Ecole Publique d'ingénieures et d'ingénieurs en 3 ans

Rapport

EPIDEMIE 2024

le 12 janvier 2025,
version 1

BRIDA Mohamed Taha

LY Johann

TOUJANI Mohamed

ZOUGGARI Taha

www.ensicaen.fr

CONTEXT AND GOALS

This project models the spread of an epidemic in a 7×7 city grid. Different roles (citizens, firefighters, doctors, journalists) interact to either propagate or contain the contamination. We use low-level mechanisms (processes, shared memory, signals, etc.) to simulate a dynamic environment.

We provide two approaches:

1.1. Single-Thread

A single thread handles all entities through a main simulation loop.

1.2. Multi-Thread

Several threads manage entities concurrently, improving responsiveness but requiring careful synchronization.

1.3. Our objectives

- Manage infection spread based on rules (e.g., moving citizens, contamination levels, capacity of buildings).
- Implement specialized behavior for firefighters (decontamination, burning corpses), doctors (healing, administering treatment), and journalists (reporting to a press agency).

ARCHITECTURE OVERVIEW

2.1. Data Structure:

We use a shared structure (in either global memory or shared memory) holding:

- Grid information (building types, contamination levels, etc.).
- Citizen states (roles, positions, contamination rates, etc.).
- Statistics for logging and real-time updates.

2.2. Communication:

- **Memory Sharing**: Central for accessing/updating the simulation state.
- **Signaling**: The timer notifies each "round" of simulation; a stop signal ends the process.
- **Message Queue**: Journalists send updates to the press agency.

2.3. Mono-Thread vs. Multi-Thread:

- **Mono-Thread:** Simpler to debug. All actions happen in a single loop, with function calls for each agent's behavior.
- **Multi-Thread:** Each role or entity runs in parallel. This introduces concurrency (and possible race conditions) but can better reflect real-time behaviors.

MONOTHREAD SIMULATION DEMO

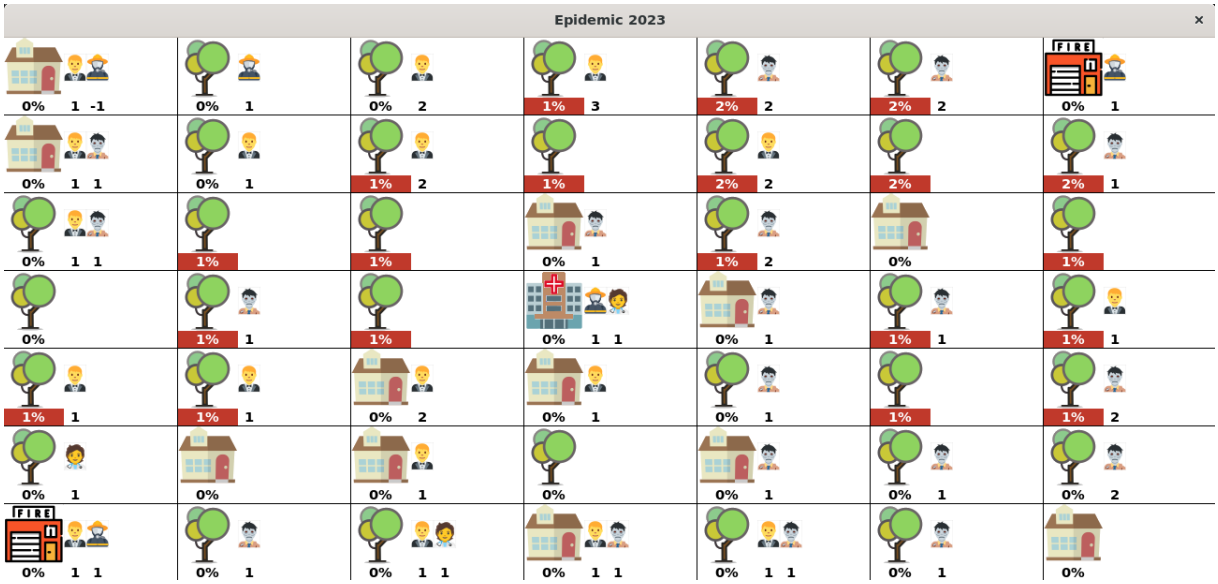

Figure 1 Simulation Board

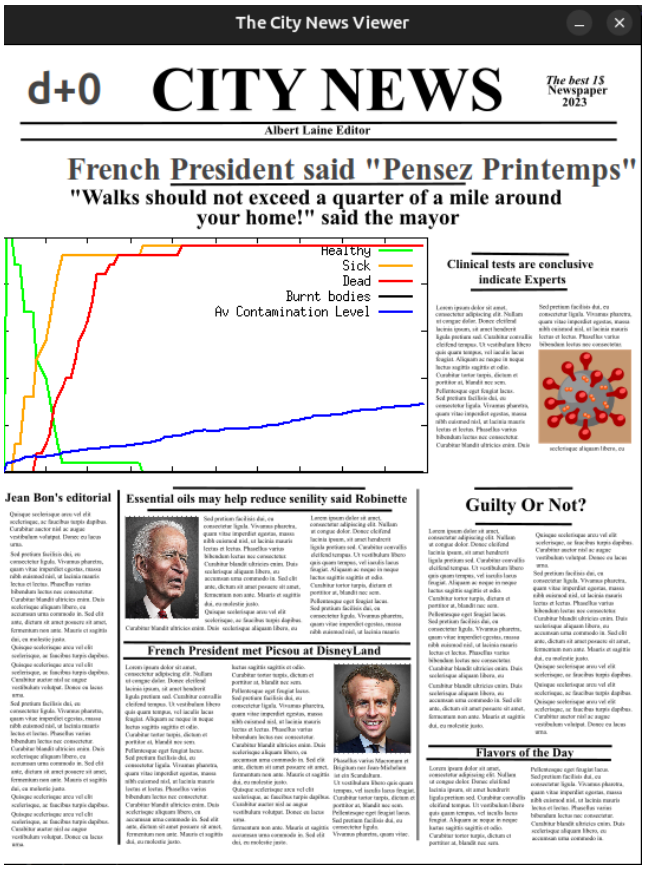

Figure 2 News Board

IMPLEMENTATION HIGHLIGHTS

- Initialization: Random placement of buildings and citizens, some terrains pre-contaminated.
- Contamination Spread: Citizens raise or lower contamination by moving around. Terrains can contaminate adjacent terrains under certain probabilities.
- Special Roles :
    - Firefighters :
        - Decontaminate citizens and places.
        - Burn corpses to prevent further contamination.
    - Doctors :
        - Use care pouches outside hospitals.
        - In hospitals, can heal without using pouches and greatly reduce mortality risk.
    - Journalists :
        - Send status reports (contamination levels, casualty counts, etc.) to a press agency process.

CHALLENGES & SOLUTIONS

4.1. Concurrent Access:

- **Solution**: In the multi-thread approach, we considered mutexes or semaphores around critical sections, but we didn't manage to implement it correctly. The mono-thread version avoids concurrency issues but yields slower event processing.

4.2. Data Consistency:

4.3. Randomization:

- **Solution**: We seeded rand() for reproducibility in debugging but allowed variability for final runs.

CONCLUSION

This project demonstrates core system programming concepts, process management, threading, shared memory, and inter-process communication via an epidemic simulation. The single-thread version is more straightforward to maintain, whereas the multi-thread approach more accurately models concurrent behavior but demands robust synchronization. Although, we didn't manage to implement it fully and correctly.

ENSI CAEN

ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

Ecole Publique d'ingénieures et d'ingénieurs en 3 ans

6 boulevard Maréchal Juin, CS 45053
14050 CAEN cedex 04