Webbtjänster med API er

Mål med lektionen!

- Veta kursmålen.
- Lite grunder om WCF

Vem är **jag**?

- Mitt namn är Björn Jönsson och jobbar på Tahoe Solutions, ni når mig via **mail**: bjorn.jonsson@tahoesolutions.se
- Jag är Certified ScrumMaster sedan 2008 och har programmerat i +10 år. Jobbat mycket med C# och .Net, SQL, JavaScript och SharePoint.

Vilka moment **omfattar** kursen?

- Hur man får datorer att kommunicera med varandra över internet med hjälp av http protokollet.
- Skapa och konsumera
  - REST services med hjälp av Microsoft ASP.NET Web API & ASP.NET MVC.
  - SOAP services med hjälp av WCF.
- Analysera och förstå de grundläggande säkerhetshoten för web-services enligt OWASP top 10 listan.
- Hur man autentiserar simot REST services med hjälp av Oauth2/OpenID-connect.
- Arbeta med JSON med hjälp av Newtonsoft Json.Net biblioteket.
- Anropa REST services med hjälp av HttpClient.
- Felsökning av REST tjänster med hjälp av Fiddler.
- Förstå innebörden av HyperMedia för REST APIer

Målet med kursen

Mål som den studerande ska ha uppnått efter avslutad kurs

Den studerande ska ha kunskaper i/om:

- Kunna skillnaden på webbtjänster skapade med REST och SOAP
- Varför API'er är viktiga för distribuerade applikationer.
- Skillnaden mellan WCF och WEB.API
- Identitetshantering med hjälp av Oath2/OpenID-connect.

Målet med kursen (forts)

Mål som den studerande ska ha uppnått efter avslutad kurs

Den studerande ska ha färdigheter i att:

- Skapa säkra REST API web-services med hjälp av ASP.NET WEB.API.
- Skapa enklare SOAP web-services med hjälp av WCF.
- Praktiskt kunna kommunicera med en extern webbtjänst.
- Kunna praktiskt skapa och modifiera data lagrad i XML och JSON.
- Kunna motivera och välja när man ska använda WCF respektive SOAP.
- Använda protokollen Oath2/OpenID-connect.

Målet med kursen(forts)

Mål som den studerande ska ha uppnått efter avslutad kurs

Den studerande ska ha kompetens för att:

- Designa och utveckla SOAP och REST web-service lösningar.
- Praktiskt utveckla webbtjänster med både WCF och WEB.API
- Välja teknologi och plattform för att skapa API-lösningar.
- Praktiskt kunna konsumera REST och SOAP baserade tjänster.
- Förstå protokollen Oath2/OpenID-connect.

Vad krävs för **godkänt**?

- Praktiskt kunna konsumera befintliga REST och SOAP baserade tjänster.
- Skriftligt kunna förklara de grundläggande skillnaderna mellan SOAP och REST.
- Skriftligt kunna förklara de grundläggande skillnaderna mellan WCF och WEB.API.
- Praktiskt kunna konsumera information lagrad i XML och JSON format.
- Skriftligt kunna redogöra för de vanligaste säkerhetshoten enligt OWASP top 10.
- Skriftligt kunna förklara skillnaderna mellan Oath2/OpenID-connect.

Vad krävs för **väl godkänt**?

- Självständigt kunna skapa en REST baserad webbtjänst med WEB.API.
- Självständigt kunna skapa en SOAP baserad webbtjänst med WCF.
- Praktiskt kunna skapa och bearbeta information lagrad i XML och JSON format.
- Praktiskt kunna demonstrera de vanligaste säkerhetshoten enligt OWASP top 10.
- Fördjupad kunskap i hur Oath2/OpenID-connect fungerar.

Vad skall vi **göra** på denna kursen?

- Anteckna gärna och framför allt fråga om det behövs!
- Följ gärna med i Slidarna på datorn

Inlämnings uppgifter

- Vi kommer att ha 1 inlämningsuppgift:
  - Detta kommer att vara en ”rapport” (redogörelse) som skall lämnas in och den är obligatorisk, dvs den **skall** lämnas in och måste vara godkänd för att man skall kunna få betyg.

En **liten** innehållsförteckning

- Lektion 1 (intro)
- Lektion 2 (services)
- Lektion 3 (clients)
- Lektion 4 (REST-services)
- Lektion 5 ()
- Lektion 6 ()
- Lektion 7 ()
- Repetition 1 ()
- Repetition 2 ()

- Tenta (Fredagen 19:e Maj dvs v20 Kl 09:00-12:00)

Vad lektionen omfattar

- Webbtjänster?
- Vad är WCF?
- Ta en titt på grundläggande termer och begrepp.

Webbtjänster?

En webbtjänst är en tjänst som erbjuds av en elektronisk anordning till en annan elektronisk anordning, som kommunicerar med varandra via World Wide Web.

I en webbtjänst, Web-teknik såsom HTTP, som ursprungligen var avsedd för människa-till-maskin kommunikation, används för maskin-till-maskin kommunikation, närmare bestämt för att överföra maskinläsbara filformat som XML och JSON.

WCF

Windows Communication Foundation (WCF)

Är ett ramverk för att bygga serviceorienterade applikationer.

Med hjälp av WCF, kan du skicka data som asynkrona meddelanden från en service slutpunkt (endpoint) till en annan. En service slutpunkt kan vara en del av en ständigt tillgänglig tjänst hostad av IIS, eller det kan vara en tjänst i en applikation.

En slutpunkt kan vara en klient hos en service som begär data från en service slutpunkt.

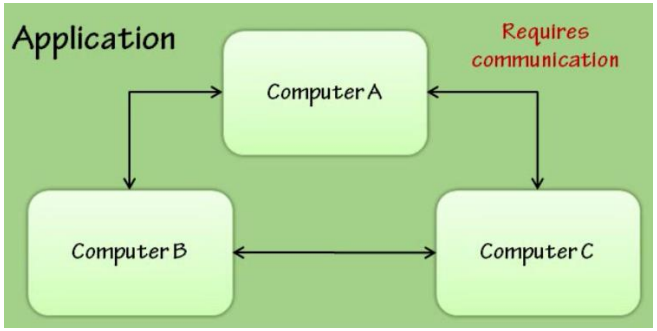
Meddelandena kan vara så enkelt som ett enda tecken eller ord skickas som XML, eller så komplicerat som en ström av binära data

WCF

När man pratar om WCF och vad WCF är så har Microsoft ändrat om benämningen till ”connected system” (tidigare var det distributed system).

Men vad är connected system:

> *Ett sk ”connected system” är ett system som är distribuerat över flera dator noder som har olika delar av ett visst program, men jobbar tillsammans och kommunicerar emellan varandra genom att skicka meddelanden.*



Historiskt

- DCOM/COM+/ES
- .NET Remoting
- MSMQ

Microsoft Web Services

Microsoft har haft ett par försök på web services genom åren.

- ASP.NET Web services (asmx)

- Web Service Enhancements (WSE)

Service Orientation

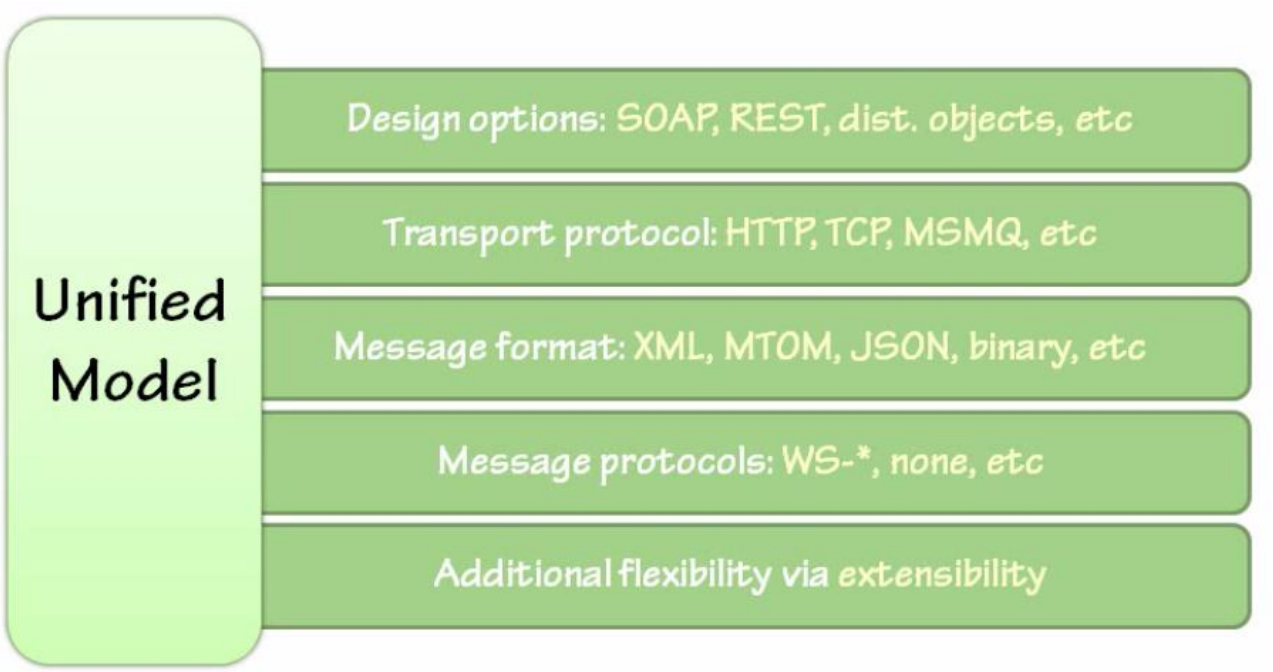
Microsoft har använt sig av både SOAP baserade och REST baserade tjänster tidigare även om de är fundamentalt olika ifrån varandra så anses de vara ”service oriented” i grund och botten.

Men vad är **Service Oriented**?

Det hela är en slags generell design paradigm (design mönster) som fokuserar på **SoC**.

Kommunikations ramverk

Med de tidigare erfarenheterna i minnet så insåg Microsoft snabbt att en av deras viktigaste design mål skulle vara att bygga en enhetlig programmeringsmodell för all kommunikations logik.



WCF

WCF vill vara default valet för att koppla samman applikationer idag.

Du använder WCF för att skriva kommunikations logiken, och sedan bestämmer du vilken av alla dess funktioner som du vill använda.

Den mesta av WCF's funktionalitet kan du hitta i en endaste assembly som heter:

System.ServiceModel.dll

Om man skall uttrycka det enkelt så är det "Ett sätt att skriva koden, men många sätt att koppla ihop punkterna".

Vilket är värdet med WCF

Services and endpoints

Med WCF så skriver vi tjänster som exponerar olika endpoints mot värden.

Vår service implementation definierar den faktiska business logiken (det är så vi skriver koden) och våra endpoints definierar de olika kommunikations möjligheterna som vi vill låta servicen supporta/stödja (det är så vi kopplar ihop punkterna).

Tjänster kan exponera multipla endpoints, dvs vi kan ha hur många vi vill. Men vi måste ha åtminstone en (för har vi inte minst en så är den inte särskilt nyttig eftersom ingen kan kommunicera med den.

Endpoints?

En endpoint är en bit information som talar om för WCF hur man (den) skall bygga de underliggande kommunikations kanalerna som kommer att användas vid körning av applikationen, både för att ta emot och skicka meddelanden.

En endpoint består av tre saker:

- **Address** – vart man skall skicka meddelanden.
- **Binding** – hur man skall skicka meddelanden.
- **Contract** – vad meddelanden måste innehålla.

endpoints

Din implementering av en webbtjänst definierar själva affärslogiken för det är så vi skriver koden, och våra endpoints definierar de olika kommunikations sätten som våran tjänst stödjer, det är så vi kopplar ihop allt.

Tjänster kan exponera multipla endpoints för konsumenter, vi kan ha hur många vi vill men minst en.

Bindings?

En bindning är egentligen ett ritning för hur WCF kommer att bygga den underliggande kommunikations kanalen.

WCF kommer med en hel hög med inbyggda bindningar för några av de vanligaste kommunikations scenariona.

Binding Name	Communication Scenario
WebHttpBinding	Interoperable RESTful communication via HTTP
BasicHttpBinding	Interoperable SOAP communication via HTTP, offering only the "basic" protocols conforming to WS-I Basic Profile
WSHttpBinding	Interoperable SOAP communication via HTTP, offering the full range of SOAP + WS-* protocols
NetTcpBinding	Cross-machine WCF communication via TCP
NetPeerTcpBinding	Cross-machine WCF communication via P2P
NetNamedPipesBinding	Same-machine WCF communication via IPC
NetMsmqBinding	Disconnected/asynchronous WCF communication via MSMQ

Konsumera tjänster med WCF

Fram till nu så har vi nästan bara pratat om att använda WCF ifrån perspektivet av att implementera tjänster, men vi kan också använda WCF för att konsumera tjänster över internet eller nätverk.

Men för att kunna konsumera en tjänst så behöver vi veta lite saker.

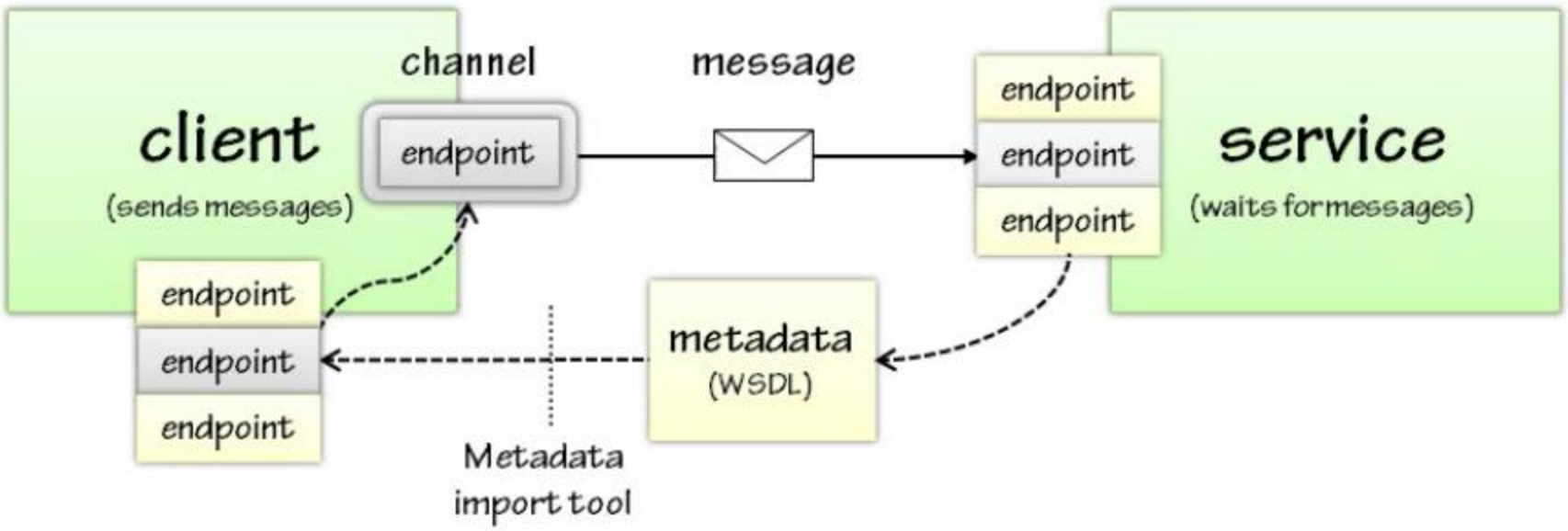
- En klient behöver veta vart den skall skicka ett meddelande.
- Den behöver veta hur den skall skicka meddelandet (protokoll?).
- Vad meddelandet skall innehålla (kontraktet)

Den kommer egentligen att behöva veta alla detaljer som är tillgängliga via en endpoint.

WCF klient

Med WCF så konsumerar man tjänster via kanaler baserade på endpoints (man bygger kanaler baserade på endpoints). Först och främst så måste klienten få tillgång till (hämta) endpointen från tjänstens metadata.

WCF klient (mer)



Övning

Börja lite enkelt med att skapa upp en konsoll applikation.

- Släng konsoll projektet som ligger där.
- Därefter lägg till ett nytt projekt(WCF Service Library)
  - Släng de två .cs filerna som följde med.
- Lägg till ett nytt item (klass fil)

Övning forts.

- I klassfilen skapa en publik klass med publika för och efternamn.
- Skapa ett publikt interface direkt under klassen som vi skapat nyss.
- Interfacet skall anropa en metod som tar ett Name objekt.
- Klassen vi fick när vi skapade från service projektet skall göras publik och skall ärva interfacet.

Övning forts.

VS 2013 och VS2015 har en inbyggd WCF testklient som vi skall använda nu.

- Öppna app.config, skriv in namnet på er service för taggen ”service” under attributet name.
- Ändra bas adressen till: ”http:/localhost:8080/helloworld”.
- Ta bort det gulmarkerade och gör taggen självstängande.

```
<endpoint address="" binding="basicHttpBinding" contract="HelloSecondWorldService.IService1">
  <!--
      Upon deployment, the following identity element should be removed or replaced to reflect the
      identity under which the deployed service runs.  If removed, WCF will infer an appropriate identity
      automatically.
  -->
  <identity>
    <dns value="localhost"/>
  </identity>
</endpoint>
```

- Kopiera sedan endpointen som har httpbinding 2 ggr. (ni skall ha tre totalt).

Övning forts.

För att vi skall kunna få lite mer smidiga adresser på våra endpointar i nästa slide så måste vi ändra på våran <baseAddress>.

I <add baseAddress=”…” ändra den till ”8080/helloworld”

```
<services>
  <service name="HelloWorldService.HelloWorldService">
    <host>
      <baseAddresses>
        <add baseAddress = "http://localhost:8080/helloworld" />
      </baseAddresses>
    </host>
```

Övning forts.

Våra endpointar skall se ut enligt följande:

```
<endpoint address="ws"
          binding="wsHttpBinding"
          contract="HelloSecondWorldService.IHelloWorld"/>
<endpoint address="basic"
          binding="basicHttpBinding"
          contract="HelloSecondWorldService.IHelloWorld"/>
<endpoint address="net.tcp://localhost:8081/helloworld"
          binding="netTcpBinding"
          contract="HelloSecondWorldService.IHelloWorld"/>
```

kompilera allt nu genom att trycka F5

Övning forts.

Testklienten ser ut enligt följande: