PUB-SUB

**1. Uvod**

Cilj projekta je razvoj distribuiranog PubSub servisa koji omogućava efikasnu razmenu podataka između proizvođača (publisher) i potrošača (subscriber). Sistem se koristi za prenos rezultata merenja jačine, napona i snage struje na različitim lokacijama gradske elektrodistribucione mreže. Proizvođači objavljuju podatke kao topike vezane za struju, napon ili jačinu i lokacije. Dok se potrošači pretplacuju na topike ili lokacije za koje dobijaju objavljene poruke.

**2. Dizajn**

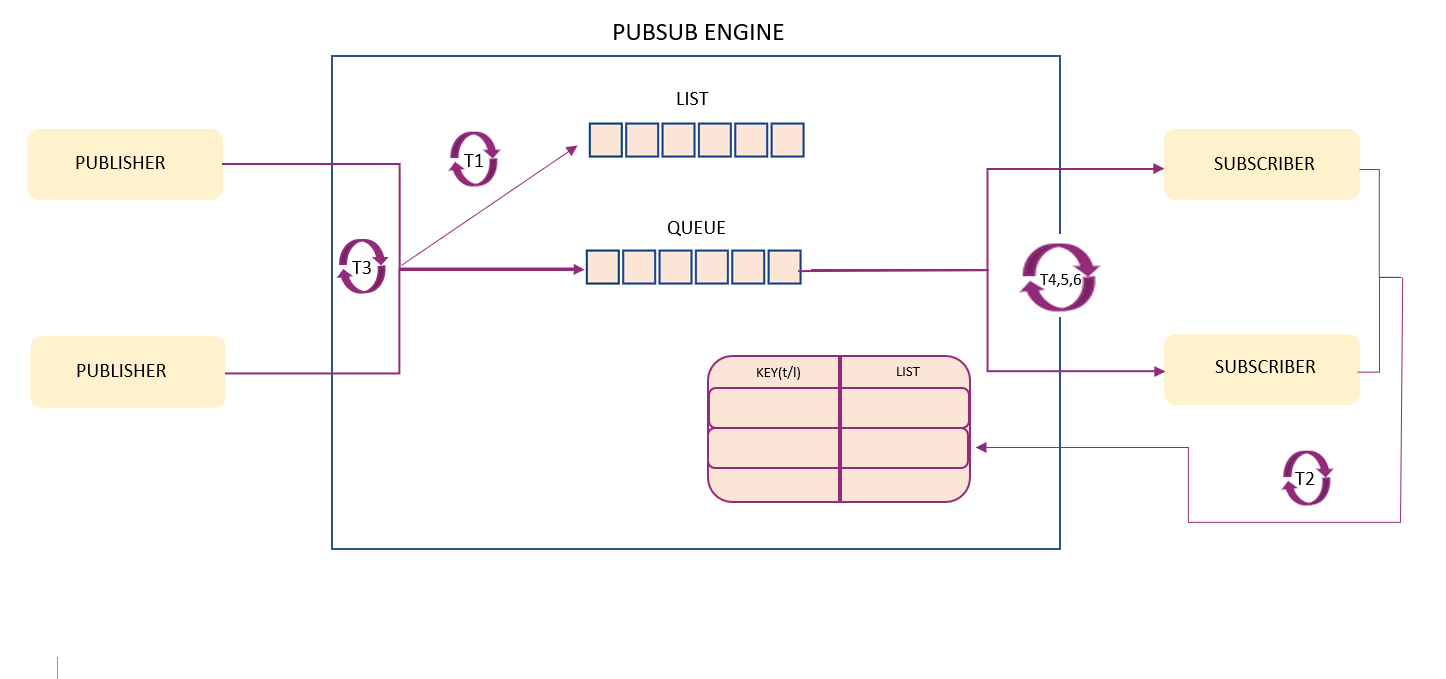
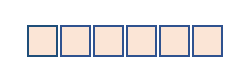
**Publisher (PB)**  
Konzolna aplikacija koja emituje poruke (artikle) koje sadrže informacije o struji: jačina, napon, snaga ili alternativno podatke o određenoj lokaciji. Svaka poruka sadrži ili topic (npr. "jačina", "napon", "snaga") ili lokaciju I izmerene podatke. Publisher se povezuje na centralni PubSubEngine i šalje poruke koje se distribuiraju subscriber-ima prema njihovim pretplatama.

**Subscriber (SB)**  
Konzolna aplikacija koja se povezuje na PubSubEngine i može da se pretplati na određene teme (topic) ili na određene lokacije.

**PubSubEngine**  
Centralni server koji upravlja konekcijama publisher-a i subscriber-a, prima poruke od publisher-a i prosleđuje ih odgovarajućim subscriber-ima prema pretplatama.

**Višethreaded arhitektura PubSubEngine-a:**

* Thread 1 – prihvata konekcije publisher-a i smešta njihove socket-e u listu publisher-a
* Thread 2 – prihvata konekcije subscriber-a i smešta njihove socket-e u listu subscriber-a, takođe proverava događaje nad prihvaćenim socket-ima subscriber-a i u slučaju pretplate na novi topic dodaje socket subscriber-a u HashTabelu, a u slučaju prekida konekcije određenog subscriber-a uklanja ga iz Liste i HashTabele
* Thread 3 – provera događaje nad prihvaćenim socket-ima publisher-a i u slučaju prijema novog artikla smešta taj artikal na Queue, a u slučaju prekida konekcije određenog publisher-a, uklanja ga iz Liste konektovanih
* Thread 4,5,6 – preuzimaju poruke sa Queue i na osnovu topica pronalaze u HashTabeli prijvaljene subscriber-e i prosleđuju im artikal.



LIST

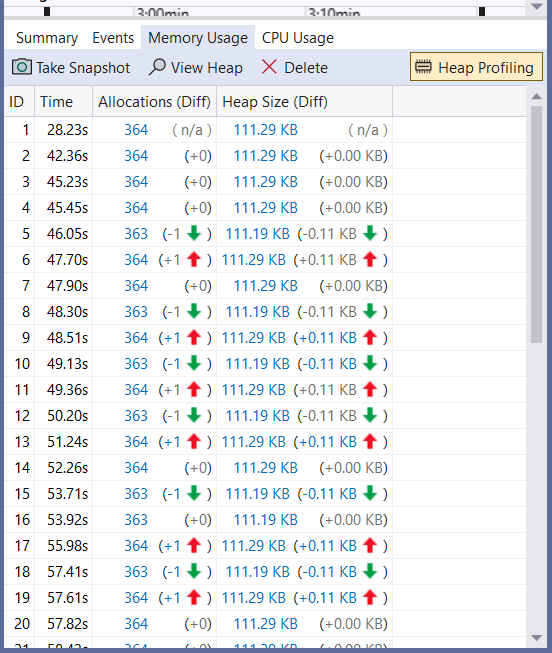
## 3. Strukture podataka

* **Queue**  
  FIFO struktura za privremeno čuvanje poruka (artikala) koje sadrže: topic, lokaciju i tekstualni sadržaj merenja. Implementirane su operacije enqueue i dequeue. U njih Thread 3 smešta poruke i nadgleda ga, a Thread 4/5/6 ih obrađuju i distribuiraju, sklanjaju iz reda.
* **List**  
  Dva primerka ove strutkure služe za čuvanje konektovanih publisher-a i subscriber-a, pomoću sturkutre tipa Uticnica koja u sebi sadrži njihov socket. Ova strukutra podataka je korišćena zbog mogućnosti pretrage, dodavanja i brisanja uz dinamičku alokaciju memorije.
* **HashTable**  
  Hash tabela u kojoj se čuvaju subscriber-i organizovani po ključevima: naziv topic-a ili lokacije. Svaki unos sadrži listu subscriber soketa koji su se pretplatili na dati topic ili lokaciju. Ova struktura omogućava brzo pronalaženje i prosleđivanje poruka odgovarajućim subscriber-ima.

## 4. Rezultati testiranja 1. Test 1: U prvom testu šalje se 50 poruka za jedan topic

## 

## 2. Test 2: U drugom testu šalje se 10000 poruka za naizmenicnim topicima



## 

## 5. Potencijalna unapređenja Sistem bi mogao da se unapredi tako što bi se uveo mehanizam koji prepoznaje kada su subscriberi postali neaktivni — na primer, ako su zatvorili konekciju ili duže vreme nisu primili poruku. Takvi korisnici bi se automatski uklanjali iz hash tabele, čime bi se izbegavalo slanje poruka na nepostojeće sokete i smanjilo nepotrebno zauzeće resursa. Pored toga, umesto da postoji jedan zajednički red poruka, moguće je uvesti posebne redove po topicu. To bi omogućilo da više tema bude obrađivano paralelno, što povećava brzinu isporuke i bolje koristi procesorske resurse, naročito kada je broj subscribera i tema veći.