TelventDMS d.o.o.
Novi Sad

Stručna praksa

Kandidat: **Jelena Pantović**

Tema rada: **Paralelno sabiranje matrica**

Mentor rada: **Saša Marjanović**

Grupa: **Middleware v3, Application Servers subteam**

Grupovođa: **Siniša Cvijanović, Damir Kopčanski**

Novi Sad, avgust, 2012.

SADRŽAJ

SPISAK SLIKA

SKRAĆENICE

WCF	- Windows Communication Foundation
WPF	- Windows Presentation Foundation

1. Uvod

Cilj ovog projektnog zadatka je paralelno sabiranje dve matrice korišćenjem WCF tehnologije, pri čemu je interfejs aplikacije potrebno realizovati pomoću WPF-a. Sistem koji se realizuje treba da se sastoji iz četiri osnovna bloka:

1. korisnika
2. rasporedjivača
3. servera podataka
4. jednog ili više izvrsilaca.

Komunikacija izmedju blokova treba da se obavlja preko WCF-a.

Korisnik treba da omogući generisanje matrica nasumičnih vrednosti i njihov prenos do servera podataka. Pored ovoga, korisnik treba da omogući korisniku aplikacije da sam unese matrice za sabiranje kao i broj izvrsilaca N. Nakon unosa, komanda se izdaje rasporedjivaču.

Server podataka ulazne podatke treba da snima u XML datoteku na disku, i to: matrice koje će se sabirati i rezultantnu matricu C. Sa druge strane, potrebno je da server dobavlja tražene podatke čitajući ih iz XML datoteke.

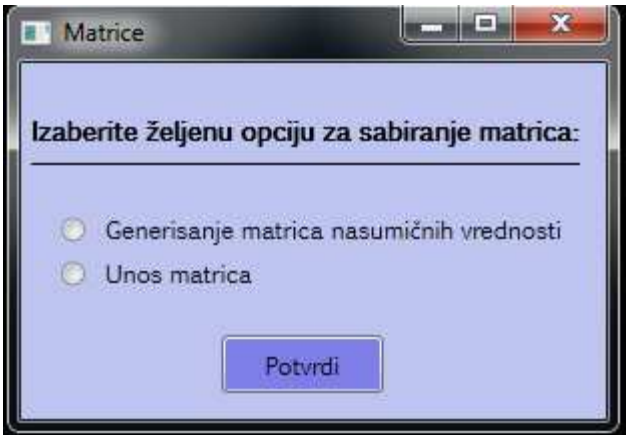
Rasporedjivač treba da omogući prihvatanje korisničke komande. Po prihvatanju komande, rasporedjivač se obraća serveru podataka, dobavlja matrice i organizuje sabiranje te dve matrice na N izvršilaca. Rasporedjivač pokreće izvršioce nakon prijema komande i gasi ih po završetku sabiranja. Kada se sabiranje završi, matrica koja predstavlja rezultat sabiranja snima se u server podataka dok se korisniku vraća referenca na rezultantnu matricu.

Izvršilac obavalja zadatak koji dobije od rasporedjivača, odnosno sabira određene delove matrica.

2. Realizacija projektnog zadatka

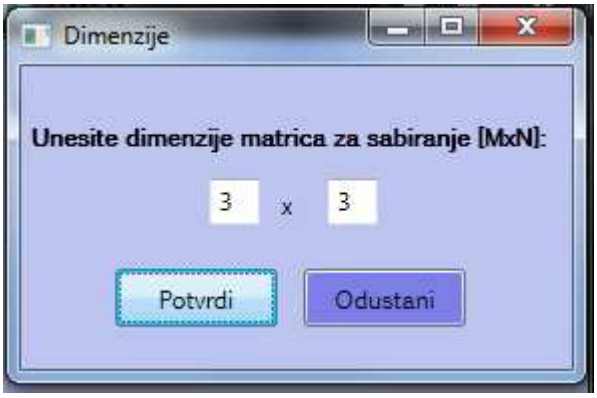
2.1 Korisnik

Interfejs ovog bloka, realizovan je primenom WPF tehnologije. Pri pokretanju aplikacije prikazuje se prozor (Slika 2.1.) gde su korisniku aplikacije ponuđene dve opcije:



Slika 2.1 Glavni prozor aplikacije

Pri izboru prve opcije i klikom na dugme “Potvrdi”, korisniku aplikacije će biti tražen unos dimenzija matrica koje će se sabirati:

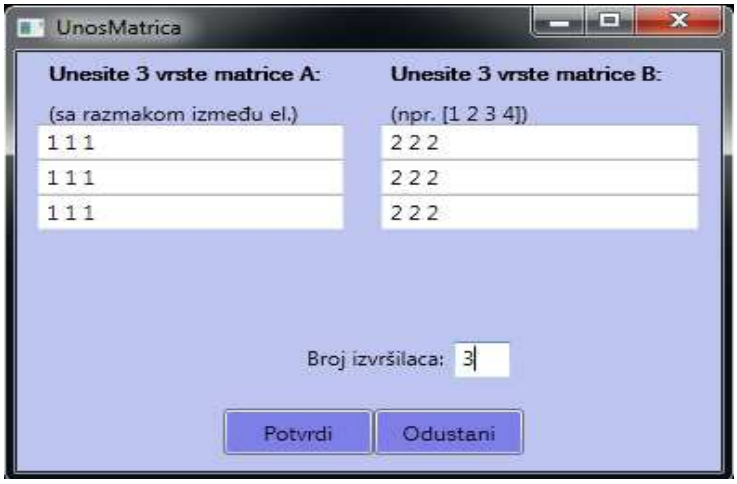


Slika 2.2 Unos dimenzija matrica

Nakon unosa željenih dimenzija, generišu se matrice A i B sa pseudo-slučajnim vrednostima kao elementima:

Slika 2.3 Unos broja izvršilaca

Korisniku aplikacije je omogućen unos željenog broja izvršilaca koji će obavljati sabiranje izgenerisanih matrica, a ukoliko odabere drugu opciju (Slika 2.1), biće omogućen i unos matrica za sabiranje:



Slika 2.4 Unos matrica

Pri unosu broja izvršilaca,dimenzija matrica, kao i samih elemenata matrice, vrši se provera unetih vrednosti. Ukoliko unešene vrednosti nisu korektne, korisnik aplikacije se obaveštava o tome (Slika 2.5) i onemogućava se nastavak korišćenja aplikacije sve dok se ne unesu ispravne vrednosti.

Slika 2.5 Primer rukovanja greškom

Nakon unosa ispravnih vrednosti, pokreće se server podataka, koji čuva izgenerise matrice u XML datoteci. Zatim se pokreće i raspoređivač, prosleđuje mu se broj izvršilaca i daje znak da počne sa realizacijom sabiranja matrica.

2.2 Raspoređivač

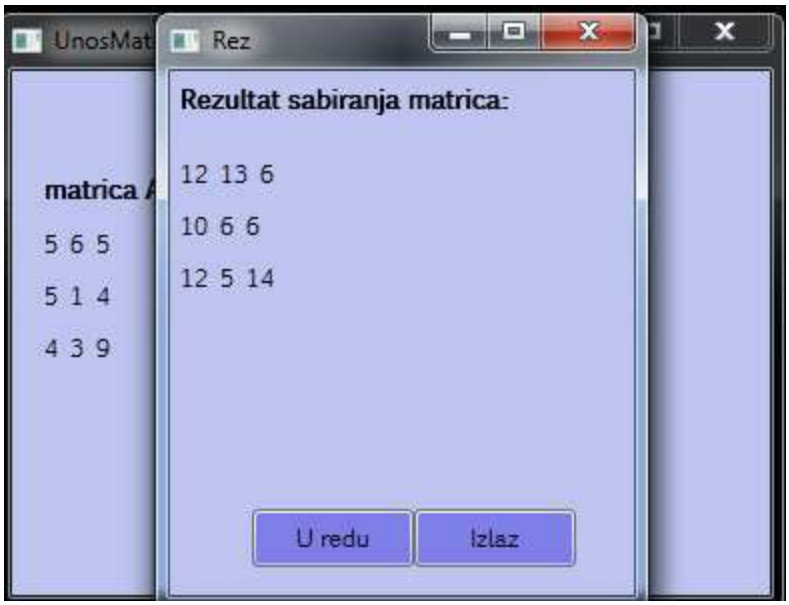
Rasporedjivač realizuje sabiranje metodom *izvrsiSabiranje* klase koja implementira sledeći interfejs:

```
[ServiceContract]
   interface IKomanda
   {
       [OperationContract]
       [FaultContract(typeof(MyFaultException))]
       int[][] izvrsiSabiranje(bool ok, int br);
   }
```

Na osnovu dobijenih podataka, raspoređivač pokreće potreban broj izvršilaca pri čemu se za komunikaciju koristi TCP protokol i otvaraju se odgovarajuće pristupne tačke za komunikaciju s rasporedjivačem. Raspoređivač nakon pokretanja izvršioca dobavlja sačuvane matrice preko servera podataka i konvertuje ih u niz radi podele zadatka između izvršilaca. Nakon odgovarajuće podele, rasporedjivač se obraća izvšiocima i prosleđuje im delove niza za sabiranje. Izvršioci obavljaju zadatak i vraćaju rezultat sabiranja, a potom rasporedjivač parcijalne rezultate spaja u niz i konvertuje ga opet u matricu i na kraju angažuje server podataka koji snima dobijeni rezultat u XML datoteku. Konverzija matrice u niz i obrnuto, kao i podela niza, vrši se metodama klase Matrica:

```
int[] konvertuj_u_niz(int[][] matrica)
int[] podeliNiz(int[] niz,int brojProxija,int brojac,int xc)
int[][] konvertuj_u_matricu(int[] niz,int rows,int cols)
```

Na kraju funkcije *izvrsiSabiranje*, raspoređivač prekida vezu sa izvršiocima gaseći procese, a potom, kao rezultat funkcije, vraća korisniku referencu na rezultantnu matricu. Nakon prihvatanja rezultata sabiranja, korisnik otvara dijalog za prikaz dobijene matrice:



Slika 2.6 Prikaz rezultantne matrice

2.3 Izvršilac

Izvršilac predstavlja WCF servis koji ima ulogu da sabere delove matrice koji su mu prosleđeni od strane raspoređivača. On se otvara na adresi koja sadrži jedinstveni port prosleđen od strane raspoređivača.

Izvršilac mplementira sledeći interfejs za izvršavanje ovog zadatka:

```
[ServiceContract]
interface ISabiranje
{
    [FaultContract(typeof(MyFaultException))]
    int[] SaberiMat(int[] a,int[] b);
}
```

Nakon što se pokrene, on preuzima podatke, vrši sabiranje i kao rezultat vraća niz rasporedjivaču. Ovaj proces ostaje aktivan sve dok se ne izvrši kompletno sabiranje matrica, nakon čega ga raspoređivač gasi.

2.4 Server podataka

Server podataka implementira interfejs čije metode služe za čuvanje i dobavljanje matrica iz XML datoteke:

```
[ServiceContract]
        interface IServerPodataka
        {
            [OperationContract]
            [FaultContract(typeof(MyFaultException))]
            void sacuvajPodatke(int[][] mat);

            [OperationContract]
            [FaultContract(typeof(MyFaultException))]
            List<int[][]> dobaviPodatke();


        }
```

Matrice za sabiranje koje su izgenerisane, ili ručno unešene od strane korisnika, čuvaju se u istoj datoteci u kojoj i rezultantna matrica. Neposredno pre čuvanja u samu datoteku, matrice se smeštaju u listu, koja se kompletna serijalizuje metodom *`void sacuvajPodatke(int[][] mat)`*. Ovoj metodi se pri svakom pozivu prosleđuje po jedna matrica koja se dodaje u listu. Pri deserijalizaciji, cela lista se preuzima iz datoteke metodom *`List<int[][]> dobaviPodatke`* i nakon toga se konkretne matrice izdvajaju iz liste.

3. Zaključak

Radeći ovaj zadatak upoznala sam se sa prednostima korišćenja WCF i WPF tehnologije.

Windows Communication Foundation omogućava da se različiti komunikacioni modeli spoje u jedinstven model. Ova tehnologija pruža programerima mogućnost da izgrade sigurne, pouzdane i brze servis-orijentisane aplikacije. Jedna od glavnih karakteristika WCF-a je interoperabilnost, koja obezbeđuje komunikaciju različitih arhitektura i platformi.

Sa druge strane, time što *Windows Presentation Foundation* razdvaja opis interfejsa od pratećeg programskog koda, omogućeno je da se interfejs aplikacije razvija i menja potpuno nezavisno od same logike aplkacije.

3. Literatura

[1] Scott Klein: *Professional WCF Programming .NET Development with the Windows® Communication Foundation*, Wiley Publishing, Inc., Indianapolis, Indiana, 2007

[2] http://msdn.microsoft.com/

[3] en.wikipedia.org/

[4] http://www.dotnetperls.com/