

Projektna Dokumentacija- Konzistencija

Boris Čuljak E2-66/2024

26/08/2024

Contents

1 Uvod	3
2 Arhitektura	3
3 Organizacija Koda	4
3.1 Sensors.Contracts.....	4
3.1.1 ISensorService.cs – WCF Ugovor (interfejs servisa).....	4
3.1.2 SensorId.cs – Identitet senzora (enum S1...S10).....	4
3.1.3 SensorReadingDto.cs – DTO poruka (podatak koji ide preko mreze)	4
3.2 Sensors.Data	5
3.2.1 Entities/SensorXReading.cs (×10) – Po jedan entitet/tabela za svaki senzor	5
3.2.2 SensorsDbContext.cs – EF6 kontekst sa 10 `DbSet<>`	5
3.2.3 Sensors.Data/Migrations/ – EF migracije.....	5
3.3 Sensors.ServiceHost	6
3.3.2 Jobs/ReplicateReconciliationJob.cs_ - Logika poravnanja (svakih 60 s).....	6
3.3.3 Services/SensorService.cs - Implementacija WCF ugovora	7
3.4 Sensors.Simulator	7
3.4.1 Client/Program.cs - Pokreće 10 radnika i upravlja gašenjem.....	7
3.4.2 Client/SensorWorker.cs - Logika jednog senzora	7
3.4.3 Client/WcfClientFactory.cs - Kreira WCF kanal.....	8
4 Pokretanje Projekta.....	8
5 Rezultati.....	8
5.1 Izlaz u Konzoli	8
5.2 Podaci u Bazi	9

1 Uvod

- **Cilj:** WCF sistem koji simulira **10 senzora temperature**. Svaki 1–10 sekundi šalje nasumičnu temperaturu u SQL bazu (EF6).
- **Replikaciono poravnanje:** Svakih **60 sekundi** računa se „konsenzus“ i upisuje u **svih 10** tabela kao `Source = "Reconciled"`.
- **Pravilo konsenzusa:** Na svakih 60 s uzimamo pregled sistema, po jedno najnovije RAW očitavanje iz svake od 10 tabela. Iz tih 10 vrednosti izračunamo prosek. Konsenzus je ona vrednost iz tog pregleda koja je u opsegu $\pm Tolerance$ (podrazumevano 5 °C) oko proseka i pritom je vremenski najskorija. Ako nijedna od 10 najnovijih vrednosti ne upada u taj opseg, kao konsenzus se uzima vremenski najskorija vrednost iz pregleda (bez obzira na odstupanje). Ta vrednost se zatim upisuje u svih 10 tabela sa Source="Reconciled" (vreme poravnato na minut).
- **Tehnologije:** .NET Framework 4.8, WCF (net.tcp), Entity Framework 6.4.4, SQL LocalDB (MSSQLLocalDB).

2 Arhitektura

Simulator pokreće 10 zadataka. Svaki zadatak na 1–10 sekundi generiše temperaturu i poziva SubmitReading preko net.tcp. Poziv je jednosmeran pa Simulator ne čeka odgovor.

ServiceHost izlaže ISensorService na net.tcp://localhost:9001/SensorService. SubmitReading servis uzima vreme sa servera u UTC, bira tabelu na osnovu SensorId i upisuje red sa Source = Raw. Upis radi EF6 preko SensorsDbContext. Klijent nema pristup bazi.

Svakih 60 sekundi tajmer u ServiceHost pokreće ReplicateReconciliationJob. Posao čita po jedno najnovije Raw očitavanje iz svake od 10 tabela i računa prosek. Ako postoji bar jedna vrednost unutar $\pm Tolerance$ u odnosu na prosek, bira se vremenski najskorija takva vrednost. Ako ne postoji nijedna u opsegu, bira se vremenski najskorija vrednost uopšte. Izabrana vrednost se upisuje u svih 10 tabela kao novi red sa Source = Reconciled. Vreme upisa je UTC i može da se poravna na početak minuta iz konfiguracije.

Baza ima 10 tabela sa istom šemom: Id, TimestampUtc, ValueCelsius, Source. Odvojene tabele modeluju replike koje se periodično poravnavaju. UTC vreme uklanja probleme sa vremenskim zonama i driftom klijenta. Server određuje šta je „najnovije“.

App.config u ServiceHost drži WCF endpoint i binding, connection string i appSettings (Tolerance, period, poravnanje na minut). `_Diagnostics_` su isključeni da bi servis radio bez admin prava.

Zavisnosti: Simulator → Contracts. ServiceHost → Contracts i → Data. Data → EntityFramework.
Contracts → .NET WCF a ne zavisi od EF-a. Simulator ne zavisi od Data sloja.

3 Organizacija Koda

SENSORSOLUTION/

SENSORS.CONTRACTS/ # WCF UGOVOR (SERVICE CONTRACT) + DTO PORUKE
SENSORS.DATA/ # EF6 ENTITETI (10 TABELA) + DBCONTEXT + MIGRACIJE
SENSORS.SERVICEHOST/ # WCF SELF-HOST SERVIS + JOB ZA PORAVNANJE + APP.CONFIG
SENSORS.SIMULATOR/ # KLIJENT: 10 „SENZORA“ KOJI ŠALJU OČITAVANJA

3.1 Sensors.Contracts

Komunikacija između klijenta i servisa. Sadrži service contract (interfejs koji opisuje šta se poziva) i DTO poruke (kako izgleda podatak koji šaljemo). Nema baze ni poslovne logike.

3.1.1 ISensorService.cs – WCF Ugovor (interfejs servisa)

Deklariše dostupne metode (npr. `SubmitReading`, `Ping`, opciono read-metode).

```
[ServiceContract]
public interface ISensorService
{
    [OperationContract(IsOneWay = true)]
    void SubmitReading(SensorReadingDto reading); // klijent šalje, ne čeka odgovor
}
```

3.1.2 SensorId.cs – Identitet senzora (enum S1...S10)

Jednoznačno označava koji senzor šalje očitavanje.

```
[DataContract]
public enum SensorId { [EnumMember] S1 = 1, /* ... */ [EnumMember] S10 = 10 }
```

3.1.3 SensorReadingDto.cs – DTO poruka (podatak koji ide preko mreze)

Struktura sa poljima: koji senzor, koja temperatura, vremenske oznake, izvor.

```
[DataContract]
public class SensorReadingDto
{
    [DataMember(Order = 1, IsRequired = true)] public SensorId SensorId { get; set; }
    [DataMember(Order = 2, IsRequired = true)] public double ValueCelsius { get; set; }
    // Server popunjava ServerTimestampUtc; Source: "Raw" ili "Reconciled"
}
```

3.2 Sensors.Data

Sloj za podatke (EF6). Definiše šemu baze (10 tabela - po jedna po senzoru), `DbContext` i migracije. Replikaciono poravnanje koristi ove tabele.

3.2.1 Entities/SensorXReading.cs (×10) – Po jedan entitet/tabela za svaki senzor

Modeluje 10 replika (1 klasa = 1 tabela).

```
public class Sensor1Reading
{
    public int Id { get; set; }
    public DateTime TimestampUtc { get; set; } // vreme upisa (server)
    public double ValueCelsius { get; set; } // temperatura
    public string Source { get; set; } // "Raw" ili "Reconciled"
}
```

3.2.2 SensorsDbContext.cs – EF6 kontekst sa 10 `DbSet<>`

Mapira entitete na tabele i podešava konvencije.

```
public class SensorsDbContext : DbContext
{
    public SensorsDbContext() : base("name=SensorsDb") { } // connection string ime
    public DbSet<Sensor1Reading> Sensor1Readings { get; set; }
    // ... do Sensor10Readings
    protected override void OnModelCreating(DbModelBuilder modelBuilder)
    {
        modelBuilder.Conventions.Remove<PluralizingTableNameConvention>(); // klasa
        == tabela
        modelBuilder.Entity<Sensor1Reading>().Property(p =>
        p.Source).IsRequired().HasMaxLength(20);
        // ... isto za ostale entitete
    }
}
```

3.2.3 Sensors.Data/Migrations/ – EF migracije

Reproducibilna šema baze.

- Enable-Migrations -ProjectName Sensors.Data -StartupProjectName Sensors.ServiceHost

- Add-Migration InitialCreate -ProjectName Sensors.Data -StartupProjectName Sensors.ServiceHost
- Update-Database -ProjectName Sensors.Data -StartupProjectName Sensors.ServiceHost

3.3 Sensors.ServiceHost

Proces (WCF self-host) sluša na *net.tcp://localhost:9001/SensorService*. Prima SubmitReading, server beleži vreme i čuva RAW u bazu. Na svakih 60s pokreće replikaciono poravnanje (konsenzus) i upisuje Reconciled u svih 10 tabela.

Timestamp je na serveru te se time eliminišu razlike klijentskih satova. „Najnovije“ je uvek po serveru.

3.3.1 Hosting/Program.cs - Startuje WCF host i tajmer

```
using (var host = new System.ServiceModel.ServiceHost(typeof(SensorService))) //
param-less: baseAddress u App.config
{
    host.Open();
    _timer = new Timer(_ => SafeRun(), null, TimeSpan.FromSeconds(60),
        TimeSpan.FromSeconds(60));
    Console.WriteLine("[ServiceHost] Reconciliation timer started (every 60s).");
}
```

Koristi se parametarski-prazan ServiceHost jer je `_baseAddress_` u App.config-u. Tako se izbegava greška „already contains an address with scheme net.tcp...“ Dodatno Pošto je naziv projekta Sensors.ServiceHost, koristi se puno ime System.ServiceModel.ServiceHost da ne dođe do konflikta.

3.3.2 Jobs/ReplicateReconciliationJob.cs_ - Logika poravnanja (svakih 60 s)

```
double tolerance = ReadDouble("Tolerance", 5.0); // ±Tolerance iz appSettings
var avg = latest.Average(x => x.value); // prosek „najnovijih“ 10 RAW
var inRange = latest.Where(x => Math.Abs(x.value - avg) <= tolerance).ToList();
var chosen = inRange.Any() ? inRange.OrderByDescending(x => x.ts).First()
    : latest.OrderByDescending(x => x.ts).First(); // fallback: najnovija
ukupno
var insertTs = ReadBool("AlignToWholeMinute", true) ?
AlignToMinute(DateTime.UtcNow) : DateTime.UtcNow;
// upis „chosen.value“ u SVE tabele sa Source="Reconciled"
db.SaveChanges();
```

3.3.3 Services/SensorService.cs - Implementacija WCF ugovora

```
var serverNow = DateTime.UtcNow;           // vreme pečatira server
db.SensorXReadings.Add(new SensorXReading { // zapis RAW u odgovarajuću
tabelu
    TimestampUtc = serverNow, ValueCelsius = reading.ValueCelsius, Source = "Raw"
});
// ...
return Project(db.SensorXReadings.OrderByDescending(x =>
x.TimestampUtc).FirstOrDefault(), SensorId.SX);
```

3.4 Sensors.Simulator

Konzolni klijent koji simulira 10 senzora. Svaki „radnik“ (task) na 1–10 s generiše nasumičnu temperaturu i poziva servis ('SubmitReading') preko net.tcp. Simulator nema EF/bazu već priča sa servisom preko WCF ugovora iz 'Sensors.Contracts'.

3.4.1 Client/Program.cs - Pokreće 10 radnika i upravlja gašenjem

```
// Ctrl+C -> otkazivanje svih radnika
Console.CancelKeyPress += (s, e) => { e.Cancel = true; cts.Cancel(); };
// Start S1..S10
for (int i = 1; i <= 10; i++)
    tasks.Add(new SensorWorker((SensorId)i, seedBase + i).RunAsync(cts.Token));
await Task.WhenAll(tasks); // sačekaj sve radnike
WcfClientFactory.Close(); // uredno zatvori WCF resurse
```

3.4.2 Client/SensorWorker.cs - Logika jednog senzora

```
// Pauza 1–10 s između merenja
var delayMs = _rng.Next(1, 11) * 1000;
await Task.Delay(delayMs, ct);
// Nasumična temperatura ~ [18, 30) °C
var value = 18.0 + _rng.NextDouble() * 12.0;
// Slanje očitavanja servisu (Source=Raw; server pečatira vreme)
WcfClientFactory.GetChannel().SubmitReading(new SensorReadingDto {
    SensorId = _sensorId, ValueCelsius = value, ClientTimestampUtc = DateTime.UtcNow,
    Source = "Raw"
});
```

3.4.3 Client/WcfClientFactory.cs - Kreira WCF kanal

```
// Jedan zajednički ChannelFactory<ISensorService> za sve radnike
var binding = new NetTcpBinding();
var address = new EndpointAddress("net.tcp://localhost:9001/SensorService");
_factory = new ChannelFactory<ISensorService>(binding, address);
// (Re)otvori kanal po potrebi
_channel = _factory.CreateChannel();
((IClientChannel)_channel).Open();
// Uredno zatvaranje (Close/Abort) pri izlasku
```

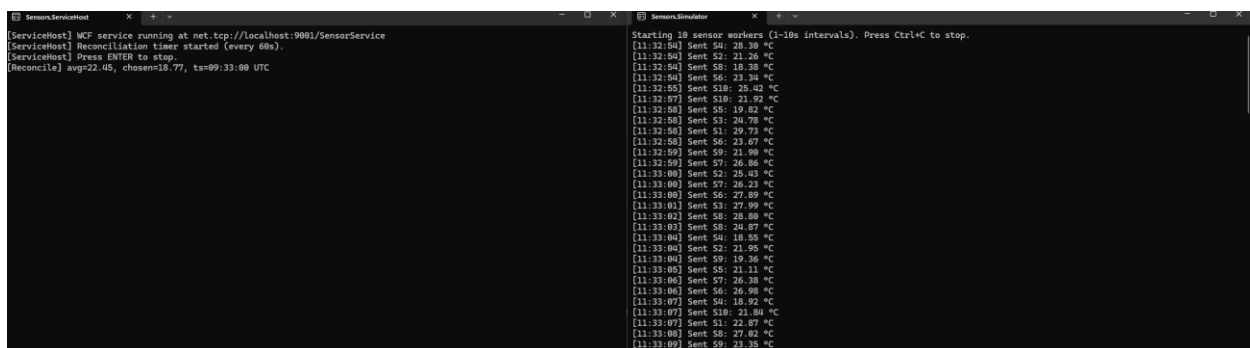
4 Pokretanje Projekta

1. Otvoriti rešenje u Visual Studio 2022+ (target: .NET Framework 4.8).
2. Podesiti Multiple startup projects: Sensors.ServiceHost = Start, Sensors.Simulator = Start (ServiceHost prvi).
3. Ctrl+F5.
4. Otvaraju se dva prozora:
 - ServiceHost: „WCF service running...” + „Reconciliation timer started...”.
 - Simulator: „Starting 10 sensor workers...” + periodične poruke „Sent S#: XX.XX °C”.

5 Rezultati

U nastavku su prikazani rezultati rada sistema. Komunikacija između simulatora i servisa u konzoli, kao i stanje podataka u bazi.

5.1 Izlaz u Konzoli



The screenshot shows two console windows side-by-side. The left window, titled 'Sensors.ServiceHost', displays the following output:

```
[ServiceHost] WCF service running at net.tcp://localhost:9001/SensorService
[ServiceHost] Reconciliation timer started (every 60s).
[ServiceHost] Press ENTER to stop.
[Reconcile] avg=22.45, chosen=18.77, ts=09:33:00 UTC
```

The right window, titled 'Sensors.Simulator', displays the following output:

```
Starting 10 sensor workers (1-10s intervals). Press Ctrl+C to stop.
[11:32:54] Sent S4: 20.38 °C
[11:32:54] Sent S2: 21.25 °C
[11:32:54] Sent S8: 18.38 °C
[11:32:54] Sent S6: 23.34 °C
[11:32:55] Sent S18: 25.42 °C
[11:32:57] Sent S18: 21.92 °C
[11:32:58] Sent S8: 19.82 °C
[11:32:58] Sent S3: 24.78 °C
[11:32:58] Sent S1: 29.73 °C
[11:32:58] Sent S6: 23.47 °C
[11:32:59] Sent S9: 21.98 °C
[11:32:59] Sent S7: 26.86 °C
[11:33:00] Sent S2: 25.43 °C
[11:33:00] Sent S7: 26.23 °C
[11:33:00] Sent S6: 27.89 °C
[11:33:01] Sent S3: 27.99 °C
[11:33:02] Sent S8: 25.88 °C
[11:33:03] Sent S8: 24.87 °C
[11:33:04] Sent S4: 18.55 °C
[11:33:04] Sent S2: 21.95 °C
[11:33:04] Sent S9: 19.36 °C
[11:33:05] Sent S5: 21.11 °C
[11:33:06] Sent S7: 26.38 °C
[11:33:06] Sent S6: 26.98 °C
[11:33:07] Sent S4: 18.92 °C
[11:33:07] Sent S18: 21.84 °C
[11:33:07] Sent S1: 22.87 °C
[11:33:08] Sent S8: 27.82 °C
[11:33:09] Sent S9: 23.35 °C
```

Na levoj strani je ServiceHost koji pokazuje:

- Da je WCF servis uspešno pokrenut na net.tcp://localhost:9001/SensorService,
- Da se reconciliation tajmer aktivira svakih 60 sekundi,
- Log poruke procesa poravnanja (avg=..., chosen=..., ts=...).

Na desnoj strani je Simulator koji prikazuje rad 10 senzora:

- Svaki red prikazuje trenutak kada je određeni senzor poslao temperaturu,
- Vrednosti se kreću u intervalu od ~18 °C do ~30 °C,
- Intervali slanja su nasumični (1–10 sekundi), što se vidi po neujednačenim vremenskim oznakama.

5.2 Podaci u Bazi

86	26/08/2025 09:3...	29.73408821771...	Raw
87	26/08/2025 09:3...	22.86601999442...	Raw
88	26/08/2025 09:3...	23.46017099426...	Raw
89	26/08/2025 09:3...	20.87527031399...	Raw
90	26/08/2025 09:3...	18.24813329067...	Raw
91	26/08/2025 09:3...	19.76102018810...	Raw
92	26/08/2025 09:3...	26.61355316853...	Raw
93	26/08/2025 09:3...	25.34396166696...	Raw
94	26/08/2025 09:3...	21.86114494309...	Raw
95	26/08/2025 09:3...	22.06258037689...	Raw
96	26/08/2025 09:3...	18.77447534202...	Reconciled
97	26/08/2025 09:3...	24.02211738658...	Raw
98	26/08/2025 09:3...	26.58232482363...	Raw
99	26/08/2025 09:3...	26.37978193367...	Raw
100	26/08/2025 09:3...	23.46216613308...	Raw
101	26/08/2025 09:3...	24.13908002811...	Raw
102	26/08/2025 09:3...	20.71333850487...	Raw
103	26/08/2025 09:3...	21.25005355256...	Raw

85	18/08/2025 12:1...	22.51706532226...	Raw
86	18/08/2025 12:1...	27.42719969219...	Raw
87	26/08/2025 09:3...	25.42329891744...	Raw
88	26/08/2025 09:3...	21.91785649765...	Raw
89	26/08/2025 09:3...	21.84113411597...	Raw
90	26/08/2025 09:3...	27.05718581800...	Raw
91	26/08/2025 09:3...	22.07325183603...	Raw
92	26/08/2025 09:3...	20.07818026751...	Raw
93	26/08/2025 09:3...	22.80112224854...	Raw
94	26/08/2025 09:3...	20.79221846665...	Raw
95	26/08/2025 09:3...	28.82107226495...	Raw
96	26/08/2025 09:3...	18.77447534202...	Reconciled
97	26/08/2025 09:3...	27.88070767646...	Raw
98	26/08/2025 09:3...	26.51848072210...	Raw
99	26/08/2025 09:3...	28.31167979087...	Raw
100	26/08/2025 09:3...	23.84354706380...	Raw
101	26/08/2025 09:3...	18.37982578034...	Raw
102	26/08/2025 09:3...	27.11791582457...	Raw

Na slici su prikazane dve tabele iz baze (Sensor1Reading i Sensor10Reading). Vidimo sledeće obrasce:

- RAW redovi: Pojedinačna očitavanja senzora, koja imaju oznaku Source = Raw. Njih generiše simulator na svaka 1–10 sekundi.
- RECONCILED redovi: Na svakih 60 sekundi dodaje se novi red sa Source = Reconciled. Ta vrednost je rezultat poravnanja:
 - U svim tabelama se upisuje ista vrednost,
 - Timestamp je poravnat na početak minuta,
 - Ta vrednost predstavlja „konsenzus“ između najnovijih RAW očitavanja.

U obe tabele se vidi red sa Source = Reconciled za isti vremenski trenutak (09:33:00 UTC), i vrednost je ista, što potvrđuje da se mehanizam poravnanja uspešno izvršava i sinhronizuje sva očitavanja.