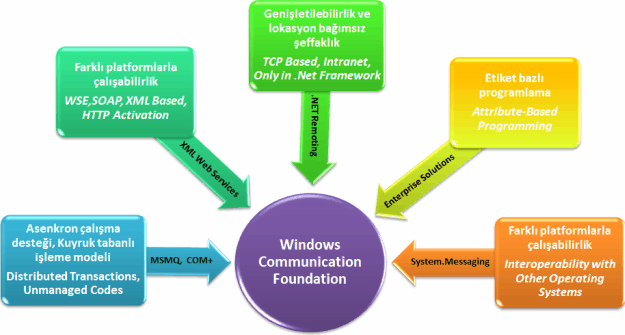


***WCF NEDİR?***

**Windows Communication Foundation WCF, servis-yönelimli mimariyi temel alarak dağıtık-uygulamalar geliştirmek için kullanılan, dağıtık mimari modelleri ve teknolojileri tek çatı altında birleştiren ve içerisinde bir çok hazır bileşen barındıran bir Framework**[**API**](http://en.wikipedia.org/wiki/Application_programming_interface)**‘sidir. (Bknz.**[**Service Oriented Architecture**](http://en.wikipedia.org/wiki/Service-oriented_architecture)**)**

**Ayrıca WCF mimarisi, .NET Remoting, XML Web Servisleri, Web Service Enhancements (WSE), Enterprise Services/COM+ ve Microsoft Message Queue(MSMQ) gibi bir çok dağıtık uygulama geliştirme modelinin tek bir programlama modeli içerisinde geliştiriciye sunmaktadır.**

****

WCF Temelleri

**.Net Framework 3.0 ile hayatımıza giren WCF, istemciler ve servisler arasındaki mesajları göndermek için oluşturulan sistemleri geliştirmek için kullanılan**[**API**](http://en.wikipedia.org/wiki/Application_programming_interface)**topluluğudur. Bu mesajlar, TCP, HTTP, MSMQ, Web Servisleri vb. gibi ortak taşıma protokolleri kullanarak Intranet ve Internet üzerinden gönderilebilir. Ayrıca mesajlar güvenli bir şekilde gönderilmekte ve dijital bütünlüğü için imzalanabilmektedir. WCF mesajlaşma işlemi için, istek/yanıt (request/response) şeklinde basit mesaj desenlerinin (message patterns) yanı sıra iki kanal üzerinden iletişim gibi daha karmaşık desenleri desteklemektedir.**

**Diğer önemli özelliği ise Web Service Enhancements (“WS-\*”) standartlarına uyan Web Servisleri ile farklı platformlarda (interoperability) çalışabilmesidir yani WCF bu standartları uygulayarak Microsoft olmayan diğer platformlarda oluşturulmuş istemciler yada servisler ile iletişim kurabilir.**

**Makaleye devam etmeden WCF içerisinde sıklıkla kullanılan genel kavram ve terimleri tanıyalım ;**

**Message:** **veri parçalarının birbirinden bağımsız hali olan paketlerdir, başlık (header) ve gövde (body) dahil olmak üzere bir kaç bölümden oluşmaktadır. Bir mesajın başlık dışında, tüm parçaları şifreli ve dijital imzalanmış olabilir.**

**Service: bir yazılım modülü (EXE veya DLL), bu modülün n tane**[**endpoint**](http://www.kazimcesur.com/wcf/#endpoint)**desteği vardır ve her endpoint n tane servis işlemi yapmamıza olanak sağlar.**

**Endpoint:** **istemciler (clients) ve host’lar arasındaki iletişimi şağlamak için kullanılan bir arayüzdür. Her endpoint’in kendi adresi vardır ve bu adresin benzersiz olması için servisin kendi ana adresine eklenmektedir. Bir WCF servisini aslında endpoint topluluğu olarak düşünebiliriz.**

**Binding:** [**endpoint**](http://www.kazimcesur.com/wcf/#endpoint)**‘lerin dış dünya ile nasıl haberleşileceğinin tanımlandığı bir takım özellikler bütünüdür. Yani [endpoint](http://www.kazimcesur.com/wcf/" \l "endpoint) ile ilitişimin (HTTP yada TCP) nasıl yapılacağının tanımlandığı noktadır. Bir binding ile**[**endpoint**](http://www.kazimcesur.com/wcf/#endpoint)**‘in kullanacağı security yada message pattern gibi detayları ayarlanabilmektedir.**

**System-provided bindings: Bazı senaryolar için optimize edilmiş [binding](http://www.kazimcesur.com/wcf/" \l "bindings) topluluğudur. Örneğin, *WSHttpBinding*Web Servislerinin farklı platformlarda (interoperability) çalışabilmesi için çeşitli WS-\* özelliklerini uygulamış hazır bir binding’tir.**

**Service Contract:** **contract servisin adını, namespace’ini ve diğer global niteliklerinin tanımlandığı bileşendir. Uygulamada bir contract, bir intarface oluşturularak bu interface’e *[ServiceContract]*etiketi uygulanılarak oluşturulmaktadır. Aşağıdaki**[**Örnek 1**](http://www.kazimcesur.com/wcf/#sample1)**de *IMyBasicService*interface’inde uygulanış şeklini gözlemleyebilirsiniz. Asıl servis kodu ise bu interface ile genişletilmiş olan sınıftır.**

**Operation Contract:** ***[ServiceContract]*** **etiketi uygulanılarak oluşturulmuş interface’in içerisinde bulunan operasyonların (metodların) servis tarafında kullanılabilmesi için uygulanan etikettir. Interface içerisindeki metodlara** ***[OperationContract]* etiketi uygulanarak kullanılır.**

**Data Contract:** **Servisi kullanan istemciler ile ortak çalışmayı sağlamak için servisin kullandığı veri nesneleri hakkında bilgi veren veriler (metadata) olarak tanımlanabilir. Bu veri türleri açıklamaları Data Contract olarak bilinir ve tipler herhangi bir mesaj (parametreleri veya dönüş türleri) içinde kullanılabilir.**

**Host:** **genelde servisin yaşam süresini kontrol etmek amacıyla kullanılan bileşendir. Host tipik olarak bir console uygulaması, bir Window Activation Service (WAS) yada Internet Information Service (IIS) üzerinde olabilir. Host eğer IIS üzerinde ise, bir virtual directory oluşturulur ve içerisine servis assembly’leri, konfigürasyon dosyası konulur. IIS üzerinde çalışan host’a istemciden bir [message](http://www.kazimcesur.com/wcf/" \l "message) gelmediği sürece servis çalışıyor durumda değildir. Eğer bir istemci yada istemciler çalıştığı session’ı sonlandırırsa IIS de uygulamayı kapatıp kullanmakta olduğu sistem kaynaklarını serbest bırakır. Aşağıda bu bileşen ile ilgili console uygulamasında host edilen Program sınıfını inceleyebilirsiniz.**

**Behaviour: servisin çalışma zamanındaki fonksiyonelliğini zenginleştirerek gerekli parametrik ayarların yapıldığı bir tipdir. Servis behaviour’ları ilgili servis sınıfına** ***[ServiceBehaviour]*** **etiketi uygulanarak kullanılmaktadır ve istenilen behaviour özelliklerinin ayarlanması ile servis çeşitli fonksiyonellikler sergileyebilmektedir.**

**Aslında behaviour’lar, bir servis yada [endpoint](http://www.kazimcesur.com/wcf/" \l "endpoint) için çalışma zamanındaki çeşitli beklentilerimizi kontrol etmek için kullanılmaktadır. Behaviour’lar kapsamına göre gruplandırılmıştır ; ortak olarak kullanılan behaviour’lar tüm**[**endpoint**](http://www.kazimcesur.com/wcf/#endpoint)**‘leri etkilemektedir, servis behaviour’ları yalnızca servisler ile ilgili özellikleri etkilemekte, [endpoint](http://www.kazimcesur.com/wcf/" \l "endpoint) behaviour’ları yalnızca ilgili [endpoint](http://www.kazimcesur.com/wcf/" \l "endpoint) özelliklerini etkilemektedir. Örneğin bir endpoint behaviour’ı ile güvenlik için kullanılan kimlik bilgilerine nereden ve nasıl erişilebileceği belirlenebilir.**

**Instancing:** **Her servisin çalışma zamanında kaç tane intance’ının çalışacağının belirlendiği bir model vardır. Bu modeller Single, PerCall, PerSession ve Shareable olmak üzere toplamda dört tanedir. Bu modellerden ilk iki tanesi .Net Remoting içerisindeki [Singleton ve Singel Call SAO](http://www.codeproject.com/Articles/29945/All-you-need-to-know-about-NET-Remoting" \t "blank) ile benzerlik göstermektedir.**

**Instance modelinin belirlenmesi işi aslında bir [behaviour](http://www.kazimcesur.com/wcf/" \l "behaviour) tanımlamaktır** ve ***[ServiceBehaviour]*etiketi ile tanımlanmaktadır. Örneğin** ; ***[ServiceBehavior(InstanceContextMode = InstanceContextMode.PerCall)]***

**Client Application:** **bir yada birden fazla [endpoint](http://www.kazimcesur.com/wcf/" \l "endpoint) ile**[**mesaj**](http://www.kazimcesur.com/wcf/#message)**alışverişinde bulunan program. Tipik olarak bir client application WCF Client nesneleri oluşturarak proxy üzerinden bu nesneler yardımıyla method çağırımları yapmaktadır.**

**Channel:** **client application ile bir servis arasındaki iletişim kanalıdır. Bir client TCP, HTTP gibi iletişim protokolleri ve bir adres kullanarak belirli bir [endpoint](http://www.kazimcesur.com/wcf/" \l "endpoint) için iletişim kanalı oluşturabilmektedir**.

**WCF Client Object:** **İstemci tarafı servis operasyonlarını enkapsule etmek için kullanılan yapılardır. Başka bir değişle WCF Client Object servis metodlarına erişim için kullanılan proxy’ler olarak tanımlayabiliriz.**

**WCF Client Object herhangi bir uygulama tarafından hatta bir servis olarak bile host edilebilmektedir.**

**Bu nedenle bir servis içerisinde diğer servislerin proxy’lerini içeren nesneler oluşturup kullanmak mümkündür.** (***service aggregation*** **– bu özellik SOA içerisinde önemli bir konseptir.) Bu proxy’ler de genellikle** ***svcutil.exe*** **komut satırı aracı kullanılarak oluşturulmaktadır.**

**Metadata:** ***svcutil.exe*** **komut satırı aracı kullanılarak oluşturulan veridir. Bu veri aşağıdaki bileşenleri içermektedir.**

* **Servisin**[**data contract**](http://www.kazimcesur.com/wcf/#datacontract)**‘larını tanımlayan XML şemayı**
* **Servis metodlarının tanımlamak için kullanılan WSDL’i**
* **Application configuration dosyalarını içermektedir.**

**Metadata exchange point:** **metadata bilgilerini yayınlamak için kullanılan bir [endpoint](http://www.kazimcesur.com/wcf/" \l "endpoint) adresidir.**

**Security:** **WCF içerisindeki güvenlik mesaj şifreleme, mesajların bütünlüğünü korumak için kimlik doğrulama (authentication) ve yetkilendirme (authorization) fonksiyonları içermektedir. Bu fonksiyonlar varolan HTTPS gibi güvenlik mekanizmaları ile sağlanmakta yada (“WS-\*”) standartlarına uyan çeşitli güvenlik tanımlarını uygulayabilmektedir.**

**Message pattern: istemci ve servis arasındaki mesajların ilişkisi ve yönünü belirler**.

**En temel message pattern’i one-way (diğer bir adıda one-shot)’dir. Bu pattern servise tek yönlü herhangi bir mesajı göndermek için kullanılır ve geriye bir cevap (response) beklenmez.**

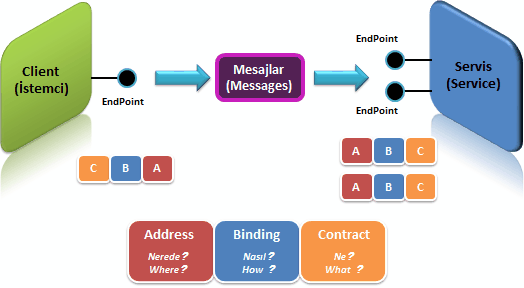
**En karmaşık pattern ise dual HTTP‘dir. Bu pattern ile istemci ile servis arasında iki tane HTTP kanalı oluşturulur. Bu şekilde istemci ve servisde kendi taraflarında çeşitli operasyonları çağırıp kullanabilmektedir.**

**Reliable messaging:** **Bir mesajın sadece bir kez alındığının ve hangi sırayla gönderildiğinin güvencesidir.**

**Sessions:** **bir session client ve servis arasında iletişim kurmak için kullanılmaktadır. Client ve servis arasındaki tüm mesajlar session kimlik bilgileri ile etiketlenmekte ve bu şekilde session tanımlanarak ilgili client mesajları ile istenen işlemler yapılmaktadır. Eğer bir session herhangi bir sebepten kesilirse aynı kimlik bilgileri ile tekrar başlatılabilir.**

WCF MİMARİSİ

WCF ABC’si



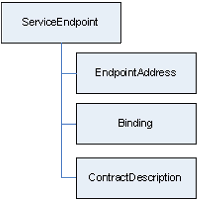
**Bir WCF servisinin aslında endpoint topluluğu olduğunu ve her endpoint’in dış dünya ile haberleşme noktası olduğunu, bir client uygulamasının bir yada birden fazla endpoint ile message alışverişinde bulunabildiğini hatırlayalım. Her endpoint ise Address, Binding, ve Contract bileşenlerine sahip olmalıdır.**

‘**ABC**‘ **aslında bir**[**endpoint**](http://www.kazimcesur.com/wcf/#endpoint)**‘in nasıl oluştuğunu anlamak için kullanılan bir betimdir ve tanımlayacak olursak:**

* + **A** **‘nın açılımı Address. Servisin nerede olduğunu belirtir**.
  + **B** **‘nin açılımı Binding. Servis ile iletişimde kullanılacak aktarım protokolü (Http, Tcp, NamedPipes vb.), kodlama (encoding text, binary vb.) ve güvenlik gereksinimlerini ( SSL, SOAP message security vb.)belirtir ve servise dış dünyanın hangi kurallar ile bağlanabileceğini ifade eder.**
  + **C** **‘nin açılımı Contract. Servis ile yapılabilecek işlemleri ifade eder**.

**Oluşturacağınız herhangi bir WCF servisi,  arkaplanda hangi yöntemi/teknolojiyi kullanırsa kullansın mutlaka yukarıda açıkladığınız bu üç temel bileşene sahip olmalıdır.**

**ServiceEndpoint** **sınıfı bir Endpoint’i temsil eder ve bir EndpointAddress, bir Binding ve ContractDescription bileşenlerini içerir.**

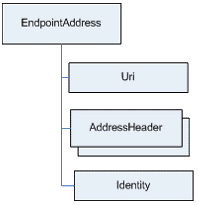


**Endpoint Address**

**Bir EndpointAddress temelde bir tanım** (**URI**) **bir kimlik bilgisi (identity), aşağıdaki şekilde gösterildiği gibi isteğe bağlı başlık topluluğudur ve servisin nerede bulunduğunu belirtir.**

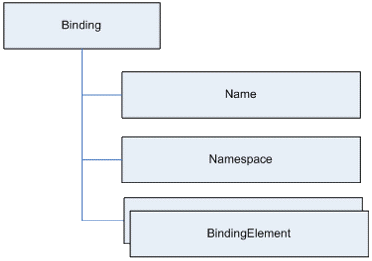
**Bir Endpoint güvenlik kimliği (identity) normalde tanım olmakla birlikte, gelişmiş senaryolarda bu bileşeni URI’den bağımsız olarak Identity adres özelliğini değiştirerek ayarlayabilmekteyiz.**

**AddressHeader’lar bir endpoint’e URI ötesinde ek adresleme bilgileri sağlamak için kullanılır. Örneğin, adres başlıkları aynı adres tanımını paylaşan çok sayıda endpoint arasında ayırım yapmak için faydalıdır.**



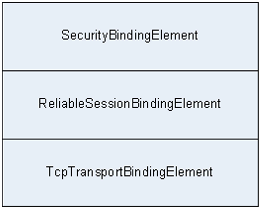
**Bindings**

**Binding bir endpoint ile nasıl etkileşime geçileceğinin belirlendiği bileşendir ve Name, Namespace ve BindingElement öğelerinden oluşmaktadır.**



**Aslında her binding element endpoint ile nasıl bir etkileşim ile iletişm kurulacağını belirtir. Binding elementleri inceleyecek olursak ;  
TcpTransportBindingElement :** **endpoint taşıma protokolü olarak TCP kullanılarak dış dünya ile haberleşileceğini belirtir.**  
**ReliableSessionBindingElement :** **endpoint’e message teslimi sağlamak için güvenilir bir mesajlaşma kullanacağını belirtir.**  
**SecurityBindingElement :** **endpoint’in SOAP message güvenlik tanımlarının kullanılacağını belirtir.**

**Her binding element’in genellikle endpoint ile iletişimin nasıl yapılacağını açıklayan özellikleri vardır. Örneğin ReliableSessionBindingElement’in Assurances özelliği sayesinde en az bir kez, en fazla bir kez, ya da tam olarak bir kez gerekli mesaj iletimine güvence verilir.**

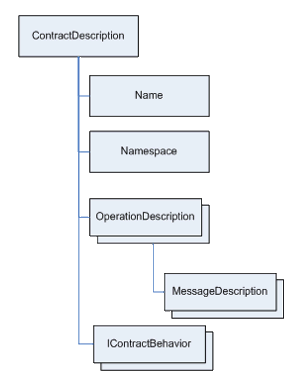


**Contracts**

**WCF Contract, endpoint’in dış dünya ile iletişiminde kullanacağı operasyonlar topluluğudur. Aslında buradaki her operasyon basit bir mesaj alışverişidir. (one-way yada request/response gibi)**

**WCF Contract’larını ve operasyonlarını tanımlamak için ContractDescription sınıfı kullanılmaktadır. ContractDescription içerisinde ise her Contract operasyonunun one-way yada request/reply şeklinde mi bir yaklaşım ile çalışacağının** **ayarlandığı** **OperationDescription** **bulunmaktadır. Her bir OperationDescription da** **MessageDescription** **topluluklarını kullanarak işlemler ile oluşturulan mesajları açıklar.**

**WCF programlama modeli içerisinde bir ContractDescription genellikle Contract’ı oluşturulan interface yada sınıf üzerinden oluşturulur. Oluşturulan tipe ServiceContract etiketi uygulanarak ve endpoint operasyonları içinde tip içerisindeki metodlara da OperationContract etiketi uygulanarak kullanılır**.



**Behaviors**

**Bir ServiceBehaviour, IServiceBehaviour interface’i ile genişletilmiş ve yalnızca servislere uygulanan olan bir türdür. Aslında behaviors, servis veya client’ın işlevselliğini genişletmeye yarayan türleridir. ServiceBehaviour’a benzer olarak IChannelBehaviour interface’i ile genişletilmiş ve yalnızca client channel’lara uygulanan Channel Behaviour türüde WCF içerisinde hazır olarak gelmektedir.**

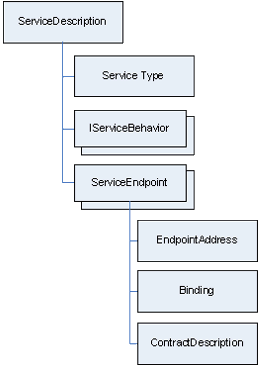
**Behaviour’lar bir channel oluşturma sürecine katılım sağlarlar ve kullanıcı tarafından belirlenen behaviour ayarlarına göre ilgili channel davranışını değiştirebilirsiniz.**

**Channels and Description**

**Service-side**

**ServcieDescription, bir WCF servisinin kullanıma açtığı endpoint’lerin, servis tarafından kullanılan behaviour’ların ve tiplerin (class vb.) yapısal olarak tanımıdır.**

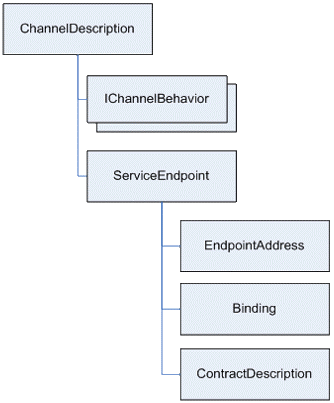
**Bir ServiceDescription sınıfı metadata oluşturmak ve channel ayarlarını yapmak için kullanılmaktadır :**



**Ayrıca ServiceDescription’ı ilgili WCF etiketlerini tiplerinize uygulayarak oluşturabilirsiniz.**

**Client-side**

**Service-side’a benzer olarak, bir ChannelDescription bir WCF istemcisinin iletişim kuracağı belirli bir endpoint’in yapısal olarak tanımıdır.**



**ServiceDescription’a farklı olarak, bir ChannelDescription istemci iletişim kanalının hangi endpoint ile iletişim kurulacağını temsil eden bir ServiceEndpoint içerir.**

**WCF Runtime**

**WCF Runtime aslında mesaj gönderme ve alma işlemlerinden sorumlu nesneler kümesidir. Bu mesajları biçimlendirme, güvenlik uygulama, çeşitli protokolleri kullanarak mesaj gönderme ve alma işlemlerini kapsamaktadır. Aşağıdaki başlıklarda WCF çalışma zamanının temel kavramları tanıyalım:**

**Message**

**Bir WCF Message bir istemci ile bir endpoint arasında veri alışverişi birimidir ve bir message aslında bir**[**SOAP message InfoSet**](http://www.w3.org/TR/soap12-part0/)**‘inin bir bellek içi gösterimidir. WCF message’ları XML formatına bağlı kalmamakla birlikte geliştiricinin belirleyeceği kodlama (encoding) biçimine göre text, binary, yada özel olarak serileştirilebilmektedir.**

**Channel**

**Channel’lar (Transport channels) ve (Protocol channels) olmak üzere iki ana kategoriye ayrılmaktadır. Transport channels TCP, UDP veya MSMQ aktarım protokollerini kullanarak stream gönderme ve alma işlemlerini gerçekleştirmektedir. Protocol channels ise SOAP spesifikasyonlarını uygulayarak message’ları işleme ve düzenleme işlemlerini gerçekleştirmektedir.**

**Endpoint Listener**

**Bir EndpointListener çalışma zamanında bir ServiceEndpoint eşdeğerdir.**

**ServiceEndpoint’in EndpointAddress, Binding ve Contract bileşenleri EndpointListener’ın listening address, message filtering ve dispatch, ve channel stack bileşenlerine karşılık gelmektedir.**

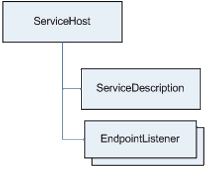
**EndpointListener içerisinde yer alan Channel (Stack) Yığını mesaj gönderme ve alma sorumludur.**

**ServiceHost and Channel Factory**

**Service Host**

**WCF Runtime genellikle** **ServiceHost.Open** **metodu çağırıldığında arka planda oluşturulur. ServiceHost, ServiceDescription nesnesinin oluşturulması ve bu nesne içerisindeki yer alan ServiceEndpoint koleksiyonunun config dosyasında yada kod içerisinde tanımlanan Endpoint’ler ile doldurma işlemlerini gerçekleştirir.**

**Sonrasında ServiceHost, ServiceDescription’ı Channel Stack oluşturmak için kullanır. Channel Stack ise EndpointListener nesnesi şeklinde olarak ServiceDescription içerisindeki her ServiceEndpoint’ten faydalanılarak oluşturulur :**



**Channel Factory**

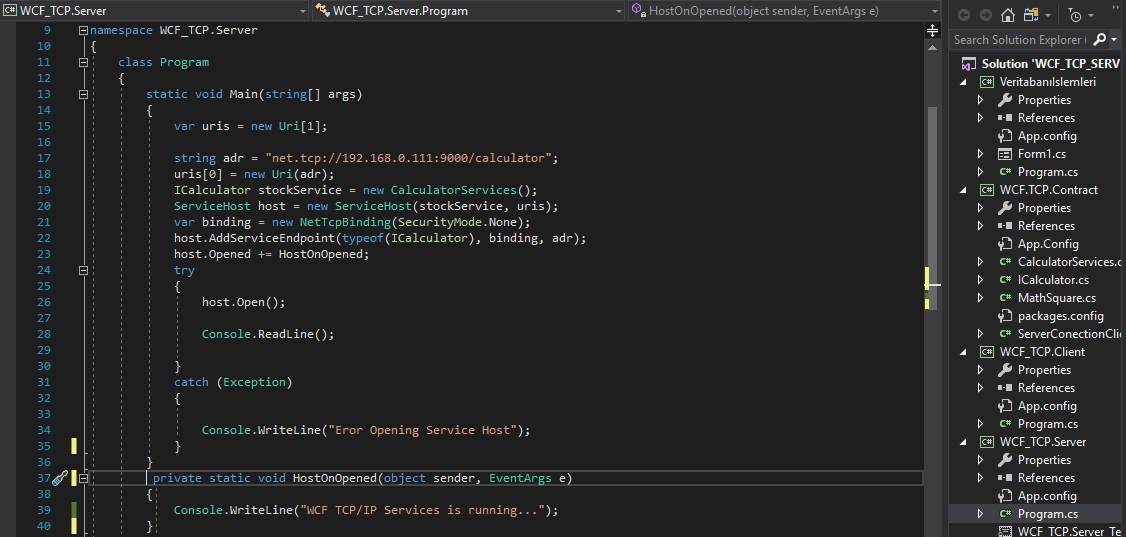
**Server-side’a benzer olarak, client runtime ChannelFactory tarafından oluşturulmaktadır. ChannelFactory ise client tarafında sunucu tarafındaki ServiceHost’un eşdeğeridir. ChannelFactory ise Contract tipi, Binding, ve EndpointAddress’ini baz alarak ChannelDescription’ı oluşturmaktan sorumludur.**

**ChannelFactory daha sonra ChannelDescription’ları client’s channel stack oluşturmak için kullanır.**

**Service runtime’dan farklı olarak client runtime EndpointListener içermez çünkü client’a devamlı gelen bir bağlantı yoktur, bunun yerine client Service’den devamlı istekte bulunur.**

**ÖRNEK -1**

**WCF TCP Server**

****

**1-)Uri sınıfı öncelik olarak belirli URL ile başlatılır.**

**2-)String tipinde adres belirtilir ve port yazılır.**

**3-)Belirtilen interface içerisinde servisin o interface içerişinde çalışması için yeni bir instance alınır.**

**4-)ServisHost sınıfından instance alınır ve uris değişkenin içine atılan adres ve o adres ile ilişkili çalışacağı sınıf(interface) ilişkilendirilir.**

**5-)Var tipinde binding değişkeni oluşturup bağlantı için yeni NetTcpBinding() instance üretilir ve isteğe göre Güvenlik Modu Seçilir.**

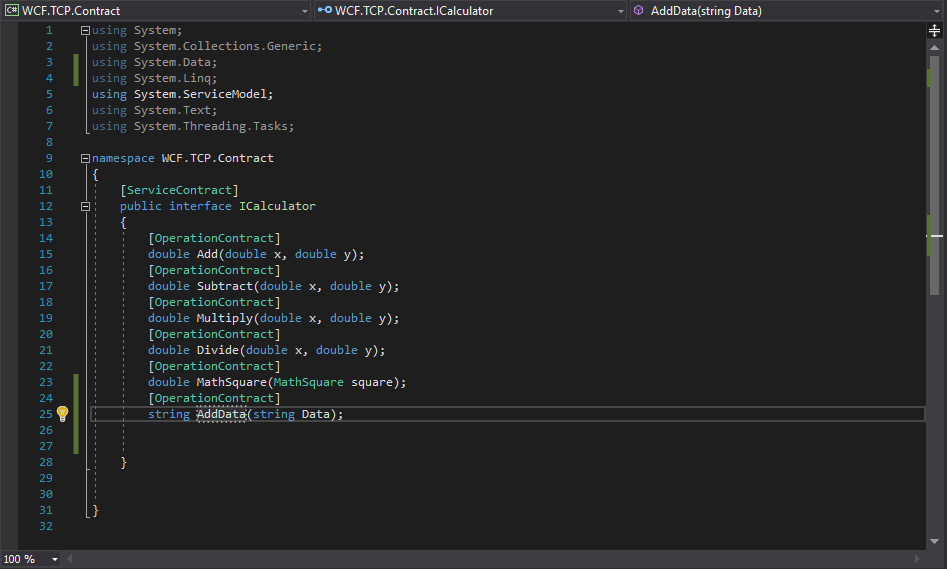
**6-)Servise Endpoint ekler ve bu endpoint servisi ICalculator inteface bağlantı tipi ve adresle ilişkilendirilir.**

**7-)Hostumuz çalıştığındaki yapacak işler için Console Kısmına Mesaj verdik.**

**8-)Hata durumunda try catch bloğuna alınır. Hata olduğunda catch kısma gelip Eror Opening**

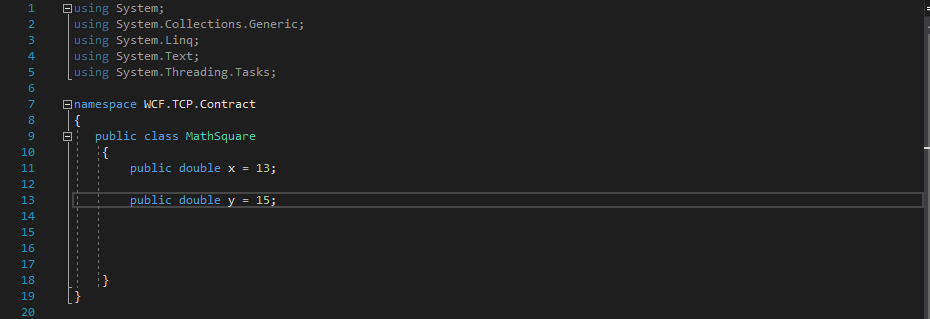
**Service Host hatası verecektir.**

**WCF.TCP.CONTRACT**

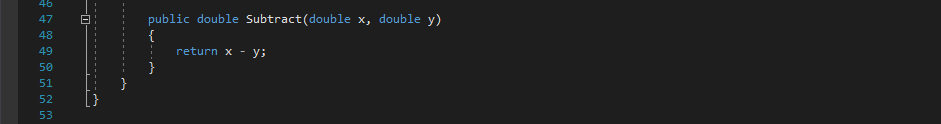
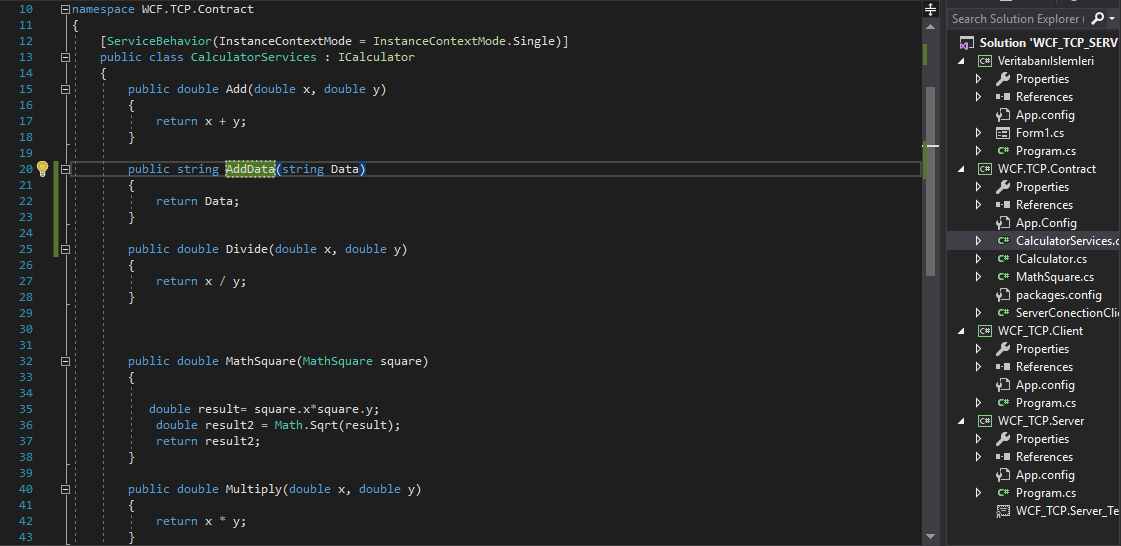
****

**Server ile ilişkilendirdiğimiz interface bu interfacedir. Bu interface ServiceContract olmak zorundadır.Server bu interfacenin ServiceContract olmasına bakacaktır.Aksi takdirde hata verecektir.**

**OperationContract kısmı ise yapılacak işleri belirtir Örnek olarak üst şemada görüldüğü gibi matematiksel işlemler yer almaktadır.**

**Not:Her bir methodun bir OperationContract olmak zorundadır.**

**MathSquare class’ı oluşturduk bunu ICalculator kısmında tanımladık çünkü Server,Servis contract olarak onu tanıyor ona göre işlem yapıyor**

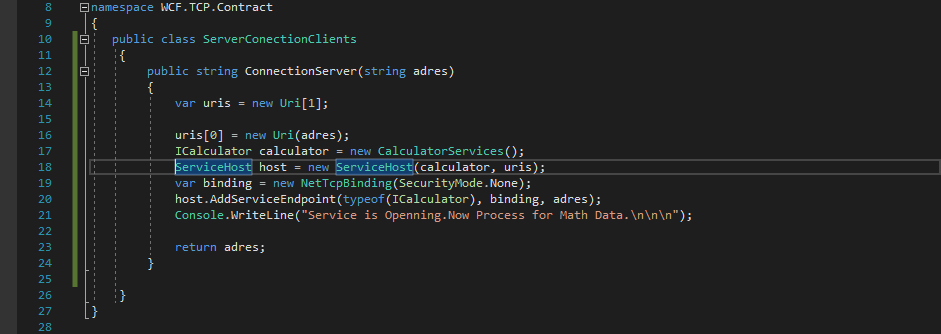
****

***ICalculator interface’sini implement ettiğimizde bu methodlar otomatik olarak gelecektir.***

***NOT:Üst kısımda [ServiceBehavior(InstanceContextMode=Contextmode.Single)] anlamı***

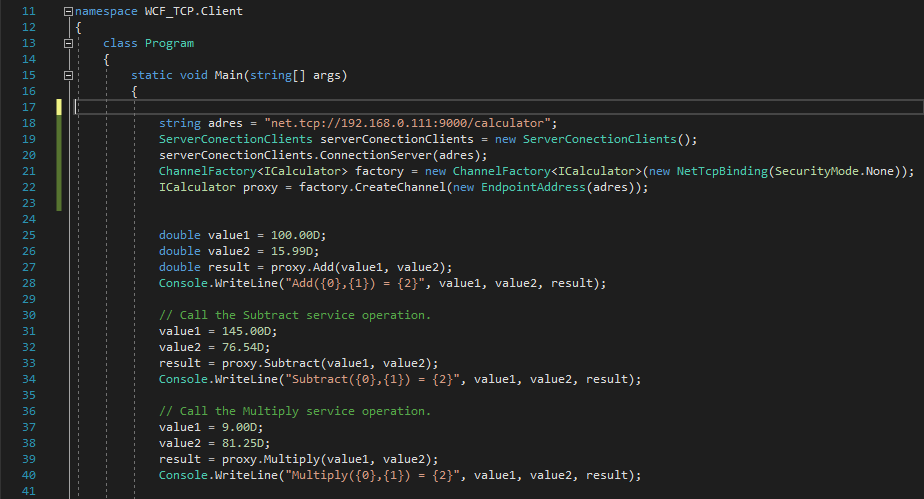
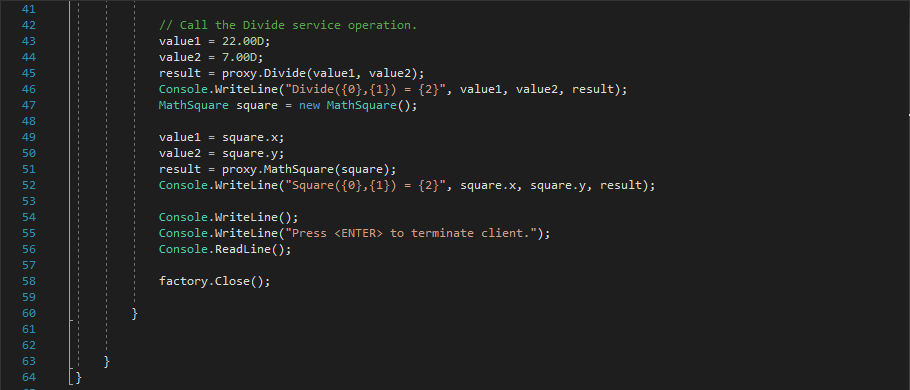
**Instancing:** **Her servisin çalışma zamanında kaç tane intance’ının çalışacağının belirlendiği bir model vardır. Bu modeller Single, PerCall, PerSession ve Shareable olmak üzere toplamda dört tanedir. Bu modellerden ilk iki tanesi .Net Remoting içerisindeki [Singleton ve Singel Call SAO](http://www.codeproject.com/Articles/29945/All-you-need-to-know-about-NET-Remoting" \t "blank) ile benzerlik göstermektedir.**

**Instance modelinin belirlenmesi işi aslında bir [behaviour](http://www.kazimcesur.com/wcf/" \l "behaviour) tanımlamaktır** ve ***[ServiceBehaviour]*etiketi ile tanımlanmaktadır. Örneğin** ; ***[ServiceBehavior(InstanceContextMode=Contextmode.Single)]***



**ServerConnectionClients sınıfını yazmamızın amacı her seferinde her clientin içerisinde bu kodları tekrar tekrar yazmamız gerekcek çünkü bu servis bağlantısına göre veriler işlenecek ve geri döndürülecek en basta Server kısmındaki kod bloğu ile ile benzer ama burada farklılıklar vardır.Her seferinde clientlere bu kodu yazacağımıza bir sınıf oluştururuz icersinde adres parametresi olan ve geriye dönüş tipi olan adresi gönderirirm o adres üzerinden veri alışverişi yapabiliriz.Amaç kod tekrarlılığını önlemektir.**

**Console Application Client kısmı**

******

**En üstte TCP/IP adres değişkeni oluşturduk ve adresimizi yazdık daha sonra ConnectionServerClient diye bir class oluşturmuştuk ondan bir instance oluşturduk ve oradaki ConnectionServer methodumuz vardı geriye adres dönderen.**

**Bizde adresimizi attık.**

**Channel Factory kısmı belirledik bizim kanalımız ICalculator olcak ve kanalımız NetTcpBinding olcak diyoruz. Security mode kısmı zaten none yapmıştık.**

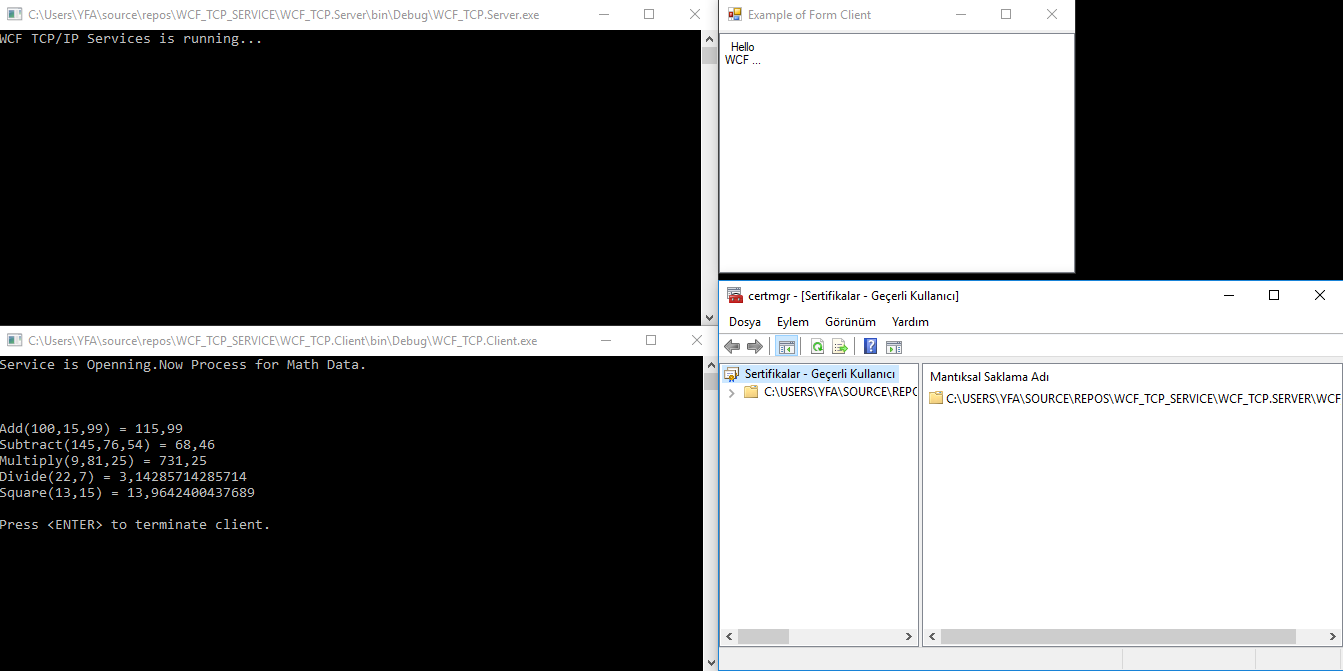
**Kanalın belirli özelliklerini tanımladıktan sonra ICalculator Proxy =factory.CreateChannel kısmı var burada bizim belirli özelliklere göre kanal oluşturup tüm işlemlerimizi artık bu Proxy kısmında gerçekleştirmemizi sağlıyor.**

**Not:EndpointAdress kısmı zaten bizim adres kısmımız oluyor**

**Örn:net.tcp://192.168.0.111./9000/calculator gibi.**

**Daha sonra bu Proxy bizim ICalculator sınıfımızda matematiksel işlemlerin methodlarımız var burada değişkenler oluşturup o interfacenin içerisindeki methodlara buradan parametreler gönderiyoruz interface ise implement edilen ICalculatorService sınıfında gerekli matematiksel hesaplamaları yapıp Client kısmında Console.WriteLine methodu ile çıktı sağlıyor.**

**Bir adet Form Applicationumuz var Console’da yapılan işlemlere benzer Form applicationdada yapabilirsiniz. Herhang bir Farklılık yoktur.**

****

**NOT:Server kısmının altında Sertfika bölümü var oradan güvenlik ayarları yapmayı unutmayınız.**

**WEB SERVİS WCF(Windows Communication Foundation) ARASINDAKİ FARKLAR**

**1-) Web servis ler tamami ile http protokolü üzerinden çalışırlar.**

**Wcf servisleri tcp,msmq,http gibi daha geniş protokol ağı kullanırlar. Bununla birlikte**

**endpoint yapıları ve kodlama standartlarındaki farkılılardan dolayı başlangışda daha fazla iş yapılıyormuş**

**izlemi versede web servislere göre daha esnek bir yapı sağlarlar.**

**Asmx(WebServiste Kullanılır.)**

**2-) Asmx SOAP ve Http kullanarak mesajlaşabilirken, WCF default olarak Http olmasıyla birlitke TCP/IP, MSMQ, NamedPipes etc. gibi yapıları da kullanabilmekte.**

* **Asmx servisler sadece IIS üzerinde Host edilebilirken, WCF IIS, WAS (Windows Process Activation Services), Console Application, Windows NT Services üzerinde host edilebilmektedir.**
* **Asmx’de güvenlik sınırlıdır. Kimlik doğrulama ve yetkilendirme işlemleri IIS,  Asp.Net ve TCP Protokolü kullanılarak yapılır ve kesin bir güvenlik tutarlılığı yoktur. WCF her bir protokol için ayrı bir güvenlik hizmeti sağlar ve bu yönden daha tutarlı bir güvenlik alt yapısına sahiptir.**
* **Asmx’de class’a [WebService] tag’i eklenmeliyken, WCF’de class’a [ServiceContract] tag’i eklenmelidir**

* **Asmx serilization işlemi için XmlSerializer kullanırken WCF XmlSerializer’den daha performanslı olan DataContractSerializer’ı kullanır.**