Progetto SO 2024/25
Ufficio delle Poste

Versione non definitiva del 26 novembre 2024

Botta, Radicioni, Schifanella C.

Indice

1 Composizione gruppo di studenti

Il progetto sarà sviluppato da un gruppo, composto da 1 a **massimo 3 componenti**. Il gruppo dovrebbe essere composto da studenti dello **stesso turno**, i quali discuteranno con il docente del proprio turno. È tuttavia consentita anche la discussione del progetto di laboratorio da parte di un gruppo di studenti di turni diversi. In questo caso, **tutti** gli studenti del gruppo discuteranno con **lo stesso docente**. Esempio: Tizio (turno T1) e Caio (turno T2) decidono di fare il progetto insieme. Lo consegnano e vengono convocati dal prof. Sempronio il giorno X. Tale giorno Tizio e Caio si presentano e ricevono entrambi una valutazione dal Prof. Sempronio (docente del T1, anche se Caio fa parte del turno T2, il cui docente di riferimento è il prof. Calpurnio).

2 Consegna

Il progetto è costituito da:

1. il codice sorgente
2. una breve relazione che sintetizzi le scelte progettuali compiute

Il progetto si consegna compilando la seguente Google Form, cui si accede con credenziali istituzionali,

- `https://forms.gle/R2J1KXBft3m5ezRt6`

la quale richiederà il caricamento di:

- progetto stesso (un unico file in formato .tgz o .zip NON .rar). Il nome del file deve essere composto dall'unione dei cognomi dei componenti del gruppo (es.: TizioCaio.zip)
- cognome, nome, matricola, email di ciascun componente del gruppo.

Dopo il caricamento del progetto, verrete convocati dal docente con cui discuterete (si veda Sezione 1 in caso di gruppo composto da studenti di turni diversi). Attenzione: **compilare una sola form per progetto** (e **non una per ogni componente del gruppo**). Una eventuale ulteriore consegna prima dell'appuntamento **annullerà la data dell'appuntamento**.

La consegna deve avvenire almeno **10 giorni prima** degli appelli scritti per dare modo al docente di pianificare i colloqui:

- se consegnate con almeno 10 giorni di anticipo rispetto alla data di un appello, il docente propone una data per la discussione entro l'appello seguente
- altrimenti, la data sarà successiva all'appello seguente.

Esempio: per avere la certezza di un appuntamento per la discussione di progetto entro l'appello del 14/01/2025, si deve consegnare entro le ore 24:00 del **04/01/2025**.

3 Valutazione e validità

Il progetto descritto nel presente documento potrà essere discusso se **consegnato entro il 30 Novembre 2025**. Dal Dicembre 2025 sarà discusso il progetto assegnato durante l'anno accademico 2025/26.

Tutti i membri del gruppo devono partecipare alla discussione. La valutazione del progetto è **individuale** ed espressa in 30-esimi. Durante la discussione

- verrà chiesto di illustrare il progetto
- verrà chiesto di commentare il codice e eventualmente di apportare piccole modifiche al progetto
- verranno proposti quesiti sul programma "Unix" del corso anche non necessariamente legati allo sviluppo del progetto.

È necessario ottenere una votazione di almeno **18** su 30 per poter essere ammessi allo scritto. In caso di superamento della discussione del progetto, la votazione conseguita consentirà di partecipare allo scritto per i **cinque appelli successivi** alla data di superamento. Non sono ammesse eccezioni o deroghe a tale periodo di validità.

In caso di mancato superamento, lo studente si potrà ripresentare soltanto dopo almeno **un mese** dalla data del mancato superamento

Si ricorda che il voto del progetto ha un peso di 1/3 sulla votazione finale di Sistemi Operativi.

4 Pubblicazione del proprio progetto, plagio, sanzioni

Copiare altri progetti o parte di essi impedisce una corretta valutazione. Per questo motivo, gli studenti che consegnano il progetto sono consapevoli che:

- la condivisione con altri gruppi attraverso canali pubblici o privati (a titolo di esempio: google drive, canali telegram, github, etc.) di tutto o parte del progetto non è consentita fino a tutto Novembre 2025;
- la copiatura di tutto o parte del progetto non è consentita;
- eventuali frammenti di codice estratti da parti di codice visto a lezione o da altri repository pubblici devono essere opportunamente dichiarati nei commenti del codice e i candidati devono essere in grado di illustrarne il funzionamento.

Nel momento in cui uno studente non rispettasse le sopra citate norme di comportamento, dopo essere stato convocato ed aver avuto modo di illustrare la propria posizione, potrà essere oggetto delle seguenti sanzioni:

- se lo studente avrà nel frattempo superato l'esame di Sistemi Operativi anche successivamente alla data di discussione del progetto, la verbalizzazione del voto verrà annullata;
- se lo studente avrà soltanto superato la discussione del progetto ma non l'esame, la valutazione del progetto verrà annullata e lo studente non potrà accedere ad ulteriore discussione di progetto prima dei due appelli successivi alla data di evidenza di copiatura.

5 Descrizione del progetto: versione minima (voto max 24 su 30)

Si intende simulare il funzionamento di un ufficio postale. A tal fine sono presenti i seguenti processi e risorse.

- Processo `direttore` Ufficio Posta, che gestisce la simulazione, e mantiene le *statistiche* su richieste e servizi erogati dall'ufficio. Genera gli sportelli, e gli operatori.
- Processo `erogatore_ticket`: eroga ticket per specifici servizi. In particolare, dovranno essere implementate le funzionalità per garantire almeno i servizi elencati in Tab. 1. Il tempo indicato è da considerarsi il valore medio per erogare il servizio, e dovrà essere usato per generare un tempo casuale di erogazione nell'intorno ±50% del valore indicato.

Tabella 1: Elenco dei servizi forniti dall'ufficio postale.

servizio	**tempario (in minuti)**
Invio e ritiro pacchi	10
Invio e ritiro lettere e raccomandate	8
Prelievi e versamenti Bancoposta	6
Pagamento bollettini postali	8
Acquisto prodotti finanziari	20
Acquisto orologi e braccialetti	20

- L'utente richiede un ticket specifico per uno dei servizi elencati in Tab. 1, attende il proprio turno, riceve la prestazione richiesta e torna a casa.

- Esistono risorse di tipo `sportello`; ogni sportello è specializzato nel fornire un solo tipo di prestazione, che varia ogni giorno della simulazione (vedi sopra elenco dei possibili servizi). Gli sportelli aprono e chiudono secondo la disponibilità degli operatori.
- `NOF_WORKERS` processi di tipo `operatore`: hanno un orario di lavoro, effettuano pause casuali.
- `NOF_USERS` processi di tipo `utente`. Il processo `utente` decide se recarsi all'ufficio postale e sceglie il servizio da richiedere (si veda la descrizione del processo nella sezione 5.5).

5.1 Processo `Direttore`

Il processo `direttore` è responsabile dell'avvio della simulazione, della creazione delle risorse di tipo sportello, dei processi operatore e utente, delle statistiche e della terminazione. Si noti bene che il processo `direttore` non si occupa dell'aggiornamento delle statistiche, ma solo della loro lettura, secondo quanto indicato. All'avvio, il processo `direttore`:

- crea un solo processo `erogatore_ticket`.
- crea `NOF_WORKER_SEATS` risorse di tipo `sportello`.
- crea `NOF_WORKERS` processi di tipo `operatore`.
- crea `NOF_USERS` processi di tipo `utente`.

Successivamente il `direttore` avvia la simulazione, che avrà come durata `SIM_DURATION` giorni, dove ciascun minuto è simulato dal trascorrere di `N_NANO_SECS` nanosecondi.

La simulazione deve cominciare solamente quando tutti i processi erogatore, operatore e utente sono stati creati e hanno terminato la fase di inizializzazione.

Alla fine di ogni giornata, il processo direttore dovrà stampare le statistiche totali e quelle della giornata, che comprendono:

- il numero di utenti serviti totali nella simulazione
- il numero di utenti serviti in media al giorno
- il numero di servizi erogati totali nella simulazione
- il numero di servizi non erogati totali nella simulazione
- il numero di servizi erogati in media al giorno
- il numero di servizi non erogati in media al giorno
- il tempo medio di attesa degli utenti nella simulazione
- il tempo medio di attesa degli utenti nella giornata
- il tempo medio di erogazione dei servizi nella simulazione
- il tempo medio di erogazione dei servizi nella giornata
- le statistiche precedenti suddivise per tipologia di servizio
- il numero di operatori attivi durante la giornata;
- il numero di operatori attivi durante la simulazione;
- il numero medio di pause effettuate nella giornata e il totale di pause effettuate durante la simulazione;
- il rapporto fra operatori disponibili e sportelli esistenti, per ogni sportello per ogni giornata.

5.2 Processo `erogatore_ticket`

Su richiesta di un processo `utente`, il processo `erogatore_ticket` si occupa di erogare il ticket relativo alla prestazione richiesta, secondo quanto indicato in Tabella 1.

5.3 Risorse `sportello`

Ogni giorno lo sportello è associato a un tipo di servizio dal `direttore`: ogni giorno ci possono essere più sportelli che offrono lo stesso servizio, oppure ci possono essere dei servizi non offerti da alcuno sportello.

Ogni sportello può essere occupato da un singolo operatore; la politica di associazione operatore-sportello è definita dal progettista e deve essere applicata all'inizio di ogni giornata.

5.4 Processo `operatore`

All'avvio, ogni processo `operatore` viene creato in modo che sia in grado di erogare uno dei servizi citati in Sezione 5. Tale mansione resta invariata per tutta la simulazione. All'inizio di ogni giornata lavorativa, l'`operatore`:

- Compete con gli altri operatori per la ricerca di uno sportello libero tra quelli disponibili nell'ufficio postale che si occupano del servizio che lui è in grado di svolgere
- Se ne trova uno, lo occupa e comincia il proprio lavoro che terminerà alla fine della giornata lavorativa
- Con un massimo di `NOF_PAUSE` volte per tutta la simulazione, l'operatore può decidere (secondo un criterio scelto dal programmatore) di interrompere il servizio della giornata anticipatamente. In questo caso:
  - termina di servire il cliente che stava servendo;
  - lascia libero lo sportello occupato;
  - aggiorna le statistiche.

Il processo `operatore` che al suo arrivo non trova uno sportello libero:

- resta in attesa che uno sportello si liberi (per una pausa di un altro operatore);
- torna a casa a fine giornata, e si ripresenta regolarmente il giorno dopo.

5.5 Processo `utente`

Ogni processo `utente` si reca presso l'ufficio postale saltuariamente per richiedere un servizio tra quelli disponibili. Più in dettaglio, ogni giorno ogni processo utente:

- decide se recarsi o meno all'ufficio postale, secondo una probabilità `P_SERV` differente per ogni utente e scelta singolarmente in fase di creazione dell'utente in un intervallo compreso tra i valori [`P_SERV_MIN`, `P_SERV_MAX`].
- In caso affermativo
  - Stabilisce il servizio di cui vuole usufruire (secondo un criterio stabilito dall'utente);
  - Stabilisce un orario (secondo un criterio stabilito dall'utente);
  - Si reca all'ufficio postale;
  - Controlla se quel giorno l'ufficio postale può servire richieste per il tipo di servizio scelto;
  - Se sì, ottiene un ticket per l'apposito servizio;
  - Attende il proprio turno e l'erogazione del servizio;
  - Torna a casa e attende il giorno successivo.

Se al termine della giornata l'utente si trova ancora in coda, abbandona l'ufficio rinunciando all'erogazione del servizio. Il numero di servizi non erogati è uno dei parametri da monitorare.

5.6 Terminazione

La simulazione termina in una delle seguenti circorstanze:

timeout raggiungimento della durata impostata `SIM_DURATION` giorni

explode numero di utenti in attesa al termine della giornata maggiore del valore `EXPLODE_THRESHOLD`

Il gruppo di studenti deve produrre configurazioni (file `config_timout.conf` e `config_explode.conf`) in grado di generare la terminazione nei casi sopra descritti.

Al termine della simulazione, l'output del programma deve riportare anche la causa di terminazione e le statistiche finali.

6 Descrizione del progetto: versione "completa" (max 30)

In questa versione:

- un processo utente, quando decide di recarsi all'ufficio postale, genera una lista di al massimo `N_REQUESTS` richieste di servizi di vario tipo (il numero deve essere scelto in modo casuale per ogni utente, per ogni giorno). Quindi si reca all'ufficio postale dove richiederà in sequenza un ticket per ogni servizio nella lista (il ticket per il servizio i potrà essere richiesto solo quando il servizio $i-1$ è stato completato).
- attraverso un nuovo eseguibile invocabile da linea di comando, deve essere possibile aggiungere alla simulazione altri `N_NEW_USERS` processi utente oltre a quelli inizialmente generati dal `direttore`;
- tutte le statistiche prodotte devono anche essere salvate in un file testo di tipo csv, in modo da poter essere utilizzate per una analisi futura

7 Configurazione

Tutti i parametri di configurazione sono letti a **tempo di esecuzione**, da file o da variabili di ambiente. Quindi, un cambiamento dei parametri non deve determinare una nuova compilazione dei sorgenti (non è consentito inserire i parametri uno alla volta da terminale una volta avviata la simulazione).

8 Requisiti implementativi

Il progetto (sia in versione "minimal" che "normal") deve

- evitare l'attesa attiva
- utilizzare almeno memoria condivisa, semafori e un meccanismo di comunicazione fra processi a scelta fra code di messaggi o pipe,
- essere realizzato sfruttando le tecniche di divisione in moduli del codice (per esempio, i vari processi devono essere lanciati da eseguibili diversi con `execve(...)`),
- essere compilato mediante l'utilizzo dell'utility `make`
- massimizzare il grado di concorrenza fra processi
- deallocare le risorse IPC che sono state allocate dai processi al termine del gioco
- essere compilato con almeno le seguenti opzioni di compilazione:

  ```
  gcc -Wvla -Wextra -Werror
  ```

- poter eseguire correttamente su una macchina (virtuale o fisica) che presenta parallelismo (due o più processori).

Per i motivi introdotti a lezione, ricordarsi di definire la macro `_GNU_SOURCE` o compilare il progetto con il flag `-D_GNU_SOURCE`.