



UNIVERZITET U NOVOM SADU  
FAKULTET TEHNIČKIH NAUKA  
NOVI SAD



Industrijski komunikacioni protokoli u elektroenergetskim  
sistemima  
- Primenjeno softversko inženjerstvo -

PR34/2017 Vesna Prica  
PR15/2017 Tamara Jović

Novi Sad, 2021.

## Sadržaj

1. Uvod: Publisher - Subscriber .....	3
2. Dizajn.....	4
2.1. Procesi.....	5
3. Strukture podataka .....	6
4. Rezultati testiranja .....	6
5. Zaključak.....	7
6. Potencijalna unapređenja .....	7

## 1. Uvod: Publisher - Subscriber

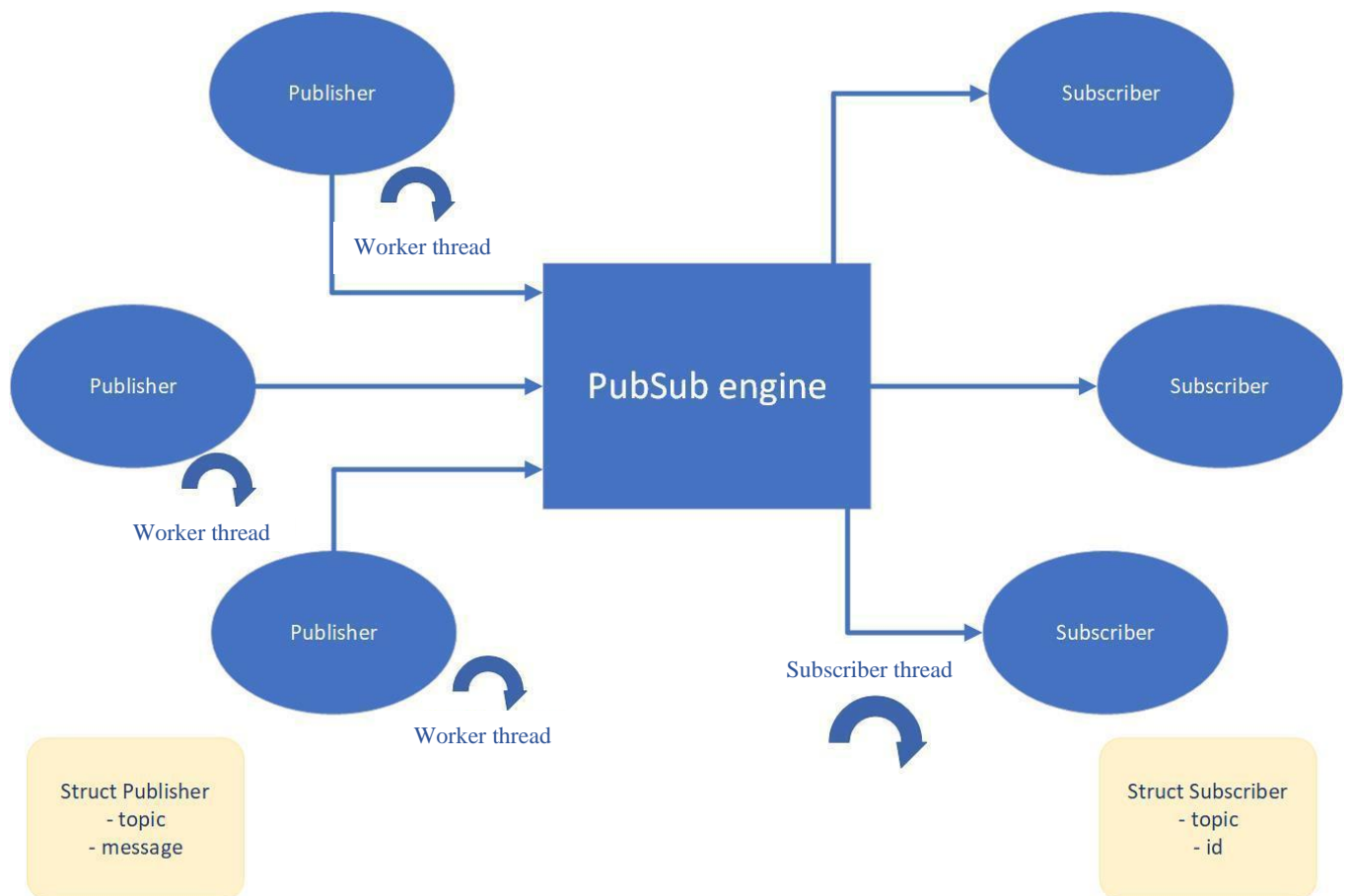
Potrebno je napraviti PubSub servis koji može da opslužuje proizvoljan broj klijenata. Servis treba da implementira sledeći interface:

- `void Connect();`
- `void Subscribe(void *topic);`
- `void Publish(void * topic, void* message)`

Topic treba da bude proizvoljan niz karaktera. Potrebno je koristiti IO completion ports mehanizma.

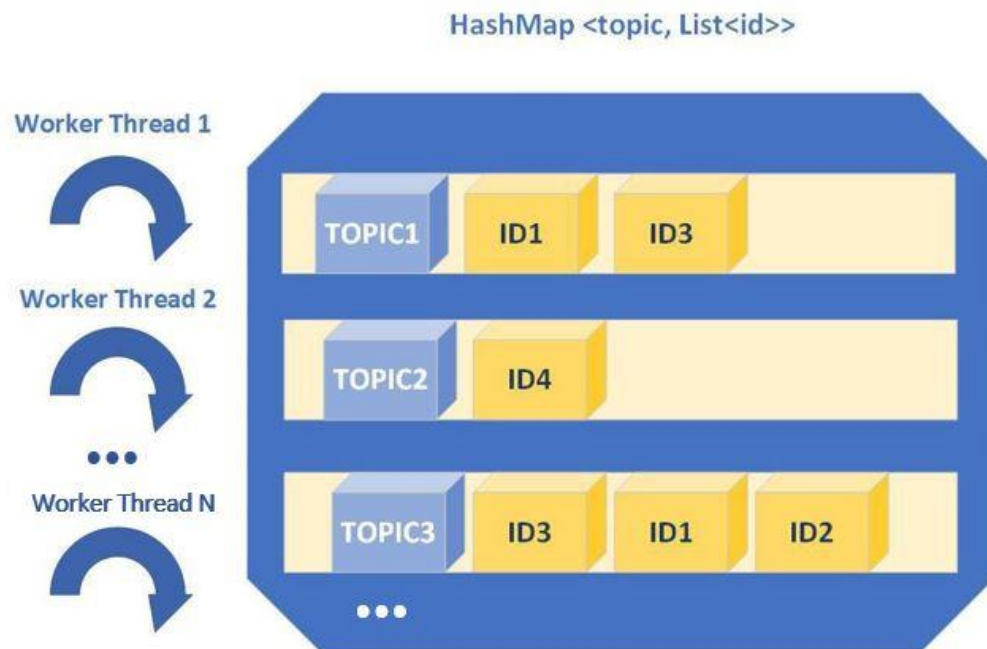
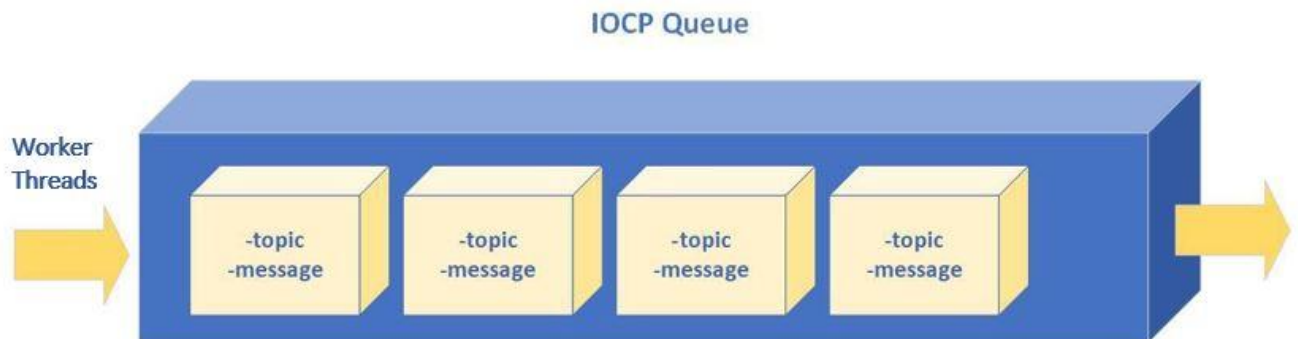
## 2. Dizajn

Komponente komuniciraju putem TCP-ja čiji su *socket*-i u neblokirajućem režimu koji garantuje bolju iskorišćenost performansi i garantuje isporuku poruka. Server *PubSub engine* omogućava povezivanje sa više klijenata i obradu njihovih zahteva. IOCP mehanizam to omogućava pomoću asinhronog rada.



## 2.1. Prosesi

- *Publisher*
- *Subscriber*
- *PubSub engine*(IOCP server)



### 3. Strukture podataka

Na server se povezuju dve vrste klijenata, *publisher*-i i *subscriber*-i, zbog toga su korišćene dve structure:

- *Publisher* sa poljima *topic*(tema) i *message*(vest/poruka na odgovarajuću temu)
- *Subscriber* sa poljima *topic*(tema na koju se klijent pretplaćuje) i *id*.

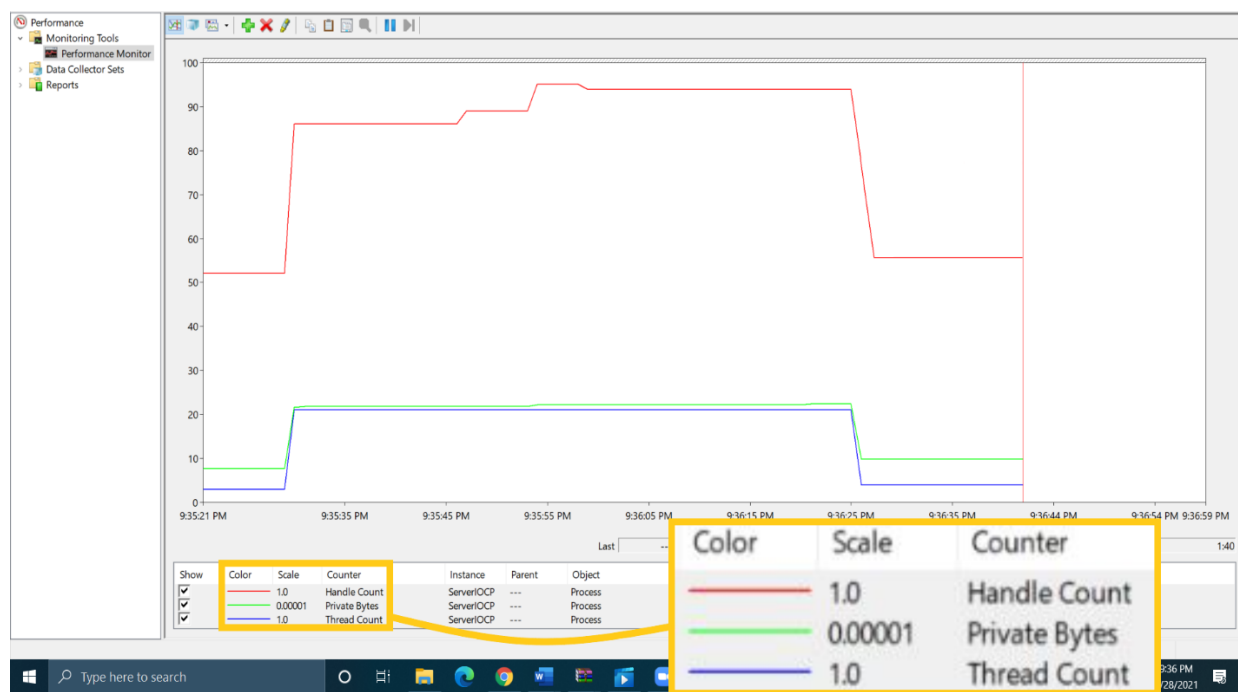
*IOCP queue* se koristi za smeštanje poslatih vesti. *Subscriber* se pretplaćuje na proizvoljan broj tema. *Publisher* šalje vest na odgovarajuću temu serveru. Na serveru se probudi jedan od slobodnih *Worker Thread*-ova, čiji je zadatak da obradi pristiglu poruku i pošalje je klijentima pretplaćenim na tu temu.

*PubSub engine* sadrži *HashMap*-u gde je ključ *topic*, a vrednost lista pretplaćenih *subscriber*-a na taj *topic*. Ključ predstavlja jedinstvenu *hash* vrednost koja se dobija preko *topic*-a, što znači da nije moguće dobiti istu *hash* vrednost za dva različita *topic*-a.

### 4. Rezultati testiranja

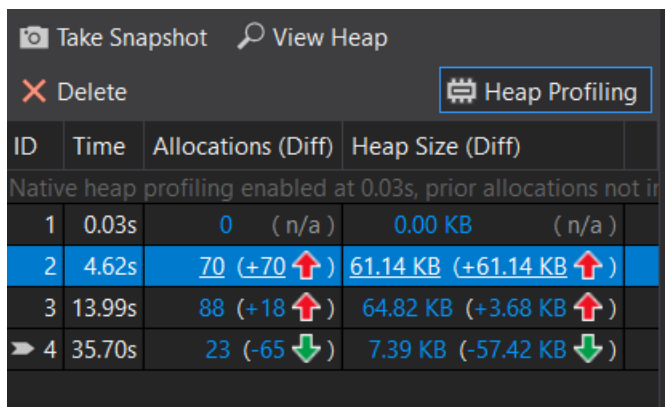
Stres testovi su vršeni tako da *Publisher* šalje 10.000 poruka servisu koji potom prosleđuje *Subscriber*-ima koji su pretplaćeni na zadatu temu. Na slici 1 može se videti celokupan rad programa tokom stres testiranja.

Testovi su rađeni uz pomoć *Windows* programa „*Performance Monitor*“. Tokom rada programa praćen broj aktivnih *thread*-ova, *handle*-ova i zauzeće memorije. Uočava se da program vremenom povećava broj *thread*-ova, *handle*-ova i memorije. Pri završetku rada se primeti opadanje iskorišćenosti resursa i ono je približno početnim vrednostima.



Slika 1. Prikaz Performance Monitor-a

Testiranje stanja *heap*-a je vršeno stavljanjem *breakpoint*-a na ključna mesta. Zauzeće memorije na *heap*-u u odabranim trenucima je prikazano na slici 2. Na 4. *snapshot*-u se može videti da na kraju programa ostane zauzeto 7KB. Istraživanjem smo došli do zaključka da je uzrok zaostale memorije upotreba sistemskih i drugih biblioteka.



ID	Time	Allocations (Diff)	Heap Size (Diff)
1	0.03s	0 (n/a)	0.00 KB (n/a)
2	4.62s	70 (+70 ↑)	61.14 KB (+61.14 KB ↑)
3	13.99s	88 (+18 ↑)	64.82 KB (+3.68 KB ↑)
4	35.70s	23 (-65 ↓)	7.39 KB (-57.42 KB ↓)

Slika 2. Snapshot heap-a

## 5. Zaključak

Što se tiče opterećenja aplikacije i procesora, maksimalno je optimizovano. Aplikacija bi potencijalno mogla da bude opterećena u slučaju rada 1000 *subscriber*-a i 1000 *publisher*-a, ali nismo u mogućnosti to da testiramo na običnim računarima.

## 6. Potencijalna unapređenja

Jedno od potencijalnih unapređenja je da se podrži neograničena dužina vesti od *publisher*-a. Veličina vesti i strukture koja se šalje je ograničena zbog korišćenja funkcije *“gets\_s”* koja kao parameter prima veličinu *buffer*-a. Optimizovano rešenje bi zahtevalo pisanje funkcije za neograničeno čitanje unosa sa tastature. Tada bi se takođe moglo podržati i učitavanje vesti iz fajla.

Naredno potencijalno unapređenje bi bilo da sa *subscriber*-ima rukuje preko IOCP mehanizma. U tom slučaju trebalo bi da se uvede protokol na nivou aplikacije pomoću kojeg bi se identifikovao tip poruke.

Takođe, aplikacija bi mogla da se unapredi tako da podržava neograničen broj *subscriber*-a. To bi se postiglo zamenom statičkog niza dinamičkim.