Odabrana poglavlja iz operativnih sistema

Projektni zadatak – Raspoređivač zadataka i paralelna obrada multimedijalnih podataka

U proizvoljnom objektno-orijentisanom programskom jeziku realizovati raspoređivač zadataka. Definisati programski API za raspoređivač i realizovati logički odvojenu GUI aplikaciju za raspoređivanje zadataka koja koristi definisani API. Kreirati tip zadatka koji vrši paralelnu obradu multimedijalnih podataka.

Pravilno dokumentovati projekat. Program se mora moći kompajlirati, izvršiti i testirati. Uz priloženi rad treba da se obezbijede i primjeri koji se mogu iskoristiti za testiranje programa. Obezbijediti nekoliko jediničnih testova kojima se demonstriraju funkcionalnosti iz stavki u specifikaciji projektnog zadatka (nije potrebno za GUI aplikaciju i fajl-sistem). Pridržavati se principa objektno-orijentisanog programiranja, SOLID principa, principa pisanja čistog koda i konvencija za korišteni programski jezik.

(15) Raspoređivač zadataka

Pravilno realizovati osnovne funkcionalnosti za raspoređivač zadataka. Raspoređivač zadataka mora biti realizovan u sklopu biblioteke. Obezbijediti jednostavan API koji korisniku omogućava da vrši raspoređivanje zadataka. Omogućiti da se za raspoređivač specificira maksimalan broj zadataka koji se konkurentno izvršavaju. Dozvoljeno je da se definiše sopstveni API koji mora biti korišten u implementaciji zadataka, kao način da se kooperativnim mehanizmima omoguće funkcionalnosti iz specifikacije projektnog zadatka. Dozvoljeno je i da se izvršavanje svakog zadataka pokreće kao zaseban proces. Obezbijediti jednostavne jedinične testove koji demonstriraju rad raspoređivača.

Osnovne funkcionalnosti koje API raspoređivača treba da omogući su:

- dodavanje zadatka u red raspoređivača, sa započinjanjem ili bez započinjanja,
- raspoređivanje zadataka za vrijeme dok se izvršavaju već raspoređeni zadaci,
- započinjanje, pauziranje, nastavak i zaustavljanje zadatka,
- čekanje na kraj izvršavanja zadatka i
- specifikacija algoritma za raspoređivanje, što uključuje FIFO (FCFS) i prioritetno raspoređivanje.

Omogućiti da se za zadatak mogu opcionalno specificirati: datum i vrijeme početka izvršavanja, dozvoljeno ukupno vrijeme izvršavanja i rok do kojeg zadatak mora biti izvršen ili prekinut.

(10) Paralelna obrada multimedijalnih podataka

Omogućiti da se za svaki zadatak specificira i dozvoljeni nivo paralelizma (npr. dozvoljen broj iskorištenih procesorskih jezgri). Definisati tip zadatka za obradu multimedijalnih fajlova. Konkretan algoritam odabrati na kursu predmeta. U GUI aplikaciji obezbijediti podršku za rad sa zadacima definisanog tipa. Zadatak definisanog tipa treba da se može rasporediti na izvršavanje upotrebom raspoređivača. Zadatak treba omogućiti istovremenu obradu veće količine ulaznih fajlova.

Realizovati zadatak tako da on može da vrši paralelnu obradu i nad samo jednim multimedijalnim fajlom. Pri tome se treba ostvariti ubrzanje (testirati nad multimedijalnim fajlom adekvatne veličine).

(10) Dodatni mehanizmi za raspoređivač zadataka

Omogućiti da se vrši prioritetno raspoređivanje sa preuzimanjem, kao i raspoređivanje bazirano na vremenskim isječcima, gdje zadaci višeg prioriteta dobijaju više vremenskih isječaka za izvršavanje. Mehanizmima kooperativnog raspoređivanja simulirati preuzimanje.

Omogućiti rad sa resursima, tako da zadaci mogu da zaključaju resurse u toku izvršavanja, pri čemu su resursi identifikovani jedinstvenim stringom. Pri tome, obezbijediti mehanizme za prevenciju ili detekciju i razrješavanje *deadlock* situacija. Riješiti problem inverzije prioriteta (PIP ili PCP algoritam).

(10) GUI aplikacija i perzistencija zadataka

Obezbijediti aplikaciju sa neblokirajućim grafičkim korisničkim interfejsom za raspoređivanje zadataka. Konstruisati aplikaciju tako da se može lako proširiti novim tipovima zadataka. Omogućiti korisniku aplikacije da specificira način raspoređivanja. Omogućiti korisniku aplikacije da specificira sva svojstva kreiranog zadatka (dozvoljeno vrijeme izvršavanja, rokove, itd.). GUI aplikacija treba da pruži podršku za pregled zadataka u toku izvršavanja, uz prijavu progresa zadatka putem progres bara, kao i za kreiranje, započinjanje, pauziranje, nastavljanje, zaustavljanje i uklanjanje završenih ili zaustavljenih zadataka.

Omogućiti serijalizaciju i deserijalizaciju zadataka. Serijalizovana polja mogu da uključuju, kao primjer, niz putanja na ulazne fajlove i putanju na kojoj treba da se generišu izlazni fajlovi. Omogućiti da se interno stanje aplikacije može serijalizovati (broj zadataka i stanje obrade). Omogućiti da se, u slučaju bilo kakvog zaustavljanja aplikacije (uključujući i neočekivano zaustavljanje), zadaci mogu nastaviti izvršavati (npr. *autosave* mehanizam) od tačke u kojoj su se prethodno zaustavili. Pretpostaviti da su zadaci idempotentni.

(5) Fajl-sistem u korisničkom prostoru

Napisati drajver za fajl-sistem u korisničkom prostoru. Fajl-sistem treba da koristi API raspoređivača i da omogući izvršavanje zadatka za obradu multimedijalnih podataka, na sljedeći način:

- Fajl-sistem treba da sadrži folder *input* u koji se mogu kopirati multimedijalni fajlovi
- Nakon što su fajlovi kopirani u folder *input*, započinje obrada tih fajlova
- Nakon što završi obrada fajlova, rezultati treba da budu dostupni u folderu *output*