ektronski fakultet
atedra za računarstvo

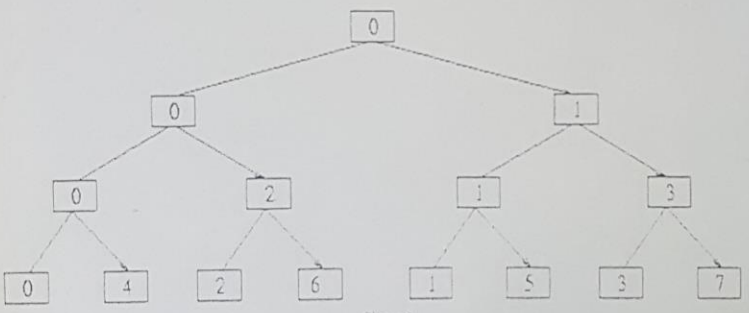
06.10.2020.

DISTRIBUIRANI SISTEMI

Napisati RMI kod koji implementira udaljeni pristup serveru za upravljanje virtuelnom službom za transport putnika.

- Implementirati klasu *Car* koja će da sadrži atribute (id – jedinstveni id vozila, address – trenutno dodeljena adresa, *isFree* – indikacija da li je vozilo slobodno, roundNum - broj realizovanih vožnji)
- Implementirati udaljeni objekat *CarManager* koji će da sadrži listu svih vozila, kao i listu zahtevanih adresa u redu čekanja. U okviru ovog objekta implementirati sledeće metode:
    a. *bool requestCar(string address)* koja vozilu koje ima najmanje realizovanih voznji dodeljuje adresu i vraća *true* (ukoliko postoji više od jednog vozila koja odgovaraju ovom kriterijumu onda se bira vozilo sa najvećim id-jem). Ukoliko ne postoji slobodno vozilo i ako postoji prazno mesto u redu čekanja onda metod ubacuje adresu u red čekanja i vraća *true* dok u suprotnom vraća *false*. Prilikom dodeljivanja adrese potrebno je pozvati metodu *notifyCar(string address)* koja je definisana u okviru interfejsa *CarCallback*.
- Implementirati minimalni kod serverske klase CarServer koja kreira instancu CarManager objekta i upisuje ga u RMI registar kao i klijentskih klasa CarDriverClient (korisnik je vozač) i CarUserClient (korisnik je putnik).

a) Napisati MPI program koji korišćenjem Point-to-Point komunikacije vrši slanje jednog podatka iz procesa 0 svim ostalim procesima u komunikatoru (broj procesa je stepen dvojke). Procesi nakon primanja prikazuju dobijene vrednosti. Procesi su uređeni u stablo, komunikacija se odvija kao što je prikazano na slici, za broj procesa=8.



Slika 1

0 -(korak=1)-> 1
0 -(korak =2)-> 2
1 -(korak =2)-> 3
0 -(korak =3)-> 4
1 -(korak =3)-> 5
2 -(korak =3)-> 6
3 -(korak =3)-> 7

b) Izvršiti distribuciju podataka za primer sa Slike 1. korišćenjem jedne grupne operacije. Nakon toga iskoristiti istu grupnu operaciju za slanje n podataka svim procesima, gde se podaci inicijalizuju u procesu sa rankom 0. Korišćenjem još jedne grupne operacije pronaći p-ti (p-broj procesa) stepen svakog od n brojeva. Korišćenjem još jedne grupne operacije naći proizvode broja procesa i svakog od n brojeva tj.$p*x_0,...p*x_{n-1}$.

3. Pomoću JMS-a kreirati sistem koji omogućava deljenje blanketa među korisnicima. Svi klijenti su povezani u necentralizovanoj mreži. Pri startovanju, svaki klijent šalje ostalim klijentima svoj identifikator. Sve aktivne stanice odgovaraju novo pridošlom klijentu svojim identifikatorom. Svaki klijent ima funkciju „preuzmi(string bblanket)" kojom može zatražiti određeni blanket od bilo kog klijenta koji poseduje dati blanket. Nebitno je koja će korisnik poslati blanket, ali samo jedan sme da pošalje. Predvideti da svaki klijent može da podesi koje blankete poseduje. Voditi računa da se poruke dostavljaju samo zainteresovanim stranama, a ne svima. Navesti na početku sve komunikacione kanale koje koristite i njihovu namenu. Voditi računa da se poruke dostavljaju samo zainteresovanim stranama, a ne svima. Navesti na početku sve komunikacione kanale koje koristite i njihovu namenu. Koristiti isključivo asinhronu komunikaciju.

4. Koristeći WCF kreirajte sistem za rad sa cisternom. Potrebno je podržati mogućnost:
   - dodavanja određenog materijala koji je opisan zapreminom koja je dodata i svojom gustinom.
   - ispusti određenu zapreminu.
   - prikazati trenutno zauzeće cisterne, njenu gustinu kao i procenat popunjenosti.
   - izlistati sve promene nad cisternom.
   - obaveštava sve klijente kada je cisterna prazna.
   - obaveštava sve klijente kada je cisterna puna.

   Smatrati da je kapacitet cisterne zadat na serverskoj strani i ne promenjiv za vreme rada.
   Obavezno izdvojiti interfejs, implementaciju, config i klijentsku stranu koja demonstrira rad servisa.

NAPOMENA: **Radovi koji budu sadržali tragove grafitne olovke će biti diskvalifikovani!**