Control de Teoria (grup 40)

29 d'Abril

Contesteu al mateix full

Nom i Cognoms:

---

1. (2 punts) Defineix breument (a l'espai assignat) els següents conceptes:
   (a) Operació d'E/S Assíncrona:

   (b) Taula de Canals:

   (c) Buffering:

   (d) Dispositiu Lògic:

2. (1 punts) Descriu els mecanismes de sincronització entre el procés que realitza una operació de E/S i el gestor que l'atén, ja sigui per informar de l'arribada d'una nova petició, o l'acabament d'una petició en curs. Considera tant el cas de E/S síncrona com el d'assíncrona

3. (7 punts) Volem oferir als usuaris d'Onion un servei de comunicació de processos per la xarxa. El servei s'oferirà a través de dispositius SOCKET, que es podran manipular amb les crides de fitxers tradicionals d´Onion (llegir, escriure, esperar, cancel·lar i tancar) i seran creats amb una nova crida *socket()*, que substitueix a la crida *obrir()* per aquest dispositiu. No hi ha un nombre predefinit de sockets (és a dir, podem tenir tants com volguem), i es vol permetre que diversos processos puguin treballar amb sockets de manera concurrent.

   - *int socket(int canal, int domain, int type, int protocol)*

   Les rutines de nivell BFS disponibles per gestionar els sockets són les següents (considereu-les implementades):

   - *void socket_create_bfs(int domain, int type, int protocol, int * id_socket)*: crea un socket. Retorna a *id_socket* un identificador intern del socket per poder operar amb ell
   - *void socket_send_bfs(int id_socket, char * buffer, int lon)*: envia el missatge de longitud *lon* contingut a *buffer* des d'un socket. En principi aquesta crida no es bloqueja mai, tot i que pel fet d'interactuar amb la xarxa, el temps de transmissió pot arribar a ser molt llarg.
   - *void socket_recv_bfs(int id_socket, char * buffer, int * lon)*: rep un missatge de longitud L des d'un socket. Si no hi ha cap missatge disponible en el socket, la crida es bloqueja a l'espera de que n'arribi un. Si arriba un missatge de longitud més petita que L, la crida retorna aquest missatge, indicant a *lon* la longitud del missatge que ha arribat

   Fixeu-vos que *socket_recv_bfs* sempre intenta rebre un missatge de longitud L, tot i que l'usuari esperi un de més petit. És per això, que els caràcters rebuts sobrants (si n'hi ha) es guardaran en un buffer, de manera que posteriors lectures es puguin fer directament del buffer (si hi ha prous caràcters) sense necessitat de cridar a nivell BFS. De la mateixa manera, si l'usuari espera un missatge de longitud major a L, caldrà fer les crides necessaries a *socket_recv_bfs* fins a obtenir el nombre de caràcters demanats.

   Per facilitar la gestió del buffer, podeu suposar que es disposa de les següents rutines (considereu-les implementades):

   - *void afegir_a_buffer(char * buff_src, char * buff_dst, int lon)*: afegeix el nombre de caràcters indicat per *lon* des de *buff_src* a *buff_dst*
   - *void extreure_de_buffer(char * buff_src, char * buff_dst, int lon)*: extreu el nombre de caràcters indicat per *lon* de *buff_src*, i els copia a *buff_dst*
   - *int consulta_num_buffer(char * buff)*: retorna el nombre de caràcters que hi ha a *buff*

   Assumirem que per un socket determinat, només pot haver-hi una petició de nivell BFS de cada tipus en curs (o sigui, com a màxim una lectura i una escriptura). Per tant, si per satisfer una determinada petició a un socket és necessari fer una crida a nivell BFS i aquell socket ja en té una de pendent d'aquell tipus, la petició es tornarà a encuar a l'espera de ser tractada.

   Considerant que només volem implementar les crides a *llegir* i *escriure*, contesteu els següents apartats:

   (a) (3 punts) Dibuixeu l'esquema de funcionament (3 nivells d'Onion i nivell d'usuari) per a aquest dispositiu utilitzant el nombre de gestors i/o processos auxiliars que us siguin necessaris, indicant totes les sincronitzacions i accessos a cues necessaris

   (b) (3 punts) Quants gestors i/o processos auxiliars calen per gestionar aquest dispositiu? Escriviu el pseudocodi del(s) gestor(s) i/o proces(sos) auxiliar(s) que utilitzeu

   (c) (1 punts) Quants descriptors de dispositiu fareu servir i quins camps (no genèrics) seran necessaris en cadascun d'ells?