):** Organized code, with good naming of variables and functions, and adequate use of comments.
4. **Documentation (5%):** Clear instructions for compiling and running the program, plus sample output in a README file.
5. **Presentation and discussion of the work (20%)** :

Delivery

The work must be submitted to PAE in a ZIP file containing:

- Program source code.
- Explanatory README file with usage instructions and execution examples.
- Logs generated during the simulation.