WCF & ASP.NET Web API – An Architect's Primer

Presented at Southern California .NET Architecture User Group; Feb 21st 2013.

Adnan Masood MS (CS). MCSD. MCPD.

Software Architect & Doctoral Candidate <u>blog.AdnanMasood.com</u> adnan.masood@owasp.org

About Me

Adnan Masood works as a system architect / technical lead for Green dot Corporation where he develops SOA based middle-tier architectures, distributed systems, and web-applications using Microsoft technologies. He is a Microsoft Certified Trainer holding several technical certifications, including MCPD (Enterprise Developer), MCSD .NET, and SCJP-II. Adnan is attributed and published in print media and on the Web; he also teaches Windows Communication Foundation (WCF) courses at the University of California at San Diego and regularly presents at local code camps and user groups. He is actively involved in the .NET community as cofounder and president of the of San Gabriel Valley .NET Developers group.

Adnan holds a Master's degree in Computer Science; he is currently a doctoral student working towards PhD in Machine Learning; specifically discovering interestingness measures in outliers using Bayesian Belief Networks. He also holds systems architecture certification from MIT and SOA Smarts certification from Carnegie Melon University.

Abstract

WCF vs. ASP.NET Web API – An Architect's Primer

ASP.NET Web API is a framework that makes it easy to build HTTP services that reach a broad range of clients, including browsers and mobile devices. The new ASP.NET Web API is a continuation of the previous WCF Web API projection. WCF was originally created to enable SOAP-based services and other related bindings. However, for simpler RESTful or RPCish services (think clients like jQuery) ASP.NET Web API is a good choice.

In this meeting we discuss what do you need to understand as an architect to implement your service oriented architecture using WCF or ASP.NET web API. With code samples, we will elaborate on WCF Web API's transition to ASP.NET Web API and respective constructs such as Service vs. Web API controller, Operation vs. Action, URI templates vs ASP.NET Routing, Message handlers, Formatters and Operation handlers vs Filters, model binders. WebApi offers support for modern HTTP programming model with full support for ASP.NET Routing, content negotiation and custom formatters, model binding and validation, filters, query composition, is easy to unit test and offers improved Inversion of Control (IoC) via DependencyResolver.

Agenda

- Difference between Web API and WCF REST Services
- How to Migrate from WCF Web API to ASP.NET Web API
- Model for RESTFul Maturity
- WCF or Web API confusing?
- WCF evolution and strengths
- What ASP.NET Web API brings to the table?
- Architectures and comparing non functional requirements

Architectural Questions

- What is the purpose of the WebAPIs?
- Why do we need REST HTTP services? What's wrong with SOAP-over-HTTP?
- Why did the WebAPIs move from WCF to ASP.NET MVC?
- Is there still a use for WCF? When should I choose Web APIs over WCF?

History of Web Services

1989 - Tim Berners-Lee invents HTTP/HTML
1998 - XML 1.0, SOAP begins ratification
2001 - SOAP standard
2000 - Fielding dissertation on REST

History of SOAP

Before SOAP we did this...

- HTTP GET/POST with Plain Old XML (POX)
- Out-of-band exchange of DTD or schema
- SOAP evolved to provide us
 - Specifications
 - Tooling
 - Metadata
 - Productivity
 - Iocation transparency

Tools Evolution

- ASP.NET Web Services (ASMX)
- Web Services Enhancements (WSE)
- •.NET 3.0 => WCF = SOAP+WS*
- •.NET 3.5 => WCF = SOAP+WS*/ HTTP
- WebHttpBinding, contract attributes, JSON
- •REST Starter Kit (Codeplex)
- •WCF 4 => Features from starter kit
- •WCF Web API => ASP.NET Web API



<u>Scenario</u>	WCF 4.5	ASP.NET Web
		<u>API</u>
Need to support specific scenarios like Message queues,	<u>lt's me!</u>	
duplex communication, end to end message security,		
distributed transactions, one way messaging etc		
Already you have existing working WCF services and	<u>lt's me!</u>	
would like to add HTTP support additionally.		
One code base to support both SOAP and RESTful	<u>lt's me!</u>	
endpoints		
Need to create a resource-oriented services over HTTP		<u>lt's me!</u>
Your project is a MVC application and want to expose		<u>lt's me!</u>
some functionality over HTTP		
Want to build only a HTTP / RESTful services		<u>lt's me!</u>
Duplex communication over HTTP	Sigr	nalR
SQL backend and need to expose OData endpoints	WCF Data	a Services

SignalR

What is ASP.NET SignalR

- ASP.NET SignalR is a new library for ASP.NET developers that makes it incredibly simple to add real-time web functionality to your applications. What is "real-time web" functionality? It's the ability to have your serverside code push content to the connected clients as it happens, in realtime.
- You may have heard of WebSockets, a new HTML5 API that enables bidirectional communication between the browser and server. SignalR will use WebSockets under the covers when it's available, and gracefully fallback to other techniques and technologies when it isn't, while your application code stays the same.
- SignalR also provides a very simple, high-level API for doing server to client RPC (call JavaScript functions in your clients' browsers from server-side .NET code) in your ASP.NET application, as well as adding useful hooks for connection management, e.g. connect/disconnect events, grouping connections, authorization.

The WCF Web API => ASP.NET Web API

- WCF Web API -> ASP.NET Web API
- Service -> Web API controller
- Operation -> Action
- Service contract -> Not applicable
- Endpoint -> Not applicable
- URI templates -> ASP.NET Routing
- Message handlers -> Same
- Formatters -> Same
- Operation handlers -> Filters, model binders

Integrated stack

- Modern HTTP programming model
- Full support for ASP.NET Routing
- Content negotiation and custom formatters
- Model binding and validation
- Filters
- Query composition
- Easy to unit test
- Improved Inversion of Control (IoC) via DependencyResolver
- Code-based configuration
- Self-host

	REST	SOAP
PROTOCOL	HTTP	HTTP
REQUEST	GEThttp://xxx/user/detail/id/1001	<pre>POST http://xxx/user <s:e><s:h></s:h><s:b> <detail id="1001"></detail> </s:b></s:e></pre>
RESPONSE	<pre>RETURN CODE: 200 GET RESPONSE: <user> <name f="sumeet" l="rohatgi"></name> <active_since t="2008"></active_since> </user></pre>	<pre>RETURN CODE: 200 POST RESPONSE: <s:e><s:h></s:h><s:b> <user> <name f="sumeet" l="rohatgi"></name> <active_since t="2008"></active_since> </user> </s:b></s:e></pre>



Simple Object Access Protocol
Uses a standard XML Schema over HTTP
Extremely cross platform compatible
Extremely Slow



Representable State Transfer
Uses standard HTTP
Can use any text format including XML

XML vs JSON

XML tag based document formatting
Javascript Notation by Douglas Crockford
JSON less verbose than XML, more lightweight
Mobile devices have limited bandwidth

Public APIs

Twitter
Facebook
Flickr
Amazon
Tunes



Available now as Nuget Package
Built-in as part of MVC 4
Take advantage of HTTP features directly

Default route will use http method for action
Controller/action/id
API/Controller/id GET/POST/PUT/DELETE

HTTP methods as Actions

HTTP Method meanings

Get - Return an existing document
Post - Create a new document
Put - Update a document
Delete - Self explanatory

Configure Transport

 Set Xml or JSON based on Content-Type or Accept header
 Accept: application/xml
 Can also use Odata

Return Codes

Now have the ability to specify return codes beside 200
 HttpResponseMessage<YourEntity>
 HttpStatusCode.Created 201
 response.Headers.Location = new Uri()

Http Status codes

201 Created
200 Success/204 Success but No Content
403 Not authorized
404 Does not exist
500 Server Error
301 Uri Moved



[Authorize()]
https over port 443
Security Tokens
OAuth

Testing WebAPI

Download Fiddler2
Firebug (Firefox)
Chrome
On Mac use CocoaRestClient

Consuming WebAPI

 Web Apps (ASP.NET, MVC, PHP, Java, ColdFusion, Ruby(Rails), Python, Perl(if you are masochistic))
 JavaScript/JQuery
 Mobile (iOS, Android, WP7, Blackberry OS)

Develop WebAPI and iOS on Same Computer

Parallels or VMWare
Set Network Adapter to Bridged
Run Visual Studio as Administrator
Host on IIS (do not use IIS Express or Casini)

Consuming WebAPI in iOS

Use NSURLConnection delegate or GCD
 Show progress while waiting on response
 Use JSON over XML
 NSJSONSerialization class (new in iOS 5)

XML Parsing in iOS

NSXMLParser (Slowest)
libxml2 (C api)
TBXML (DOM, Fastest, no write or xpath)
TouchXML (DOM supports xpath, no write)
KissXML (based on Touch, can write)
GDataXML (DOM, from Google)
RaptureXML (DOM, supports xpath)

Create DefaultHttpClient();
Create request with HttpGet(Url);
Create response handler BasicResponseHandler();
httpClient.execute(request, handler);

Call WebAPI from Android

JSON in Android

use the JSONObject to parse
JSONObject jo = new JSONObject(jString);
jo.getJSONObject("car");
jo.getJSONArray("cars");

XML Parsing in Android

DOM, SAX and Pull
W3C Dom parser
Standard Java Sax Parser
SJXP (Pull parser)

WebAPI as persistence

Don't use WebAPI as default persistence on Mobile
Both Android and iOS have device persistence
local storage, CoreData and SQLite
iCloud to sync between iOS devices



Comparison

Transport Coupling

- HTTP is an application protocol, not just a transport protocol
- TCP, named pipes, MSMQ, UDP are transport only
- WCF is decoupled, message can traverse any

Performance

Sometimes a faster protocol/serialization mechanism is needed

Security

Web API

- HTTP Services
- HTTPS / SSL
- Authorization header or custom headers
- OAuth 2.0
- WCF Services
 - HTTPS / SSL
 - SOAP Message Security
 - WS-Trust
 - OAuth 2.0

Error Handling

- Mostly automatic
- Helpful to control how things are returned to Ajax clients
- Setting status code and message



IIS or Self hosting

Slides courtesy Michelle L. Bustamante

Feature Comparison

- Productivity
 - Design effort
 - Complexity
 - Client code and proxy generation
 - Communication stack
- State
 - Both should be stateless
 - Caching
 - Built in to HTTP, but beware

The WS* Overload

Security WS-Security WS-SecureConversation WS-Trust OASIS Web Service	rity Policy WS-Federation SAML	Reliable Messaging WS-RM Policy WS-ReliableMessaging	WS-Policy WS-PolicyAttachment
Transactions WS-CAF WS-Coordination	and t WS* o	then overload	WS-Discovery WS-MetadataExchange WSDL
WS-BusinessActivity WS-AtomicTransaction	WS-Transfer WS-Enumeration WS-ResourceTransfer WSRF	Messaging WS-Eventing WSN WS-Addressing SOAP	MTOM sWa DIME

SOAP vs REST – Focus on the necessities



Web API Selling Points

- If we need a Web Service and don't need SOAP, then ASP.Net Web API is very useful.
- Web API Used to build simple, non-SOAP-based HTTP Services on top of existing WCF message pipeline.
- Web API No need for configurable like WCF REST services
- Web API No need for Data contracts
- Web API Could create fully blown REST Services
- Simple service creation with Web API. With WCF REST Services, service creation is difficult.
- WCF is any wire protocol. Web API is focused at one thing, being easy to define and expose and consume, while also facilitating doing things in a RESTful way.
- Web API is light weight architecture.

Comparison

FEATURE	WCF	WEB API
Transport	HTTP/S, TCP, UDP, MSMQ, named pipes, custom	HTTP/S
Protocols	WS*	HTTP
Content Format	SOAP+XML	Any media type, format
Types	Data contracts (opt in)	CLR Types (opt out)
Service Interface	Service contracts	URL patterns, HTTP methods
State Management	Stateless with Per Call	Stateless
Caching	Handled by application	Built-in to HTTP
		Prefer application control
Hosting	IIS or self-host	IIS or self-host
Error Handling	Faults. behaviors	Exceptions, HTTP status codes filters
Security	Windows, Basic, Certificate WS*, Authorization header	Windows, Basic, Certificate Authorization header
Client	Proxy generation	IApiExplorer discovery

The hypermedia Venn Diagram



The unified model? Kinda, sorta Obsolete



Revision - WCF to ASP.NET Web API

- ApiController (!=Controller, no common BaseClass)
- ASP.NET Routing (MapHttpRoute)
- Convention over Configuration
- Web API to go / NuGet Packages
- Web API hosted in ASP.NET: AspNetWebApi
- Self-hosted Web API: AspNetWebApi.Selfhost
- HttpClient including XML and JSON formatters:
- System.Net.Http.Formatting
 - JsonValue for navigating and manipulating JSON:
- System.Json
- Go Live License

How to Migrate from WCF Web API to ASP.NET Web API

 http://wcf.codeplex.com/wikipage?title=How%20to%20Migrate %20from%20WCF%20Web%20API%20to%20ASP.NET%20 Web%20API

Competing with Node.JS? Node-style Web API?

LiteWebServer server = new LiteWebServer("http://localhost");

```
server.Get("/Hello", (r) => new HttpResponseMessage() {
    Content = new StringContent("Hello World!")
});
```

```
server.Post("/Echo", (r) => new HttpResponseMessage() {
    Content = new StringContent(
        r.Content.ReadAsStringAsync().Result
        )
});
server.Open();
```

Resources & Credits

- http://www.asp.net/web-api
- http://blogs.msdn.com/b/henrikn/
- http://weblogs.asp.net/scottgu/archive/tags/Web+API/defa
- ult.aspx
- http://stackoverflow.com/questions/tagged/asp.net-web-api
- https://github.com/ChristianWeyer/Thinktecture.Web.Http
- http://blog.alexonasp.net
- <u>http://aspnet.uservoice.com/forums/147201-webapi/suggestions/2618312-allow-for-non-asynchronouscalls-to-the-web-api</u>
- Thanks to Michele Leroux Bustamante's slide-deck from Windows Azure connections, March 26-29, 2012 Las Vegas, NV which I thoroughly enjoyed.
- Ida Flatow's article on web API <u>http://www.codeproject.com/Articles/341414/WCF-or-ASP-NET-Web-APIs-My-two-cents-on-the-subjec</u>
- Alexander Zeitler's Web API Round up
- David Fekke Web API www.fekke.com/Media/Default/powerpoint/webapi.ppt

Summary

- Choose wisely and quantitatively; avoiding the shiny object syndrome.
- Enterprise WCF implementations will continue to be important...
- The trend to HTTP services is here to stay, embrace it
- If you are starting from scratch for a mobile / web heavy service, look at HTTP services first

Thank You!

Adnan Masood

adnan.masood@owasp.org

@adnanmasood

- Blog: www.AdnanMasood.com
- Pasadena .NET User Group: <u>www.sgvdotnet.org</u>