**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**KHOA CÔNG NGHỆ THÔNG TIN**

A picture containing logo

Description automatically generated**🙡🕮🙣**

**BÁO CÁO ĐỒ ÁN**

**PHÁT TRIỂN HỆ THỐNG QUẢN LÝ TẬP TIN ĐƠN GIẢN (YFS)**

MÔN : CHUYÊN ĐỀ HỆ THỐNG PHÂN TÁN – 20MMT

GVLT: Trần Trung Dũng

GVTG: Huỳnh Thụy Bảo Trân

Thành phố Hồ Chí Minh, tháng 2 năm 2024

1. **THÔNG TIN CÁ NHÂN :**

|  |  |  |
| --- | --- | --- |
| **MSSV** | **Họ Và Tên** | **Email** |
| 20127090 | Lê Thanh Tú | thanhtuhr01@gmail.com |
| 20127607 | Phạm Việt Quang | pvquang0703@gmail.com |

1. **THÔNG TIN VỀ ĐỒ ÁN:**
   * Giới thiệu tổng quan về YFS là một ứng dụng hoặc hệ thống được thiết kế để giúp người dùng tổ chức, lưu trữ, và quản lý các tập tin và thư mục trên máy tính hoặc hệ thống thông tin. Mục tiêu của YFS là cung cấp một phương tiện dễ sử dụng và hiệu quả để quản lý các tài liệu và dữ liệu của người dùng.
   * SES multicast có thể được hiểu như một phương pháp gửi thông điệp từ một nguồn đến nhiều người nhận trong một mạng hợp tác, trong đó SES viết tắt của "Simple Email Service" - một dịch vụ của Amazon Web Services (AWS) cung cấp việc gửi email một cách linh hoạt và hiệu quả.
   * Sử dụng thuật toán SES multicast để quản ý thứ tự thực hiện cho các lệnh đọc/ghi.
   * Ngôn ngữ chính là: python.
   * Sử dụng docker như những máy riêng biệt.
   * Link video demo:
   * Link github: https://github.com/tu11aa/You-Files-System-YFS-.git
2. **MÔ TẢ HOẠT ĐỘNG CỦA PROJECT:**
3. ***Cấu trúc của projcect:***

A diagram of a computer

Description automatically generated

- message.py: để đóng gói message khi gửi qua đường mạng.

+ class MessageType: sẽ chứa kiểu của message như là broadcast, mount, read, write, start\_write, end\_ write.

+ class Message: khởi tạo 1 message bao gồm

- yfs\_logger: để ghi log ra giúp cho việc kiểm tra và cách mà hệ thống vận hành.

+ class YFSLogger: để in log ra file log.

- yfs\_server: là file chính giúp cho mỗi bản thân process có thể chạy và hoạt động theo thuật toán.

+ class YFS: chứa các hàm giúp cho process có thể chạy đúng thuận toán SES.

- yfs\_user\_interface: một file giúp người dùng có thể tương tác được trong 1 process hoặc tương tác từ process này sang process kia.

- test\_yfs và testcases: chứa các test case và các hàm run testcase (18 lệnh đọc và 12 lệnh ghi).

- Logs:

A screen shot of a computer

Description automatically generated

1. ***Cách hoạt động của project:***

***A screenshot of a computer

Description automatically generated***

- Chủ yếu yfs\_server sẽ đảm nhiệm việc chạy chính của hệ thống, ở hệ thống này mọi gói tin khi gửi tụi em điều áp dụng SES để sử lý trình tự của các gói tinh.

- Thuật toán SES:

+ Mỗi process lưu 1 vector V\_P kích thước N - 1, N số lượng processes.

+ Mỗi phần tử của V\_P chứa (P’,t): P’ là id của process đích và t là vector timestamp.

- Gửi Message SES:

+ Gửi message M, time stamped tm, cùng với V\_P1 đến P2.

+ Thêm (P2, tm) vào V\_P1. Ghi chồng lên (P2,t), nếu có.

+ (P2,tm) không đc gửi. Các message có (P2,tm) trong V\_P1 chỉ đc chuyển đến P2 khi mà tm < tP2.

- Chuyển giao message SES:

+ If V\_M (in the message) không chứa (P2, t), thì có thể chuyển msg này.

+ (P2, t) exists If tm > tP2, buffer the message. (Don’t deliver).

+ else (tm <= tP2) deliver it.

- Interface của user:

+ Sẽ cho người dùng tương tác với 2 lệnh chính đó là lệnh read và lệnh write.

+ Và yêu cầu người dùng nhập đúng lệnh.

- Logger: là hàm sẽ giúp in các lệnh và cái gói tin mà nó đã gửi ra file/terminal giúp cho người dùng có thể thấy.

- Quy trình mà hệ thông chạy:

+ Đầu tiên khi một process được bật lên thì process sẽ gửi broadcast lên các máy còn lại. Khi có một process khác đang có mặt trong hệ thống thì process nhận đó sẽ lưu lại địa chỉ của process gửi.

+ Sau khi broadcast thành công thì các process sẽ gửi các lệnh mount (Chứa Folder Peer) cho nhau .

- Lệnh read: Có hai trường hợp:

+ Trường hợp 1: nếu nó đọc file của chính nó thì hệ thống sẽ gọi hàm read\_file để đọc.

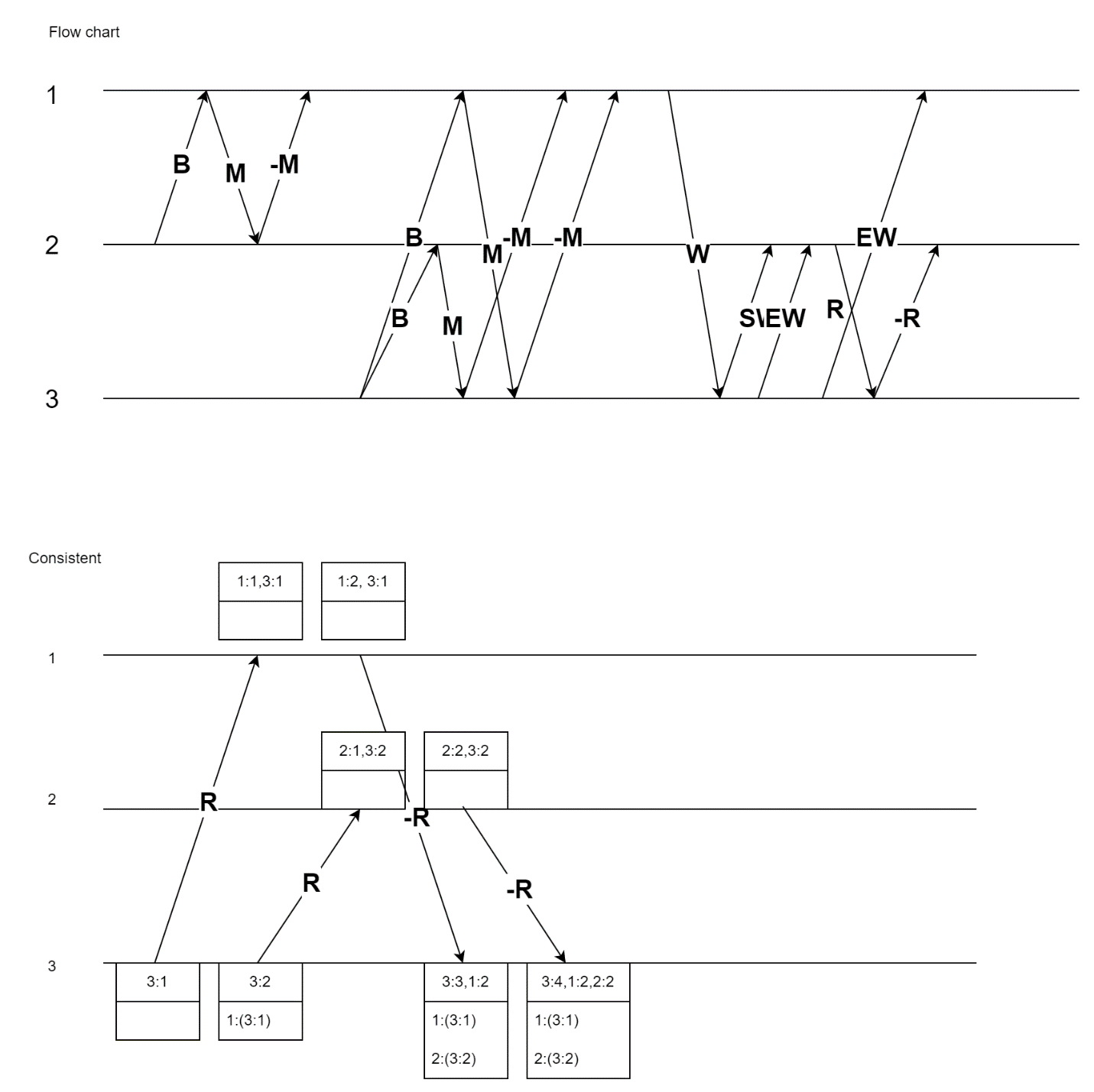
+ Trường hợp 2: nếu nó đọc file của process khác thì sẽ gọi hàm send\_read để xin phép đọc từ process được yêu cầu và prcess được yêu cầu nhận được lệnh đọc sẽ gửi lại nội dung file trong gói tin response.

- Lệnh write:

+ Trường hợp 1: viết file vào file chính nó thì gọi hàm write\_file.

+ Trường hợp 2: nó sẽ gọi hàm send\_write nếu nó muốn viết file của process khác.

+ Cả 2 trường hợp trên điều phải gửi start\_writing và end\_writing để thông báo các processes còn lại

- Flowchart của hệ thống: 

1. **YÊU CẦU VÀ CÁCH CHẠY:**

**# You-Files-System-YFS-**

A network file sharing protocol that defines the way files are stored and retrieved from storage devices across networks

**# Prerequisites**

- Docker with Compose version 3

- Python 3

**# How to run**

Create .env file at root folder

- Set DEBUG=1 at .env to get detailed logs, else set DEBUG=0

**## With user interface:**

1. Add UI=1 to .env

2. Run make\_all\_ui.bat

3. In each opened termial, run ./start\_server.sh to run yfs server

4. Input commands

**## Autorun without user interface:**

1. Add UI=0 to .env

2. Run make\_all.bat

3. See logs in ./Logs and files in ./Dirs

1. **TÀI LIỆU THAM KHẢO:**

- <https://docs.python.org/3/library/socket.html>

- <https://docs.docker.com/compose/compose-file/compose-file-v3/>

- SES slide