Sunucu İstek Yoğunluğunun Multithread İle Kontrolü

Yasir Erkam ÖZDEMİR

***Abstract*— Sunucuya gelen isteklerdeki aşırı yoğunluğun, multithread kullanılarak alt sunucularla birlikte azaltılması istenmektedir.**

1. PROBLEM TANIMI

Ana Sunucu 10000 istek kapasitesine sahiptir. 500 ms zaman aralıklarıyla [1-100] arasında rastgele sayıda istek kabul etmektedir. İstek olduğu sürece 200 ms zaman aralıklarıyla [1-50] arasında rastgele sayıda isteğe geri dönüş yapmaktadır.

Alt Sunucu 5000 istek kapasitesine sahiptir. 500 ms zaman aralıklarıyla [1-50] arasında rastgele sayıda ana sunucudan istek almaktadır. İstek olduğu sürece 300 ms zaman aralıklarıyla [1-50] arasında rastgele sayıda isteğe geri dönüş yapmaktadır.

Alt sunucu oluşturucu: Mevcut olan alt sunucuları kontrol eder. Eğer herhangi bir alt sunucunun kapasitesi %70 ve üzerinde ise yeni bir alt sunucu oluşturur ve kapasitenin yarısını yeni oluşturduğu alt sunucuya gönderir. Eğer herhangi bir alt sunucunun kapasitesi %0 a ulaşır ise mevcut olan alt sunucu sistemden kaldırılır. En az iki alt sunucu sürekli çalışır durumda kalması gerekmektedir.

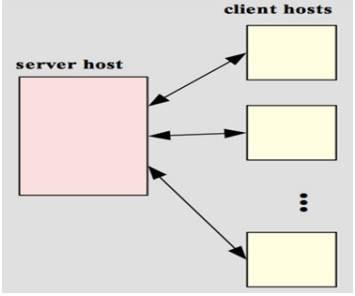
Sunucu takip: Sistemde mevcut olan tüm sunucuların bilgilerini canlı olarak göstermektedir.

Uygulama çalıştırıldığında 1 adet ana sunucu ve 2 adet alt sunucu bulunmaktadır. Ana sunucuya 500 ms aralıklarla istekler gelmektedir. Alt sunucular 500 ms aralıkla ana sunucudan istekleri alıp kendileri geri dönüş yapmaktadır. Alt sunucu oluşturucu, alt sunucuları kontrol ederek %70 ve üzerinde kapasitesi olması durumunda yeni bir alt sunucu oluşturarak kapasitesi %70 in üzerine çıkan sunucudaki isteklerin yarısını yeni sunucuya aktarmaktadır. Yeni oluşturulan alt sunucunun kapasitesi %0 a düşerse alt sunucu ortadan kalkmaktadır. Sunucu Takibinde canlı olarak sunucuların kapasite bilgileri ekrana yazdırılmalıdır.

2. YAPILAN ARAŞTIRMALAR

2.1. İstemci - Sunucu (Client - Server) Mimarisi

İstemci - sunucu mimarisi, ağ içindeki her bilgisayar ya da işlemin (process); ya istemci (client) ya da sunucu (server) olduğu mimaridir.



Şekil 1. Client server mimarisi.

İstemciler, bilgisayarlar üzerinde çalıştırılan uygulamalardır. İstemciler, sunuculardan aşağıdaki konularda istemde bulunabilirler:

- Dosyalar
- Cihazlar
- İşlemci gücü

Örneğin, en çok kullanılan istemcilerden biri e-posta istemcisidir. Size elektronik mektup gönderme ve alma olanağı sağlar.

Sunucular, ağ (network) kaynaklarını yöneten bilgisayarlardır. Fonksiyonlarına göre aşağıdaki isimleri alırlar:

- Dosya sunucuları
- Yazıcı sunucuları
- Ağ sunucuları

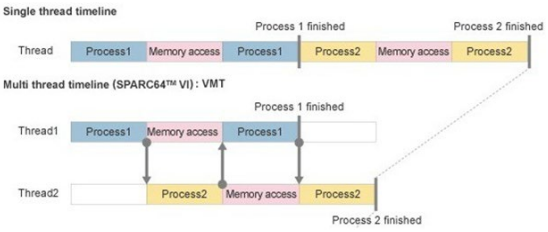
Örneğin veritabanı sunucusu veritabanı sorgularını işleyen bir bilgisayar sistemidir.

Haberleşme ağları istemci ve sunucuları birbirine bağlayan iletişim devreleridir [1].

2.2. C# ile Threading işlemleri

Threading konusu, yazılım geliştirme sürecinde çok önemli bir kavramdır. Thread'ler sayesinde birçok işi aynı anda eş zamanlı olarak yapmak mümkündür. Yürütülen iş parçacıklarını bir süre bekletmek veya istenen anda sonlandırmakta mümkündür.

Threading, aynı ortamda aynı anda birden fazla işi yapmaya denir. Thread'ler ise bu işlerin her biridir. Thread'ler aynı anda çalıştığında işlemciye process'ler olarak gider ve sıraya alınır. Tek çekirdekli işlemcilerde birden fazla thread çalıştığı zaman sırayla threadlerin processlerini işleme alır ve bir thread diğer thread'in bitişini beklemeden processleri işlemcide işleme girer. Çok çekirdekli işlemcilerde ise farklı thread'ler farklı çekirdeklerde aynı anda çalışabilme durumuna sahiptir [2].



Şekil 2. Single thread ve multi-thread.

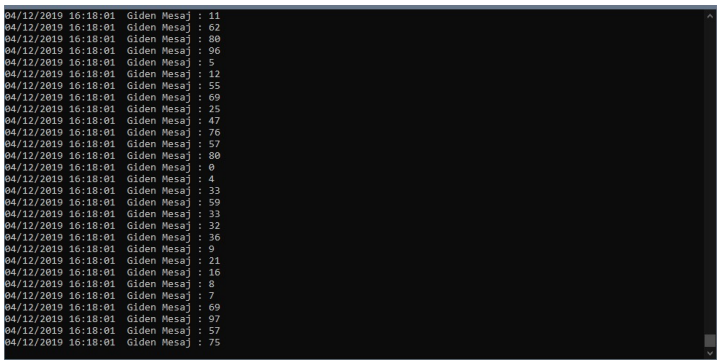
Thread'ler genel olarak, oyunlar, paralel çalışan task'ler, event handling gibi durumlarda kullanılmaktadır. Ayrıca çok çekirdekli işlemcilerde tam performans yararlanmak için de thread'ler kullanılır. İnsan üzerinden thread'leri açıklamak gerekirse, bir insan aynı anda

konuşurken gözleri başka bir yere bakıp elleriyle ayrı işler yapabilmektedir. Bu tam olarak multithreading örneğidir.

Thread'lerin kullanıldığı bir başka alan ise server-client uygulamalar. Günümüzde popüler olarak Whatsapp, forum siteleri, online oyunlar bu mimariyi kullanmaktadırlar. Birden fazla kullanıcı birbirini beklemeden işlerini halletmek durumunda olduğundan her bir kullanıcının işlemleri ayrı threadlerde yönetilmektedir [3].

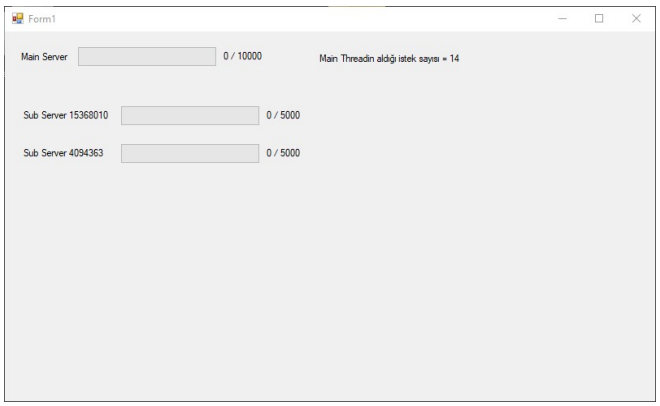
3. TASARIM

3.1. Client Penceresi

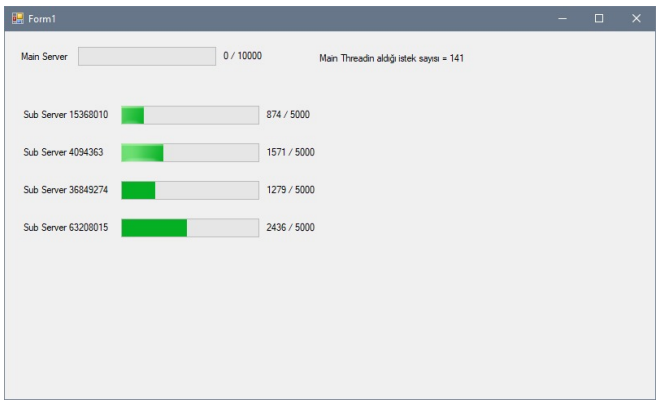


Şekil 3. Client penceresi.

3.2. Server Penceresi



Şekil 4. Server penceresi.



Şekil 5.

4. GENEL YAPI

Server/ Client diğer adıyla Sunucu/İstemci yapısı günlük hayatta veri iletimi sırasında çokça kullandığımız bir yapıdır. İnternette bir siteye bağlanırken istemeden zaten bu yapıdan faydalanırız. Veri beklenen sitenin sunucusu (Server) default olarak belirtilmiş olan 80 numaralı portu gelen bağlantı istekleri için sürekli olarak dinler.

Gelen bağlantıları diğer bir port ile meydana getirdiği soket ile sarmalar ve yönlendirir. Bundan dolayı sunucu yani server sürekli olarak 80 numaralı portu dinlemiş olur. Siteye bağlanan kişi yani client sitenin bilgi aldığı taraftaki sunucuya bağlantı isteğinde bulunur.

Eğer server client un isteğini kabul ederse , ona bir soket nesnesi açar ve farklı bir port'tan kendisiyle iletişim kurmasını sağlar. Soket yapıları tamamiyle mantıksaldır Fiziksel hiçbir elemanla ilgisi yoktur.

Server tarafında sürekli olarak bir dinleyici (listener diye bilinen) soketi bulunur. Bu dinleyici soketler yalnızca belli bir portu dinlerler. Tüm bağlantıları isteklerini bu port üzerinden kabul eder veya bunları reddeder. Bu soketler kabul ettiği bağlantılarla ilgilenmek için handler denen nesneler oluşturur ve aynı şekilde bu portu dinlemeye koyulur. Böylece sürekli olarak dinlenen port değişmemiş olur.

Tipik olarak bir soketin oluşumu aşağıda verilecektir. Burda stream şeklinde bir soket nesnesi oluşturulacaktır. Çünkü TCP/IP protokolü stream şeklinde ele alınır. Son olarak da protokolün tipi TCP olarak belirlenir.

Öte yandan listener soketinin hangi IP adresinden geleceğinin bilgisi, hangi portun dinlenmesi gerektiği belirtilmedir. Default olarak "127.0.0.1" adresi localhost IP adresidir.

5. REFERANSLAR

[1] https://gelecegiyazanlar.turkcell.com.tr/konu/web-programlama/egitim/401-node.js/istemci-sunucu-client-server-mimarisi (Access date: 20.11.2019)

[2] https://www.bayramucuncu.com/c-ile-threading-islemleri/ (Access date: 20.11.2019)

[3] https://medium.com/hesapkurdu-development/threading-in-c-hesapkurdu-f7f1e58d7c1 (Access date: 20.11.2019)