

KISIM : ARAŞTIRMA SAYFA NO: 1

YAPILAN İŞ : **UDP / TCP ARAŞTIRMA** Tarih : 18/08/2023

UDP nedir, TCP ile farkları nelerdir, bunlarla alakalı araştırmalar yaptım. **(EK1)**

UDP temelde bir haberleşme protokolüdür. UDP ile TCP’yi aynı anda çok fazla gördüm. Araştırdığım kaynaklarda ikisinin de pek çok alanda kullanıldığını gördüm. İkisinin birbirinden ayrıldığı yönleri vardı.

TCP, veri gönderiminde verinin ulaşıp ulaşmama durumuna göre geriye bir response döndürürken, UDP’de ise bu iş biraz tam tersi olup verinin gidip gitmediğini umursamadan

çalışan bir protokoldü. Bu durum aslında “TCP daha güvenilirdir.” cümlesine sık

rastlamamıza neden olan sebepti.

TCP verileri bağlantı kurarak bilgisayardan bilgisayara haberleşirken UDP’nin işlemden işleme bağlantı kurup veri gönderdiğini öğrendim.

Veri gönderme işlemleri sırasında verilerin sıralı gidip gitmeme durumu da bu protokollerde farklılık gösteriyordu. TCP’de veriler sıralı giderken UDP’de bu durum karma bir şekilde gerçekleşiyordu.

UDP, TCP’den farklı olarak “Datagram” adını verdiğimiz verileri gönderir. TCP paket göndererek haberleşir.

UDP ve TCP’nin bant genişlikleri farklıdır. TCP’nin daha fazla bant genişliğine sahip olduğunu öğrendim bu da veri akışı için daha iyi bir seçenek olmasına neden oluyor.

Ürün geliştirme seminerine katıldım ve bu seminerde problem çözme yaklaşımları ve V-model gibi ürün geliştirme metodolojileri hakkında sunum yapıldı.

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : ARAŞTIRMA SAYFA NO: 2

YAPILAN İŞ : **TCP HANDSHAKE – ENDPOINT ARAŞTIRMA** Tarih : 21/08/2023

TCP'nin iletişim başlatma ve sonlandırma (SYN-ACT flagları) işlemlerini içeren el sıkışma (handshaking) sürecini araştırdım. **(EK1)**

TCP güvenli veri iletişimi için **HANDSHAKE** dediğimiz bir yapıyı kullandığını öğrendim. Bu işlem ağ üzerinde veri iletiminin güvenliğini sağlamak amacıyla yapılıyordu.

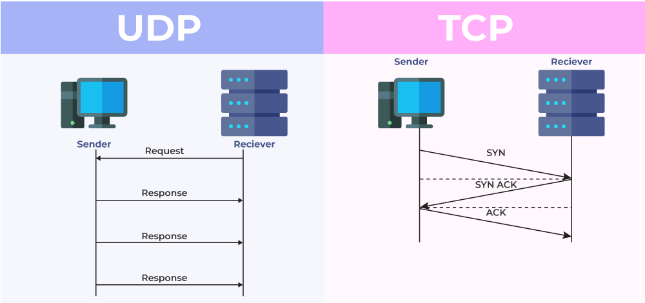
Üç adımda gerçekleşiyordu:

Birinci adımda, istemci tarafı sunucu tarafına bir iletişim sürecine girmek istediğini bildiren bağlantı isteği atıyordu, bunu da “SYN” dediğimiz bir flag ile işaretliyordu. Aynı süreçte rastgele bir iletişim numarası da gönderiyordu. Sonrasında suncudan bir cevap dönmesini bekliyordu.

İkinci adımda, sunucu istemciye yanıt verir. İstemciye bir ACK, ki bu da

aslında sunucunun bağlantı açma onayı anlamındadır, gönderir. Ekstra olarak sunucu da bir “SYN” flagi gönderir dolayısıyla sunucu için de bir iletişim numarası bilgisi gidiyor.

Son adımda istemci tarafı da bir ACK flagi ile bir bağlantı açma isteğini onayladığı zaman iki taraf tabiri caizse el sıkışır ve aslında iki taraf için de güvenli bir veri alışveriş süreci başlamış olur.



Endpoint'leri araştırdım. Ayrıca, veri iletimi için User Datagram Protocol (UDP)

tabanlı bir sunucu ve istemci uygulaması tasarlamak ve oluşturmak amacıyla bu

iletişim protokolünü ve yapısını detaylı olarak inceledim. **(EK1)**

Endpoint’in veri kaynağını veya hedefini temsil eden cihaz ya da program olduğunu öğrendim. UDP, bu endpointler arasında datagramlarını iletmek için kullanılıyordu. Endpointler sonradan da yazılımını yazacağımı şekilde içerisinde bir IP adresi ve port tutuyordu. Dolayısıyla bu şekilde hedef bildirmiş oluyoruz.

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : YAZILIM SAYFA NO: 3

YAPILAN İŞ : **UDP HABERLEŞME YAZILIMI YAPMAK**  Tarih : 22/08/2023

Client ve Server tarafında gerekli kodlamaları tamamladım. **(EK2)**

Her iki tarafın aynı bilgisayar üzerinde çalışmamasıyla ilgili bir problem ortaya çıktı, problem üzerinde araştırmalar yaptım.

Windows işletim sistemindeki güvenlik önlemleri nedeniyle ortaya çıkan kod

sorununu çözdüm. Bu çözüm sürecinde, gelişmiş güvenlik ayarları üzerinden yeni bir kural ekleyerek ve kod üzerinde gereken düzenlemeleri yaparak sorunu giderdim. (Aynı bilgisayar üzerinde server ve client aynı portta çalışmıyordu. Sebebi Windows güvenlik duvarı ve gereksiz 2 ya da 3 satır koddan oluşuyordu.)

Kontrol Eden : …………………………………………………… İmza : ………….………………



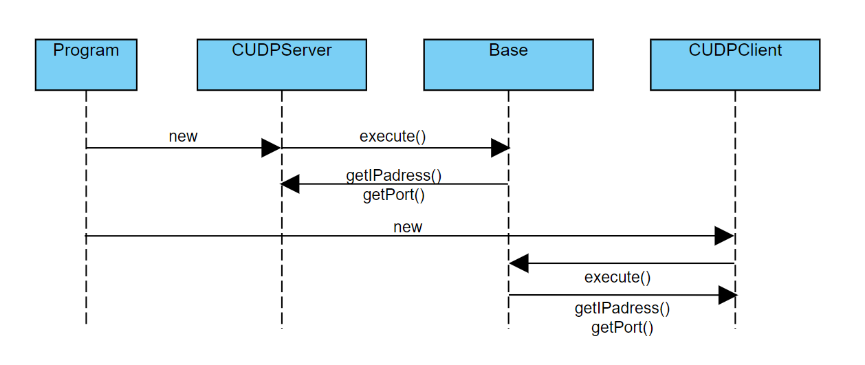
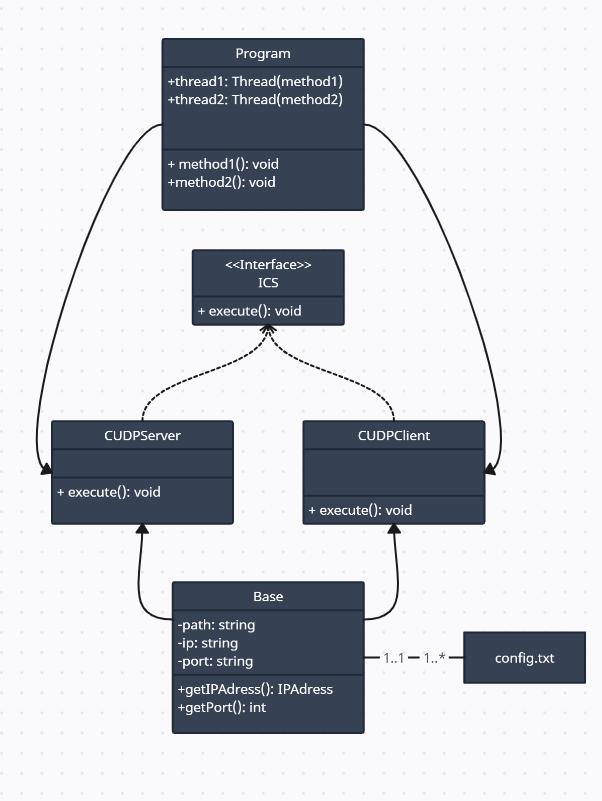
KISIM : YAZILIM VE ARAŞTIRMA SAYFA NO: 4

YAPILAN İŞ : **UML VE SEKANS DİYAGRAMLARI ÇİZMEK VS ARAŞTIRMA** Tarih : 23/08/2023

Yazdığım kod için, UML (Unified Modeling Language) diagramı çizerek, yazılım tasarımını görsel olarak temsil etmek amacıyla bir model oluşturdum. Ayrıca, sistem içindeki süreçlerin ve etkileşimlerin ayrıntılı bir gösterimi için sekans diyagramı çizdim.

Bununla birlikte, HTTP (Hypertext Transfer Protocol) protokolünü daha iyi anlamak

ve iletişim süreçlerini analiz etmek için **temel** düzeyde kısaca bilgi araştırdım. Burda temel anlamda http’nin internet üzerinden bilgi alışverişi amacıyla kullanıldığını, client-server mimarisinde çalıştığını, TCP üzerine kurulduğunu ve başlıca metotlarının “GET, POST, PUT, DELETE, HEAD, CONNECT, TRACE” olduğunu öğrendim.



Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : YAZILIM SAYFA NO: 5

YAPILAN İŞ : **STRUCT – UNION ARAŞTIRMA** Tarih : 24/08/2023

Bellekteki veri tutma yöntemlerinden biri olan ‘struct’ ile ‘union’ arasındaki farkları araştırdım. Bu araştırma sırasında özellikle ‘union’un bellekteki veri saklama durumunu inceledim. (tek adreste tutma).

Union ve struct yapısı arasında bellek ayırmada bir fark olduğunu öğrendim. Struct değişkenini depolamak için gereken bellek miktarı, tüm üyelerinin bellek miktarı toplamıdır. Ancak union depolamak için gereken bellek union yapısının en büyük üyesi için gereken bellektir.

Bunun için incelediğim örnek şu şekildeydi:

Bir çalışanın isim (char), maaş (float) ve çalışan nosunu (int) tutan union ve struct yapısı kurulmuştu. Burda sizeof() fonksiyonunu kullanarak yapıların boyutunu

incelediğimde Union’un 32, Struct yapısının ise 40 byte yer kapladığını gördüm.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Yani aslında struct yapısı tüm üyelerinin kapladığı alan kadar yer kaplarken, union yapısı en büyük yer kaplayan name değişkeninin alanı kadar yer kaplıyordu.

Union yapısında aynı anda yalnızca bir üyeye erişebildiğimizi ve geri kalan üyelerin çöp değer olarak adlandırıldığını öğrendim. Struct yapısında böyle bir durum söz konusu değildi.

Ayrıca mühendislik alanındaki bir problemi çözmek için nasıl bir yol izlendiğini mühendis bakış açısından gözlemleme fırsatı buldum (tersine mühendislik). Mühendis,

mevcut uğraştığı problemde kodun adım adım gerisine giderek hatayı buldu ben de bu durumu gözlemlemiş oldum.

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : ARAŞTIRMA SAYFA NO: 6

YAPILAN İŞ : **UART HAKKINDA ARAŞTIRMA** Tarih : 25/08/2023

UART nedir, ne işe yarar, nerelerde kullanılır, çalışma mantığı üzerinde araştırma yaptım. Doküman hazırladım. **(EK4)**

Seri ve paralel haberleşmenin ne olduğunu araştırdım. Paralel haberleşme verinin her bir bitini farklı farklı kanallardan gönderebilen bir haberleşme türüydü. Seri haberleşme ise veriler arka arkaya sıralı bir şekilde tek kanaldan gidiyordu. Dolayısıyla paralel haberleşmede hattan kayıp hızdan kazanç, seride hattan kazanç hızdan kayıp vardı; öncelikle bunları öğrendim.

Senkron ve asenkron haberleşmeyi araştırdım. Senkron haberleşmede veri ile birlikte clock sinyali gönderiliyordu, asenkronda ise clock sinyali yoktu baud rate durumuna uygun veri aktarılıyordu.

UART’ın, evrensel asenkron alıcı verici anlamına gelen bir donanım arayüzü

olduğunu öğrendim. Bilgisayar, mikrodenetleyiciler gibi cihazların arasında iletişimi sağlar. Yapısı basittir, iki kablo kullanır. Tek yönlü, çift yönlü şeklinde (full duplex, half duplex, simplex) veri aktarımı yapılabilir. Verilerin frameler şeklinde iletildiğini yapısını araştırarak gördüm. Araştırdığım haberleşme türlerinden asenkron şeklinde çalıştığını öğrendim.

UART’ın, mikrodenetleyicilerde, sensörlerde, GPS alıcılarında, bluetooth modüllerinde, Wi-Fi modüllerinde, otomotiv sistemlerinde, telemetri uygulamalarında vb. kullanıldığını; bilgisayarlarla ve diğer cihazlarla seri port üzerinden iletişim kurduğunu öğrendim.

metin, diyagram, çizgi, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturulduUART’ın çalışma prensibinde, UART 1 veri kaynağından verileri paralel haberleşme şeklinde alır kendi içerisinde start-stop bitlerini, parite bitini koyup frame haline getirdikten sonra UART 2’ye verileri seri bir şekilde iletir. UART 2, verileri seri bir şekilde aldıktan sonra start-stop bitlerini çözer, parite bitinde verinin doğru gelip gelmediğine bakar, doğru gelmişse alıcı veri kaynağına paralel haberleşme şekilde gönderir, yanlış gelmişse UART 1’e verinin yanlış geldiğini bildirir.

Sunum yapmam istendi ve sunum için bu konuyu da konu başlıkları arasına dahil

ettim.

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : ARAŞTIRMA SAYFA NO: 7

YAPILAN İŞ : **UART HAKKINDA ARAŞTIRMA** Tarih : 28/08/2023

UART ile parite bitinin kullanım mantığı, çalışma mantığı ve çeşitli farklı terimler hakkında araştırma yaptım. **(EK4)**

**ekran görüntüsü, metin, dikdörtgen, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu**

TX ve RX’in çapraz bağlandığını; TX’in, veri iletim hattı, RX’in ise veri alma hattı olduğunu; GND’nin toprak bağlantısı olduğunu ve veri iletiminde kararlılığı sağladığını öğrendim.

USART’ın UART’tan daha gelişmiş olduğunu ama çalışma mantıklarının aynı olduğunu öğrendim. USART, UART’tan farklı olarak ekstra senkron haberleşme de yapabiliyor. Yeni çıkan teknolojilerde daha çok bu kullanılırken, uzun mesafe iletimlerde RS232 gibi teknolojiler kullanılabiliyor.

metin, ekran görüntüsü, yazı tipi, sayı, numara içeren bir resim

Açıklama otomatik olarak oluşturuldu

Verilerin bir bitini start, bir bitini stop bit olarak sayarsak bir byte veri gönderebilmek için aslında 10 bit göndermemiz gerektiğini yandaki şekilden yola çıkarak söyleyebiliriz.

UART’ta verilerin doğru gelip gelmediğini kontrol edebildiğimiz bir mekanizmanın olduğunu gördüm. Bunu parite bitlerini kullanarak yapıyorduk. UART’ı tek parite mi çift parite

mi şeklinde ayarladıktan sonra, verilere tek parite ise “1”, çift parite ise “0” değerini atadığımızı gördüm. Gönderdiğimiz verideki bitlerde kaç tane bir olduğunu sayıp, örneğin tek parite ise ve tek sayıda bir sayısı olduğunu görüyorsak parite bitinin de “1” olmasından dolayı verinin aslında doğru gittiğini anlayabiliyorduk. Çift paritede durum tam tersi oluyordu. Tabii bu durumda birden fazla veri bozulursa parite bitiyle güvenlik aslında pak fazla sağlanamıyordu, bu da parite mantığının çok güvenilir olmadığını gösteriyordu.

Kontrol Eden : …………………………………………………… İmza : ………….………………

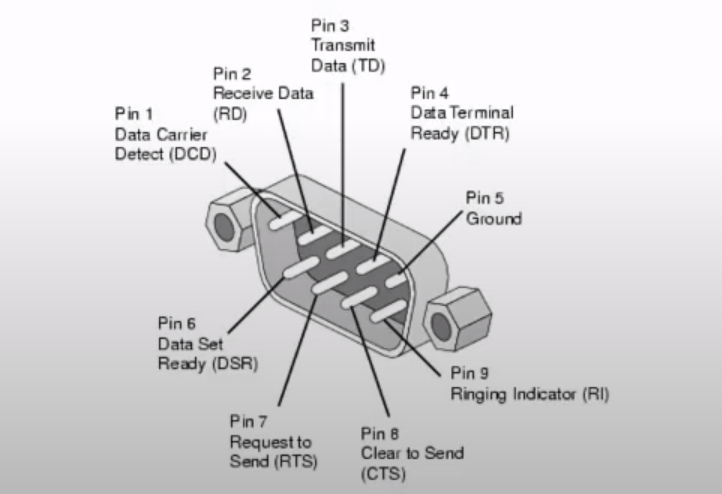


KISIM : ARAŞTIRMA SAYFA NO: 8

YAPILAN İŞ : **UART HAKKINDA ARAŞTIRMA - SOLID PRENSİPLERİ ARAŞTIRMA** Tarih : 29/08/2023

Uart’ın avantajları ve dezavantajları hakkında araştırma yapmaya devam ettim. Avantajları arasında; basit ve yaygın kullanılabilir olması, seri iletişim kullanıyor olması, asenkron çalışması (tabii burda iki tarafın da baud rate oranının aynı olması şartıyla), çok sağlıklı olmasa da hata ayıklama mekanizmasına sahip olması, donanım ve yazılım desteklerinin olması, düşük maliyet, veri aktarım mesafesinin RS232 gibi teknolojilerle arttırılabilmesi, gerçek zamanlı, iyi belgelenmiş ve paket yapısının iki taraftan da ayarlandığı sürece rahat bir şekilde kullanılması vardı. Dezavantajları arasında; frame boyutunun 9 bit ile sınırlı olması, birden çok ana sistemi desteklememesi, baud rate’in aşağı yukarı yüzde 10

kadar değişebilmesi ve seri iletim kullanıldığından iletimin yavaş gerçekleşmesi vardı.

 Yandaki RS232 kullanılarak veri iletim yolunu uzatabildiğimizi öğrendim. Fakat bu teknoloji artık eskidiği için yerine daha farklı çözümler getirilmiş. Artık yeni bilgisayarlarda bu giriş yok peki bunun yerine ne kullanılıyor diye araştırdığımda Uart-USB Converter’lar olduğunu gördüm. Bunları kullandığımızda bilgisayarda sanal bir COM Port oluşuyor ve biz bu şekilde mikrodenetleyicilerde iletişim kurabiliyoruz.

Solid prensipleri nedir, ne için kullanılır, kullanılmazsa ne olur gibi genel bir araştırma yapmaya başladım. Yapacağım sunum için Single Responsibility, Open Close, Liskov Substition, Interface Segregation ve Dependency Inversion başlıklarını detaylı bir şekilde inceleyip kod örnekleriyle birlikte sunum şeklinde hazırladım. **(EK5)**

SOLID prensiplerinin amacının yazılımda sürdürülebilirlik olduğunu, geliştirdiğimiz kodların uyumunun sağlıklı bir şekilde uyarlandığı, az müdahaleyle istenilen işleri yapabildiğimiz gibi gereksinimleri amaçlayan prensiplerdir.

Single Responsibility’nin her fonksiyonun veya classın kendi amacı dışında kullanılmaması gerektiğini; Open Close’un, kodların geliştirmeye açık değiştirmeye kapalı bir amacı hedeflediğini; Liskov Subsitition’un aynı sınıftan kalıtım alan ancak aynı özelliklere sahip olması istenmeyen sınıfların nasıl tasarlanacağını; Interface Segregation’un farklı özellikler için farklı interface’ler oluşturulmasını gerektiğini; Dependency Inversion’un ise üst sınıfların alt sınıflara bağlı kalmaması gerektiğini amaçladığını öğrendim. Bunlarla ilgili

birçok örneği inceleyip sunumda da anlattığım slaytı **eklerde** bulabilirsiniz.

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : ARAŞTIRMA SAYFA NO: 9

YAPILAN İŞ : **SUNUMA HAZIRLIK VE GENEL ARAŞTIRMA** Tarih : 31/08/2023

Şu ana kadar araştırmalarım hakkında genel tekrar yaptım.

elektronik donanım, elektronik bileşen, devre bileşeni, elektronik mühendisliği içeren bir resim

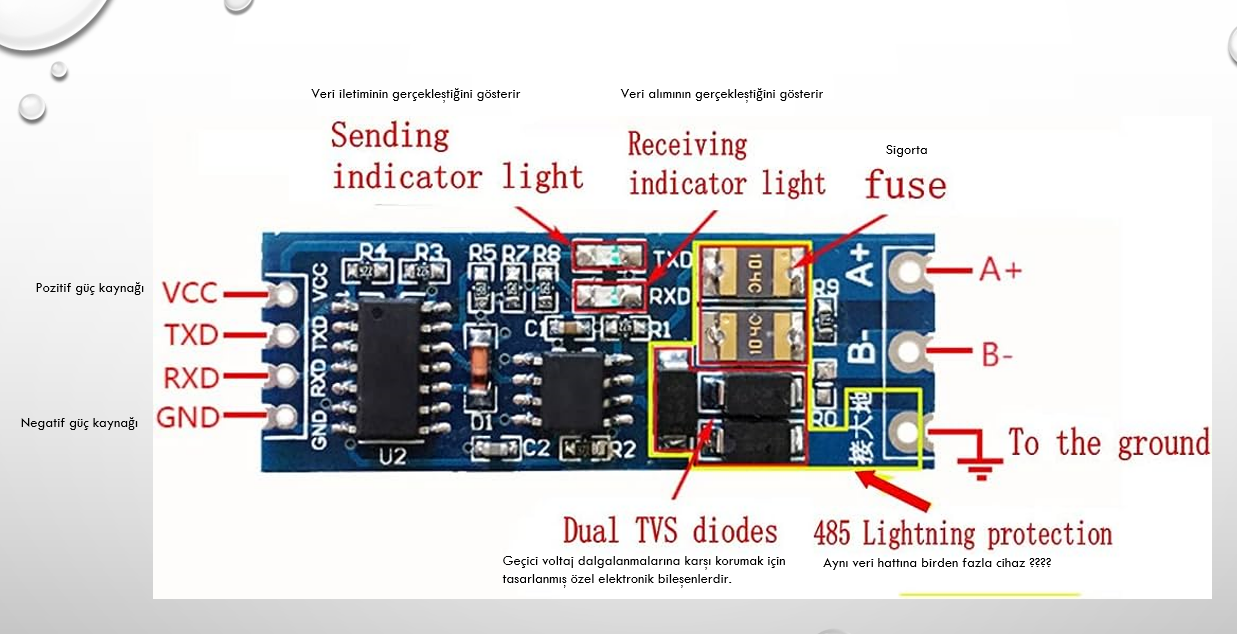
Açıklama otomatik olarak oluşturuldu UART haberleşme setup’ını inceledim. Soldaki haberleşme örneği görsel açıdan bir “Haberleşme nasıl olur?” sorusunun güzel bir somut örneğini bana verdi.

UART’ta topladığım bilgileri sunum tarzında yeniden düzenledim. Buffer, flow

kontrol gibi bazı terimleri araştırdım. **(EK6)**

Buffer’ın, UART’ta kullanılan veri depolayan kısım olduğunu; flow (akış) kontrolün yazılımsal tarafında XON ve XOF karakterlerinin haberleşmeyi başlattığını ve durdurduğunu, donanımsal tarafta ise RTS / CTS hatları kullanıldığını öğrendim. RTS ve CTS çalışma mantığını, TCP tarafındaki Handshake mantığıyla hemen hemen aynı buldum.

TTL’in ise mikrokontrolcünün uart ile gönderip aldığı sinyalin adı olduğunu öğrendim.



Yazılım koordinatörüm tarafından “Whatsapp UDP mi kullanır TCP mi ?” araştırması istendi. Bunun sonucunda Whatsapp’ın her ikisini de kullandığını; TCP’nin Noise Pipes Protocol ile şifrelenmiş mesajları içerdiğini; UDP tarafında ise aramaların RTP (Real Time Transport Protocol) ile gerçekleştiğini, SRTP (Secure Real Time Protocol) ile de bu

protokolün şifrelendiğini öğrendim.

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : ARAŞTIRMA SAYFA NO: 10

YAPILAN İŞ : **DESIGN PATTERNS GİRİŞ** Tarih : 01/09/2023

UDP kodunda gönderilmek istenen veri kullanıcıdan alınsın diye değişiklik yaptım.

Design patterns ile ilgili sunum için araştırma yapmaya başladım. **(EK8)** Araştırma yaptıkça Design patterns’ın SOLID prensipleriyle baya uyumlu çalıştığını gördüm. Yazılım projeleri büyüdükçe ortalık biraz karışmaya başlıyor, bazı durumlar işin içinden çıkılmaz hal alıyor, problemlerin çözümü zorlaşıyor. Dolayısıyla tasarım kalıplarının yazılımın gelecekteki

gidişatı üzerinde inanılmaz bir etkisi olduğunu gördüm. Bu desenler, belirli durumlarda çeşitli kullanımlar sunarak çözümler verir. Kodları anlaşılır kıldığını ve bakımlarının kolaylaşmasını sağladığını öğrendim. Bunu yazılımda bir kılavuz olarak algıladım.

Araştırdığım kaynakta 23 tasarım kalıbı olduğunu gördüm, bunlar üçe ayrılmıştı: Creational, Structural, Behavioural.

Creational, nesne üretimi üzerinde duruyordu; Structural, nesnelerin sınıflara uyarlanması üzerinde duruyordu; Behavioural ise nesnelerin birbiriyle iletişimi ile ilgileniyordu.

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : ARAŞTIRMA SAYFA NO: 11

YAPILAN İŞ : **SEMİNER VE DESİGN PATTERNS ARAŞTIRMA DEVAM** Tarih : 04/09/2023

“Hava Savunma Sistemleri Bilgilendirmesi ve Test Videolarının İzletilmesi” Programına katıldım. SAGE’nin temelde nelerle uğraştığından (mühimmat), kimlerle çalıştığından (Roketsan, Aselsan) tarzında bilgiler verildi. “Siper” hava savunma sisteminin

ilk defa ateşlendiği video izletildi.

Singleton, prototype, factory method, adapter ve builder üzerinde araştırmalar yapıp kod örneklerini inceledim. **(EK8)**

Singleton’ın, nesne oluşturmayı tek sefer için yaptığını; Prototype’in, nesnelerin kopyalanmasını sağladığını; Factory Method’un, birden fazla classın tek class üzerinden nesne oluşmasını sağladığını; Builder’ın, constructor kullanırken her değişkeni değil de bazı değişkenlerin değerini vermemizi sağladığını; Adapter’ı kullanarak uyumlu olmayan sınıflar için sınıflar yazıp senaryo durumuna uyarlandığını; Composite ile bir container tanımlayıp birden fazla classı tek bir yerden kontrol etmeyi ve son olarak Observer ile bir nesnenin diğer nesneler tarafından haberdar edilmesi tasarımını öğrendim. Bunların kod incelemelerini gerçekleştirdim. Bunların hepsini de bir powerpoint dosyasında sunum şeklinde topladım. (Eklerde bulunabilir.)

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : ………………………………………………………..………………… SAYFA NO: 12

YAPILAN İŞ : DESIGN PATTERNS’TA COMPOSITE VE MULTIPLECLIENT Tarih : 05/09/2023

Design patterns’ta composite araştırmasını tamamladım. **(EK8)**

MultipleClient UDP nasıl çalışır hakkında araştırma yaptım.

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : YAZILIM SAYFA NO: 13

YAPILAN İŞ : **MULTIPLECLIENT KODU YAZMAK** Tarih : 06/09/2023

MultipleClient UDP kodunu yazdım. **(EK9)** Base classında dosyada yazan ip ve port bilgilerini aldım. Bunları bize veren iki tane getter metodu yazdım. MultipleClient classında sonsuza kadar sayı yazdıran ve bunun verisini byte’a çevirip send fonksiyonu ile gönderen bir execute fonksiyonu yazdım. Server classı’nda da execute adlı bir fonksiyon yazdım ki bunları ICS’den implement ettim. Burda da gelen verileri yakalamayı amaçladım. Program ana classımda da birden fazla thread kullanarak MultiClient classının execute fonksiyonunu eş zamanlı olarak çağırdım.

Çoklu clientlerde milisaniyeler ile yazdırılan verilerde sunucuda kayıplar yaşadım. Bu problem üzerinde çalıştım.

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : YAZILIM SAYFA NO: 14

YAPILAN İŞ : **ACK MEKANİZMASI YAZMAK - STRUCT YAPISI GÖNDERİP ALMAK** Tarih : 07/09/2023

UDP MultiClientta TCP’de olduğu gibi ACK mekanizması yazmaya çalıştım, sonuç başarısız oldu. Fakat en başından beri denediğim fakat kabul edilmeyen 1 ms bekletme (threadle uyutma) kabul edildi dolayısıyla burdaki görevimi tamamladım.

Üç tane struct yapısını UDP MultiClient ile aktarma ve alma görevi aldım.

Marshal yapısını kullanarak bu görevi yapabilmenin yolunu buldum. Farklı threadlerle verileri gönderip almak için çalışmaya başladım. **(EK10)**

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : YAZILIM SAYFA NO: 15

YAPILAN İŞ : **KOD YAZIP GÖZLEMLEMEK VE SEMİNER** Tarih : 08/09/2023

Bir gün önce yazdığım kodun verilerini farklı threadlerle gönderdim veri kaçırıp kaçırmadığını gözledim.

Farklı bir struct yapısında aynı mantığın çalışması istendi ve sonuç başarılı. **(EK11)**

Kıdemli Başuzman Araştırmacısı Sn. Serhat VARIŞ tarafından düzenlenen “**Genel Kompozit Eğitimi**” seminerine katıldım. Parçaları nerde bastıklarını gezdirdikleri bir seminerdi (3D yazıcılar, SLS teknolojisi).

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : SUNUM SAYFA NO: 16

YAPILAN İŞ : **SUNUM YAPMAK VE SEMİNERLER** Tarih : 11/09/2023

Design patterns, SOLID prenciples, UART, UDP/TCP konularını içeren uzun bir süredir hazırlandığım sunumu gerçekleştirdim. Bu zamana kadar araştırdığım tüm konular hakkında bir powerpoint sunusu hazırlamaya çalışmıştım. Vakti geldiğinde tüm Yazılım Mimarisi Tasarımı birimi olarak toplantı odasına geçip orada sunumumu gerçekleştirdim. Göktuğ takım liderimiz sunumumu dinleyip bitirdikten sonra kendi bilgilerini ekleyerek

anlattıklarımın üzerinden geçti. Verimli bir sunum oldu.

Mülakat Simülasyonu seminerine katıldım. Bir mülakata girdiğimizde sorulan sorulara ne şekilde cevap vermeliyiz, tavrımız net mi olmalı yoksa arada kalır bir tavır mı

sergilemeliyiz tarzında içeriği olan, stajyer olarak içimizden seçilen kişiler üzerinde

simülasyon yapıldı.

Kıdemli Başuzman Araştırmacısı Sn. Kenan ÜNAL tarafından düzenlenen “**Mühimmat ve Füze Sistemleri Tasarımında Uçuş Mekaniğinin Rolü**”eğitimine

katıldım. Füzelerin manevraları, hedef kitlendikleri boyuncaki hareketleri üzerinde konuşuldu.

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : ARAŞTIRMA SAYFA NO: 17

YAPILAN İŞ : **ORYANTASYON – DEBUG YAPMAYA GİRİŞ** Tarih : 12/09/2023

Oryantasyona katıldım. Mühimmat olan Bozok, SOM/J, KGK, HGK, NEB gibi birçok mühimmatın tarihsel çizgide dinletisi gerçekleştirildi. Mühimmatların olduğu odaya gidip inceleme yaptık.

İş sağlığı güvenliği hakkında sunum yapıldı. Genel olarak SAGE’de nasıl aday mühendis olarak çalışmaya başlanır üzerinde konuşuldu.

Bir gün önceki yaptığım sunumda debug üzerinde bilgi eksikliğim fark edilmişti. Bundan dolayı debug üzerinde araştırma yapmam istendi.

Debug nedir, neden yapılır, nasıl yapılır üzerinde araştırma yapmaya başladım. **(EK12)**

Debug ile hata ayıklama sürecini kolaylaştırabildiğimizi gördüm. Hatalar yazım hataları (noktalı virgül) veya mantık hatalarından çıkabildiği gibi debug asıl mantık hatalarında işe yarıyordu. Breakpoint kullanarak programı çalışmaya başlattıktan sonra istediğimiz yerde durdurabiliyoruz. Durduğumuz yerde step over, step into, step out gibi kontrol

mekanizmalarını kullanarak bir sonraki adımdaki kontrolümüzü sağlayabildiğimizi öğrendim. Conditional breakpoint kullanarak sadece belirli durumlarda progremın durmasını sağlayabildiğimizi gördüm: “for döngüsündeki i değişkenini 330 olunca dur” gibi.

Değişkenleri görebildiğimiz bazı pencerelerin bulunduğunu gördüm. Locals’de debug aşamasında hangi bloktaysanız o bloğun değişkenlerinin değişikliklerinin o pencerede gösterildiğini, Autos’da da aynı mantık hakim olsa da Locals gibi blok içi değil o an hangi değişken değişiyorsa onun takibinin yapıldığını, Watch ile tamamen değişken takibinin bize bırakıldığını öğrendim.

Immediate penceresinde kod yazıp değişkene müdahale edebildiğimizi, Call Stack ile hangi fonksiyonun nerden çağırıldığını, Thread Window ile thread takibini yapabildiğimizi öğrendim.

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : YAZILIM SAYFA NO: 18

YAPILAN İŞ **: PRATİKTE DEBUG NASIL YAPILIR ?** Tarih : 13/09/2023

Breakpoint, conditional breakpoint, Windows (locals, autos, watch, immediate), call stack ve thread window anahtar kelimeleri üzerinde araştırmalar yapıp slayt hazırladım. **(EK12)**

Bunlar üzerinde pratik yapabilmek için basit kod örnekleri üzerinde çalıştım.

metin, ekran görüntüsü, yazılım, web sayfası içeren bir resim

Açıklama otomatik olarak oluşturuldu

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : YAZILIM SAYFA NO: 19

YAPILAN İŞ : **UART HABERLEŞME İÇİN SANAL PORT AÇMAK** Tarih : 14/09/2023

Debug işlemlerinin kod üzerinde incelemelerimi bitirdim.

UART’ta iletişim için kod yazmam istendi. Bunun için normalde donanıma (sensör vs) ihtiyacım vardı fakat benden donanım olmadan bir haberleşme kurmam istendi. Dolayısıyla ben de bilgisayardan nasıl sanal port açıp veri gönderebileceğimi araştırırken **com0com**

adında bir program buldum.

Kullanmam gereken SerialPort sınıfını .NET 3.1 kullandığım için göremiyorum.

Bunun üzerinde çalıştım (.NET 6.0 kullanmam genel anlamda istenmiyor).

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : YAZILIM SAYFA NO: 20

YAPILAN İŞ : **SANAL PORT AÇMAK VE YÖNLENDİRMEK – VERİ GÖNDERMEK** Tarih : 15/09/2023

.Net framework’ü 3.0’a güncelleyince SerialPort sınıfını tanıdı.

Serialport’a port numarasını vs girdiğimde hata “sadece windowsta desteklenir” tarzında hata aldım. Problem bu sefer de 3.0 kullanmaktan kaynaklanıyor.

Veri gönderip almaya çalıştım. Aynı port üzerinde iki farklı bağlantı gerçekleştiremedim bunun çözümünü araştırdım.

Port yönlendirmenin mümkün olduğunu öğrendim, port yönlendirmeye çalıştım.

Port yönlendirmeye gerek yokmuş, kullandığım com0com yazılımı zaten bunu yapıyormuş. Yazdığım kodda “writeline” yerine “write” kullanmışım veri alamama sebebim buymuş. Kod çalıştı. **(EK13)**

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : ………………………………………………………..………………… SAYFA NO: 21

YAPILAN İŞ : UARTTA STRUCT VERİSİ GÖNDERME Tarih : 18/09/2023

Bugün Uart iletişimde UDP’de yaptığım gibi struct yapısı gönderme görevi aldım. Bunu observer design çerçevesinde yapmam istendi. Trendyol mantığıyla çalışacak, ürün

fiyatı 1 saniyede bir değişecek, dinlenen veri ile de bir grafik çizdirmem gereken bir projeye başladım.

Trendyol1 ve Trendyol2 diye proje açtım. Uart ile structı byte array olarak karşı tarafa iletebildim ama karşı taraftan bu veriyi alıp structa çeviremedim, System.AccessViolationException hatası aldım. **(EK14)**

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : ………………………………………………………..………………… SAYFA NO: 22

YAPILAN İŞ : LITTLE ENDIAN AND BIG ENDIAN ARAŞTIRMA Tarih : 19/09/2023

Sorunu çözemedim, bu yüzden projeye devam etmeden önce little endian ve big

endian kavramları üzerinde araştırma yapmam ve bunun sunumunu hazırlamam istendi. **(EK15)**

Araştırmamı yaptım. Sorunu çözecek bir yardımı dokunmadı.

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : ………………………………………………………..………………… SAYFA NO: 23

YAPILAN İŞ : AccessViolationException ÇÖZÜMÜ – PROGRAMA DEVAM – PLOTLAB ARAŞTIRMA Tarih : 20/09/2023

AccessViolationException sorununu çözdüm. Uart biraz ilkel bir haberleşme

protokolü olduğu için stringi struct yapısı içinde görünce server tarafı patlıyormuş. String yerine int bir değer koyup çalışmaya devam ettim. **(EK14)**

Observer design patterni programa entegre ettim. Bir yayınlayan bir de dinleyici arayüzlerini entegre edip gerekli fonksiyonları yazdım. Mitov.com’un PlotLab ürününü nasıl projede kullanabilirim onu araştırdım. **(EK14)**

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : ………………………………………………………..………………… SAYFA NO: 24

YAPILAN İŞ : PLOTLAB ÜRÜNÜ İÇİN UYGUN VS SÜRÜMÜNÜ KURMAK Tarih : 21/09/2023

PotLab, visual studio 2022’de destek vermediği için 2017 sürümünü indirdim. 2017 sürümünde de eklentiyi kuramadığı için 2019’u indirdim. 2019’a eklentiyi kurdum fakat bu sefer de proje çalıştırmakta problem çıkardı.

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : ………………………………………………………..………………… SAYFA NO: 25

YAPILAN İŞ : PLOTLAB’I UART HABERLEŞME OLMADAN YENİ PROJEDE KULLANMAK Tarih : 22/09/2023

VS 2019’u kaldırıp baştan kurdum. Console kısmında bir şey değişmedi ama form kısmı çalıştı.

Trendyol1 ve Trendyol2 projemi tek proje içinde UART olmadan observer

design ile yapmam istendi ve yaptım. **(EK16)**

PlotLab dokümanını okuyarak gerekli DLL dosyalarını proje içerisine aktardım. Scope ile Waterfall bileşenlerini kullanarak grafik üzerinde veri akışı sağlayabiliyor muyum diye test ettim ve sonuç başarılı.

ekran görüntüsü, multimedya yazılımı, yazılım, grafik yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin, ekran görüntüsü, yazılım, multimedya yazılımı içeren bir resim

Açıklama otomatik olarak oluşturuldu

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : ………………………………………………………..………………… SAYFA NO: 26

YAPILAN İŞ : VERİLERİ GRAFİK ÜZERİNDE GÖSTERMEK Tarih : 25/09/2023

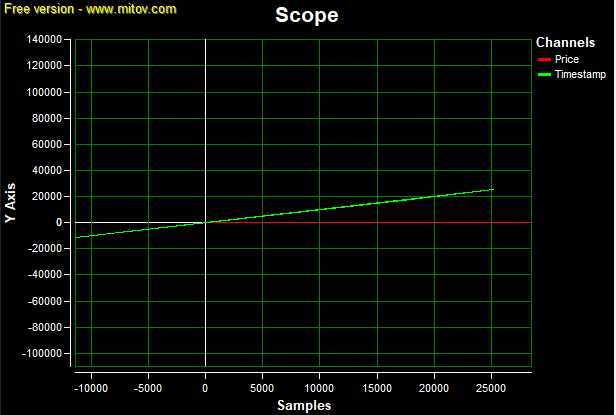
Random veri yazıp observer design mantığıyla aldığım bilgileri PlotLab yazılımı üzerinde göstermeye çalıştım. Hemen yapamadım, kütüphaneyi bir süre incelemeye çalıştım. Tek bir veriyi gösterebiliyorum fakat bizim elimizde olan senaryo bir ürünün hem fiyatını hem timestamp bilgisini x ve y koordinatları üzerinde göstermek. Bunun için çalışmaya devam ettim. Ayrıca kodda problemler observer designım pek sağlıklı değil veri yazdırma işini çözdükten sonra bu işle ilgilenmem gerekecek.

Tek bir veriyi gösterebildiğim grafik:

metin, ekran görüntüsü, diyagram, öykü gelişim çizgisi; kumpas; grafiğini çıkarma içeren bir resim

Açıklama otomatik olarak oluşturuldu

İki veriyi göstermeye çalışınca elde ettiğim görüntü:



Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : ………………………………………………………..………………… SAYFA NO: 27

YAPILAN İŞ : VERİLERİ GRAFİK ÜZERİNDE GÖSTERMEK Tarih : 26/09/2023

Grafik üzerinde gösterme işine bakmaya devam ettim. Daha önce de yapabildiğim gibi product.price değişkenini grafikte gösterebiliyorum (Y ekseninde) ama X için bir düzenleme fonksiyonu mevcut değil.

Price değişkenini koyabildiğim fonksiyon scope1.Channels[0].Data.AddYData(buffer1.ToArray()); kodu. Aynı şeyin Y versiyonu yok.

scope1.Channels[1].Data.AddXYData(buffer1.ToArray(), buffer2.ToArray());

Şeklinde bir fonksiyon yazılmış ama bu kodu yazınca da aşağıdaki şekildeki gibi garip şekiller çizdirdi.



PlotLab kütüphanesinin AddYData gibi fonksiyonlarının hangi parametreleri

aldıklarını inceledim.

Ekstra olarak Get ile başlayan fonksiyonlar var anladığım kadarıyla bunlar grafikten

veri çekiyor. Set ile başlayan fonksiyonlar var, olan veriyi düzenlemek için yazılmış fonksiyonlar. Dolayısıyla benim Add ile başlayan fonksiyonlara odaklanmam gerekiyordu.

Mühendisle birlikte durumu değerlendirdik, yazılımı biraz inceledik. Kaynak yetersizliği konusunda hem fikirdik. Bir örnek bulduk X koordinatına tarih bastıran. İstediğimiz kaynak buydu.

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : YAZILIM SAYFA NO: 28

YAPILAN İŞ : **VERİLERİ GRAFİK ÜZERİNDE GÖSTERMEK** Tarih : 27/09/2023

Bulduğumuz örneği inceledim. Kodları biraz taklit ettim fakat X ekseninde tarih yazdıramadım. Çıktı bir süre hep aşağıdaki gibi kaldı:

metin, ekran görüntüsü, ekran, görüntüleme, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

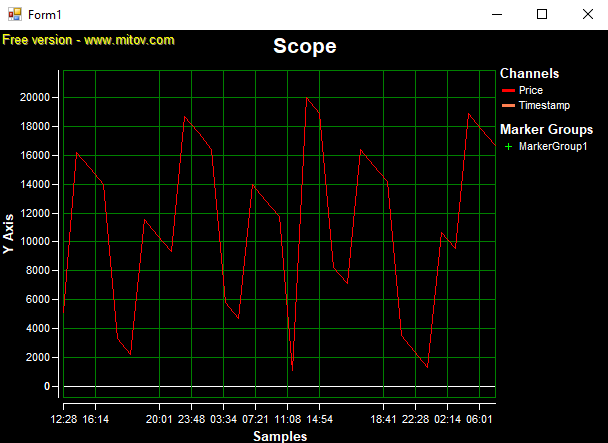
Grafiğin böyle olmasını sebebini buldum. scope1.Channels[0].Data.AddXYPoint(observer.updateTimestamp().ToBinary(), observer.updatePrice()); yazdığım bu kod aslında bana son güncellenen veriyi getirmiyormuş. Dolayısıyla daha önce bahsettiğim kod düzeni problemime geri dönmek durumunda kaldım. Veriyi form sınıfı içerisinde bastırıyorum bunu observer sınıfı içerisinde yaptırmam gerekiyordu. Bu hem SOLID prensipleri açısından yapılması gereken bir şeydi

hem de benim doğru verime ancak bu şekilde ulaşabilirdim.

Grafik bir süre yukarıdaki şekilde kaldı verinin doğru gelmesine rağmen. Onun da sebebi gelen tarih verisi hep aynı gün olduğu içinmiş. Ben de artık en başta bir tarih verisi gönderip bu gönderdiğim veriye her seferinde bir gün ilave ettim. Dolayısıyla artık aşağıdaki grafik görünümüme ulaşmış oldum.

metin, ekran görüntüsü, ekran, görüntüleme, öykü gelişim çizgisi; kumpas; grafiğini çıkarma içeren bir resim

Açıklama otomatik olarak oluşturuldu



Yazılımı SOLID prensiplerine uygun değiştirmeye çalıştım.

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : ………………………………………………………..………………… SAYFA NO: 29

YAPILAN İŞ : …………………………………………...………………… Tarih : 28/09/2023

Kontrol Eden : …………………………………………………… İmza : ………….………………



KISIM : ………………………………………………………..………………… SAYFA NO: 30

YAPILAN İŞ : …………………………………………...………………… Tarih : 29/09/2023

Kontrol Eden : …………………………………………………… İmza : ………….………………