SUNUCU İSTEK KONTROLÜ PROJESİ

Abdulcelil KURT , Ozan ÇELİK

KOCAELİ ÜNİVERSİTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

celilkurt82@gmail.com , celikozan1996@gmail.com

I.GİRİŞ

Bu projede sunucu istek yoğunluğunu istendiği şekilde multithread yöntemi ile kontrol edip işin alt sunucularla paylaşılmasını sağlıyoruz. Kaç alt sunucu olacağını oluşturduğumuz kontrol sunucusu sağlıyor. Gerektiği yerde Yeni sunucu oluşturuyor veya gereksiz sunucuyu durdurup siliyor. Kontrol sunucusu yoğunluğunu kontrol ettiği herbir sunucuyu işlem süresince durdurur ki kontrol edilirken sunucunun doluluk oranı değişmesin.

II.TEMEL BİLGİLER

Sunucu istek kontrolü projesi Java dilinde yazılmıştır.Geliştirme ortamı olarak "NetBeans IDE 8.2" kullanılmıştır.Grafik kütüphanesi olarak Swing kullanılmıştır.

III.ALGORİTMA

Programın başlatılmasıyla iki alt sunucu ve kontrol sunucusunu başlatılır. Sonra main'de count(int) 10000'i geçmediği sürece her 20 ms'de bir değer azaltılıp 50 ms'de bir değer artırılır. Eğer count 10000'i geçerse varolan herbir sunucu durdurulur ve program son bulur. Kontrol sunucu'su sürekli alt sunucuların doluluk oranını kontrol edip gerekli noktalarda sunucuları durdurup siler, yeni sunucu oluşturur veya sunucuların doluluk oranlarına göre önceliklerini günceller. Alt sunucular 30 ms'de bir değer azaltıp 50 ms'de bir artırır. Programın başlamasıyla gui ekranı da açılır ve sunucuların doluluk oranını gösterir.

IV.CLASSLARA AİT DEĞİŞKEN ve METHOD TANIMLARI

A.*YazLab2 Classı*

- window w1: GUI tasarımında kullanılan değişken
- int limit: Ana sunucunun alabileceği işlem sayısını tutar. Ana sunucu count'ı limit'i geçince diğer alt threadleride durdurup sildikten sonra durur.
- int rndLimit: Oluturulacak rastgele sayıların üst sınırını belirler.
- int count: Sunucudaki mevcut iş sayısını tutar.
- ArrayList<childThread> childList: Alt threadleri tutar.
- int meter: Sunucunun iç saati görevi görür.
- controlThread cThread: Kontrol thread'idir. Alt sunucuların doluluk oranlarını kontrol eder, gerektiğinde yeni sunucu açar, varolan sunucuyu kapatır ve alt sunucuların çalışma önceliğini doluluk oranlarına göre günceller.
- int getStatus():Sunucunun doluluk yüzdesini döndürür.
- void start(): Gui penceresini açar, iki alt sunucuyu ve kontrol sunucusunu tanımlayıp başlatır.
- void decrease(): [1,rndLimit/2] arasında rastgele bir sayıyı count'dan mümkünse çıkarır.
- void increase():[1,rndLimit] arasında rastgele oluşturulan sayıyı count'a ekler.
- void setProgressBar(int progress): main'in progress bar'ını günceller.

B. *ChildThread Classı*

- int count, int limit, int rndLimit ve int meter Ana sunucuna kullanıldıkları şekliyle kullanılırlar.
- boolean key: true olduğu sürece thread çalışır.
- void getStatus(),void decrease() ve void increase() fonksiyonu YazLab2 class'ındaki fonksiyonlarla aynı işi yapar.
- void start(): Oluşturulan nesne için thread oluşturup başlatır.
- void run(): Ana sunucuda biriken iş 10000'i geçmedikçe ve eğer 2 alt sunucudan biri değilse count'ı sıfırdan farklı olduğu sürece 30 ms'de bir count'u azaltıp 50 ms'de bir artırır.
- void setPriority(num): num [1,10] aralığında ise thread'in önceliğini num olarak günceller.

C.*ControlThread Classı*

- int key: true olduğu sürece sunucu çalışır.
- window w2: Ana sunucuda oluşturulan window nesnesinin kopyasıdır.
- void run(): Ana sunucu çalıştığı sürece herbir alt suncunun doluluk oranını kontrol eder. Herhangi bir alt sunucunun yüzdesi, 70'i geçerse yeni bir sunucu oluşturarak varolan sunucun işinin yarısını yeni oluşturduğu sunucuya atar, 0'sa ve sadece iki alt sunucudan biri değilse bu alt sunucuyu siler. Eğer bir alt sunucunun doluluk oranı 15'in altındaysa önceliğini 1 olarak günceller, eğer doluluk oranı 30'un üstündeyse önceliğini 9 olarak günceller.Tüm bu kontroller sırasında kontrol edilen alt sunucunun count'u değişmesin diye işlemler senkronize kod bloğunun içinde yapılır.

V. KARŞILAŞILAN PROBLEMLER ve PROBLEMLERE AİT ÇÖZÜMLER

1-Alt sunucu kontrolleri

- Alt sunucuların kontrolleri oluşturduğumuz bir başka thread olan controlThread tarafından sağlandı.Sunuculara ait durumlar anlık olarak takip edilerek doluluk oranı %70'i geçen sunucular için yeni bir alt sunucu oluşturulup,doluluk oranının yarısı kadarı yeni oluşturulan alt sunucuya devredildi.%0 değerine ulaşan alt sunucular ise arraylist yapısı olarak tuttuğumuz dizi içinden kaldırıldı.

2- Ana sunucunun dolmasını engellemek:

- Doluluk oranı %30'un üstünde olan alt sunucuların önceliğini yükselterek olabildiğince hızlı alt thread açılmasını sağladık.

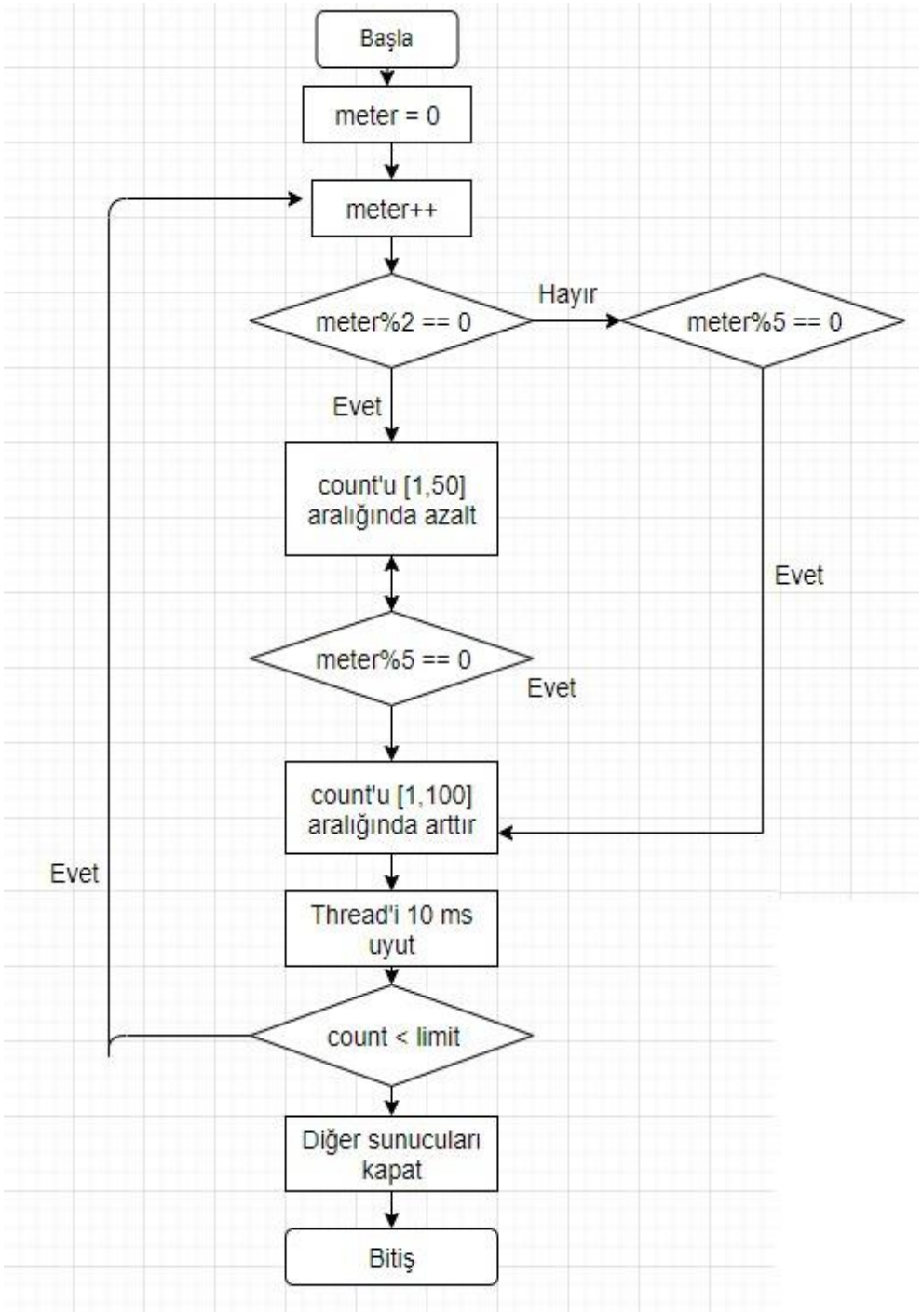
3-Senkronizasyon:

- Kontrol sunucusunda bir alt sunucunun doluluk oranını kontrol ederken yüzdedin değişmemesi için üzerinde çalıştığımız alt sunucuyu senkronizasyon kod bloğu ile başka bir sunucunun veya kendisinin bir değişiklik yapmamasını sağladık.

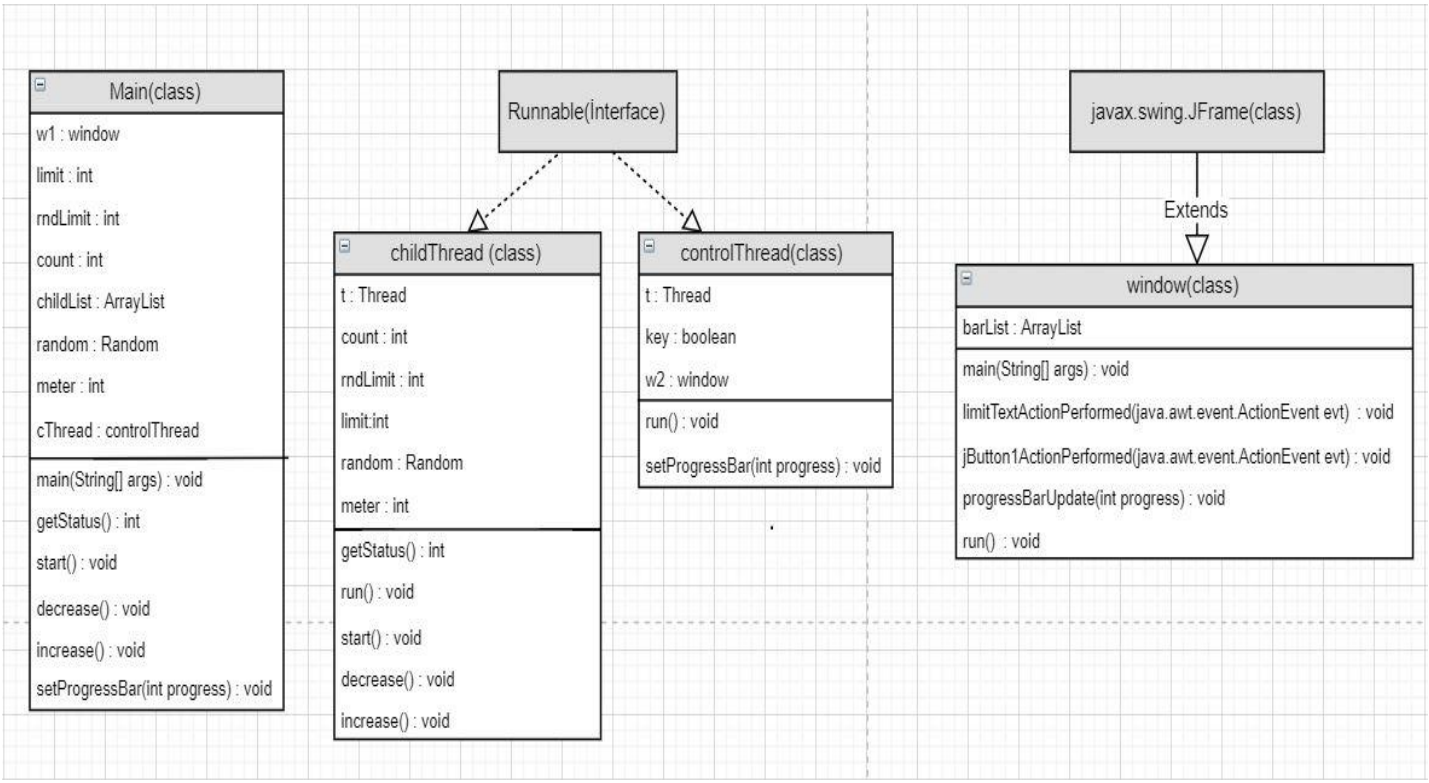
4-Olabildiğince az alt sunucu çalıştırmak:

- Doluluk oranı %15'in altına düşen alt sunucuların önceliğini azaltarak kapanmalarını hızlandırdık .

VI. AKIŞ ŞEMASI



VII. UML DİYAGRAMI



VIII. KAYNAKÇA

Thread genel:

https://docs.oracle.com/javase/tutorial/essential/concurrency/runthread.html
http://tutorials.jenkov.com/java-concurrency/creating-and-starting-threads.html
https://www.tutorialspoint.com/java/java_multithreading.htm
https://www.geeksforgeeks.org/java-lang-thread-class-
java/https://www.journaldev.com/1079/multithreading-in-java

Senkronizasyon için:

https://www.tutorialspoint.com/java/java_thread_synchronization.htm
https://javarevisited.blogspot.com/2011/04/synchronization-in-java-synchronized.html
https://www.geeksforgeeks.org/synchronized-in-java/