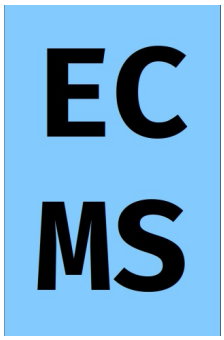


Virtualizacija procesa

Čitanje i keširanje podataka o potrošnji

Energy Consumption Management System(ECMS)

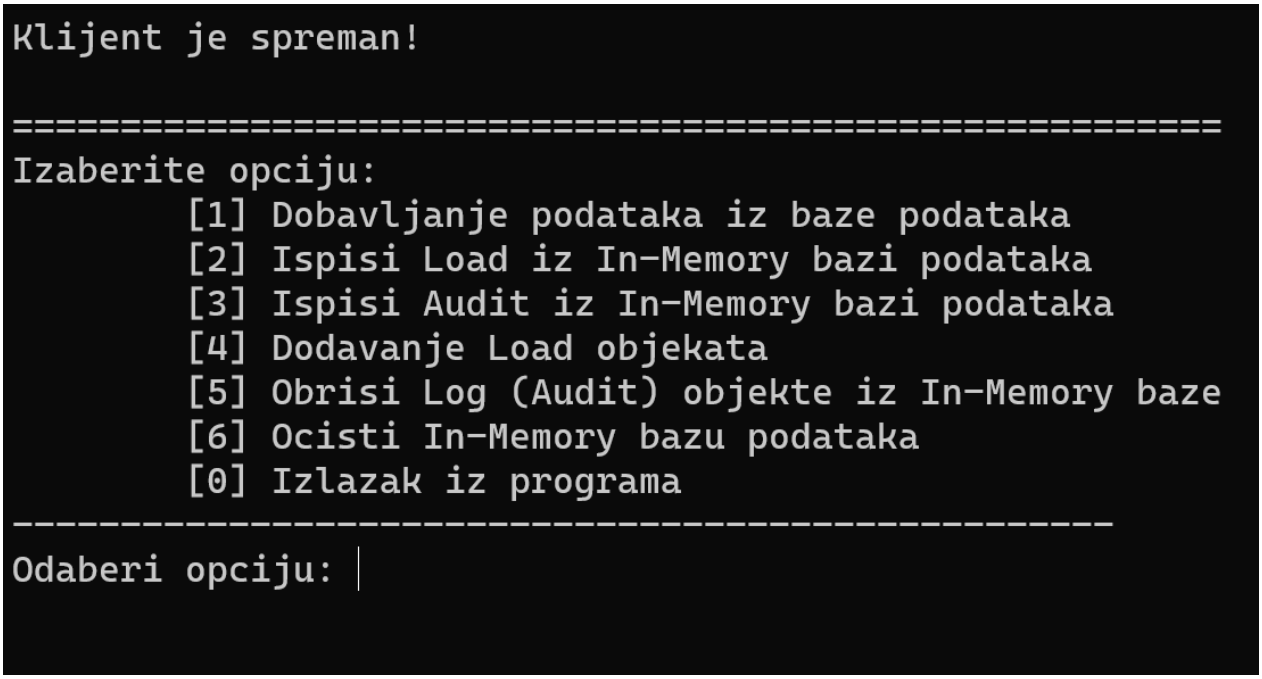
Predmetni zadatak 3

- Dejan Dobrilović PR 85/2020
- Duška Damjanović PR 156/2020
- Tatjana Kosić PR 33/2020
- Luka Bodroža PR 99/2020

Opis projektnog zadatka

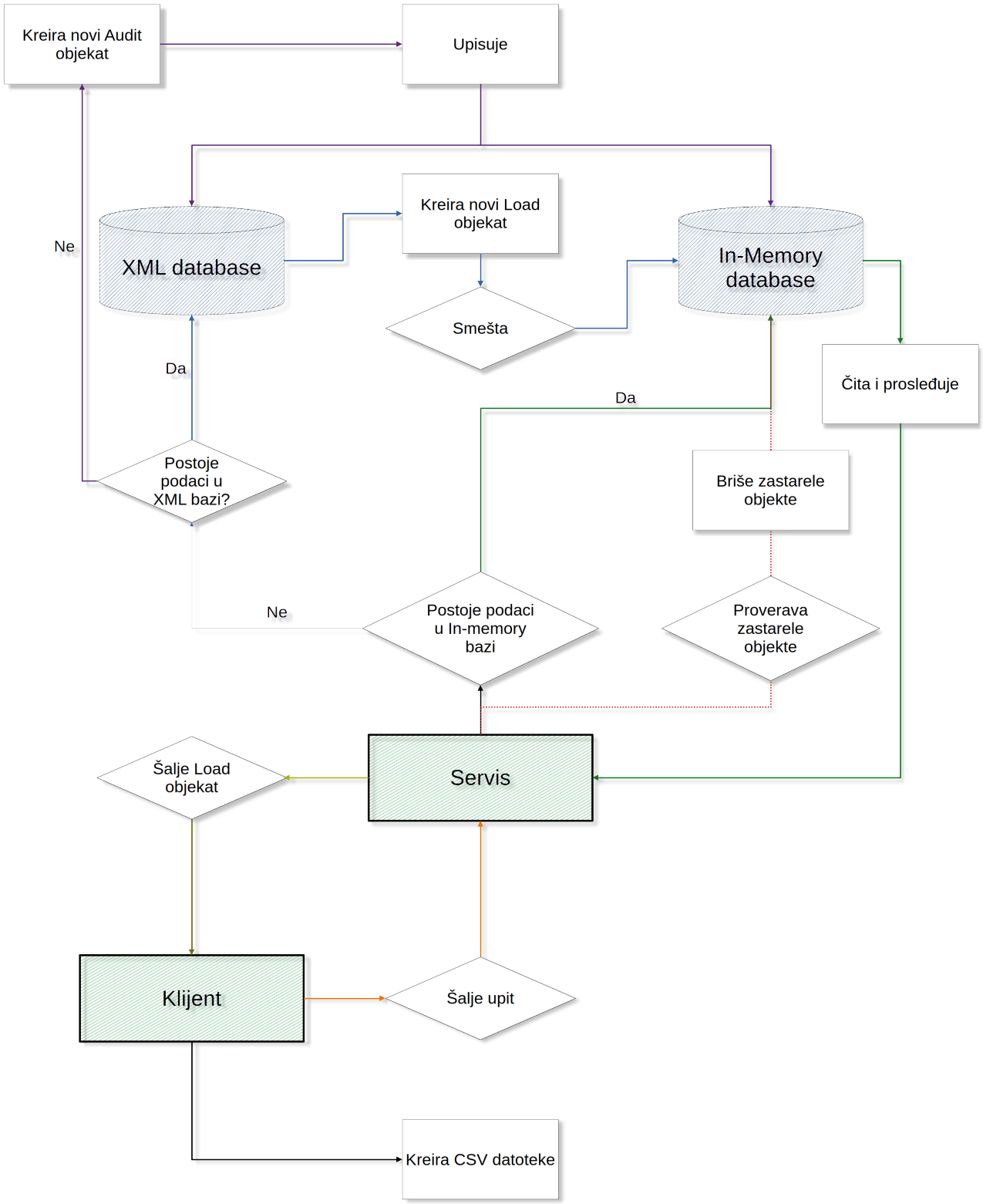
Projektni zadatak predstavlja aplikaciju čija je svrha čitanja i keširanje podataka o potrošnji električne energije, uz uvažavanje zahteva performansi. Čitanje podataka se vrši iz eksternih skladišta podataka(File System-a), te aplikacija implementira istovremeno XML bazu podataka i brzu, keš memoriju, In-Memory bazu podataka. Uspešno pročitan podatak iz XML baze smešta se u In-Memory gde ostaje određen vremenski period nakon kojeg se briše. Postojanje In-Memory baze podataka olakšava dostupnost podataka prilikom čitanja, kao i same performanse programa.

Program poseduje mogućnost ispisa trenutnog sadržaja In-Memory baze podataka, ispis podataka o dešavanjima unutar baze podataka, mogućnosti manualnog dodavanja novih podataka, čišćenje baze podataka od zastarelih podataka o dešavanjima, kao i manualno čišćenje celog sadržaja baze podataka. Ukoliko se svi podaci ne obrišu manualno, u internoj bazi podataka će se nalaziti svi aktualni podaci (ne stariji od vremena unapred definisanog u konfiguraciji programa), a zastareli podaci će biti obrisani automatski.



Slika 1. Meni sa listom operacija

Arhitektura projekta



Load

Id: int
TimeStamp: DateTime
ForecastValue: Double
MeasuredValue: Double

Audit

Id: int
TimeStamp: DateTime
MessageType: Enum {Info, Warning, Error}
Message: String

Tok podataka

Klijent na zahtev šalje upit serverskoj aplikaciji i unosi datum za koji očekuje da dobije podatke. Na osnovu unetog datuma, servis prvo pokušava dobavljanje podataka iz In-Memory baze podataka. U slučaju pogotka, servis prosleđuje podatke do klijenta, u suprotnom traži podatke u XML bazi podataka. Ukoliko podaci postoje u XML bazi, servis ih dobavlja, kopira u In-Memory bazu i tražene podatke prosleđuje do klijenta, a u XML bazu upisuje podatke o uspešnom dobavljanju podataka. U slučaju nepostojanja podataka ni u In-Memory ni u XML bazi, u XML bazu se ispisuje informacija o neupešnom dobavljanju podataka.
Na kraju, klijent na osnovu primljenih podataka koji ispunjavaju njegove zahteve upisuje merenje u CSV datoteke.
Pretraga podataka na osnovu traženog datuma vraća sve vrednosti merenja za traženi datum, te ispisuje njihovu emitovanu kao i izmerenu vrednost sa identifikacionim brojem i tačnim vremenom merenja.

Opis interfejsa i osnovne funkcionalnosti sistema

Program se sastoji iz 2 konzolne aplikacije : Client i Server i biblioteke Common sa zajedničkim klasama za oba projekta. Client i Server komuniciraju preko WCF-a (Windows Communication Foundation), frejmvorka koji služi za pravljenje servisno-orijentisanih aplikacija i omogućava slanje servisnih poruka sa jednog na drugi servis.
Funkcionalnosti programa podeljene su na 4 dela :

1. Service Controller - omogućava upravljanje osnovnim funkcionalnostima na servisu
2. XMLBase Manipulation - omogućava rukovanje XML datotekama i podacima u njima
3. Client Controller - omogućava upravljanje osnovnim funkcionalnostima na klijentu
4. CSV Data - omogućava upravljanje CVS datotekama

Sekcije Service Controller i XMLBase Manipulation su implementirane na servisnom delu, dok su Client Controller i CSV Data implementirani na klijentskom delu aplikacije.

Service Controller:

List<Load> PostRequest(DateTime)	Dobavljanje i obrada zahteva klijenta
bool CheckInMemoryBase(DateTime)	Proveravanje postojanja traženih podataka u In-Memory bazi podataka
List<Load> GetDataFromInMemory(DateTime)	Dobavljanje traženih podataka iz In-Memory baze
void RemoveOldData()	Uklanjanje podataka starijih od predefinisanog vremena

XMLBase Manipulation:

List<Load> ReadData(DateTime)	Čitanje traženih Load podataka iz XML datoteke
bool CheckXMLBase(Load, DateTime)	Potvrda validnosti učitanog podatka za upisivanje u In-Memory bazu
bool WriteLoadInMemory(Load)	Upisivanje Load objekta u In-Memory bazu
void WriteAuditData(int, AuditType, DateTime)	Upisivanje Audit objekta u In-Memory bazu
void LogAudit(string, Audit)	Upisivanje Audit objekta u XML bazu

Client Controller:

DateTime GetInput()	Dobavljanje datuma za željenu pretragu
void GetRequest(List<Load>)	Dobavljanje traženih podataka od servisa

CSV Data:

void LogCSV(List<Load>)	Upisivanje Load objekata u CSV datoteku

Korišćene tehnologije

Tehnologije koje su korišćene prilikom izrade projekta:

1. Okruženje: Visual Studio 2022
2. Programski jezik: C# - .NET Framework (Console Application, Class Library)
3. WCF (Windows Communication Foundation) je frejmvork dizajniran da podrži razvoj distribuiranih sistema tamo gde servisi imaju udaljene potrošače (klijente).
4. Delegati predstavljaju pokazivače na metode koje imaju iste argumente i povratnu vrendnost kao sam delegat (korišćeno prilikom brisanja zastarelih podataka iz In-Memory baze podataka).
5. XMLDocument je klasa koje reprezentuje XML dokument i korišćena je za otvaranje,modifikaciju, validaciju i navigaciju u XML dokumentu.
6. Dispose pattern omogućava ručno oslobađanje resursa, nasledjivanjem i implementiranjem IDispose interfejsa .
7. FileStream, StreamWriter i StreamReader omogućavaju bezbedno rukovanje fajlova u režimima otvaranja, čitanja i pisanja.

Moguće nadogradnje i izmene

Dodatne funkcionalnosti koje bi mogle biti implementirane su:

-Automatsko generisanje/prikupljanje merenja - servis na predefinisan interval prikuplja ili generiše Load objekte i smešta ih u postojeću XML datoteku.

-Sortiranje merenja u datotekama i/ili In-Memory bazi po datumu ili vrednosti merenja - radi optimalnije pretrage servis bi mogao posedovati dodatan thread koji na predefinisan vremenski period ili na predefinisani threshold za broj nesortiranih merenja nakon kojeg se automatski vrši sortiranje podataka u XML datotekama.

-Izmeštanje zastarelih datoteka u zasebne fajlove - radi lakšeg ručnog čitanja i efikasnije pretrage xml datoteka moguće je zadati vremenski interval, na primer

jednu godinu, nakon čega bi se aktualne Load i Audit XML datoteke menjale novim.

-Analiza vrednosti merenja - moguće je za potrebe analize podataka implementirati funkcionalnosti računanja prosečnih merenja za određeni vremenski raspon ili analiza prosečnih odstupanja Measured Value od Forecast Value.

-Skladištenje oštećenih i neispravnih unosa i merenja - u zasebnu XML datoteku bi se mogla skladištiti sva merenja i drugi podaci radi njihovog ručnog pregleda ili oporavka.

-Automatski oporavak loših podataka - na osnovu podataka o prosečnim rezultatima merenja moguće je napraviti zaseban thread koji bi se pobudio radi oporavka loših podataka, ukoliko je to moguće.

-Privremeno oslobađanje interne memorije - zaseban thread bi mogao da na unapred definisani threshold određene Load i Audit datoteke privremeno sačuva na disk i na određeni period ih učitava i proverava njihovu validnost (moguća transformaciji podataka u vid content table gde bi thread prvo označavao podatke za brisanje u datoteci privremeno sačuvanih pojava, a zatim na osnovu parametara poput broja podataka za brisanje ili vremena pozivanja brisao označene podatke, a ostale vraćao u In-Memory bazu podataka ukoliko za njih ima mesta.