Aspecting EF and WCF

Scott Reed
Brain Hz Consulting, Inc.
scott@brainhzconsulting.com

Aspect Oriented Programming

- **Increase modularity**
- **Separate cross-cutting concerns**
  - Exceptions, Security, Logging, Caching, Threading
- **Frameworks like PostSharp and Spring.NET**
  - Allow applying these techniques across code base
- **But we aren't going to talk about those today…**

Services

- **One of the most important places to apply aspects is at a service or communication boundary**
- **When one program is calling another**
- **Great examples:**
  - WCF client calling a WCF service
  - Using Entity Framework to call a database

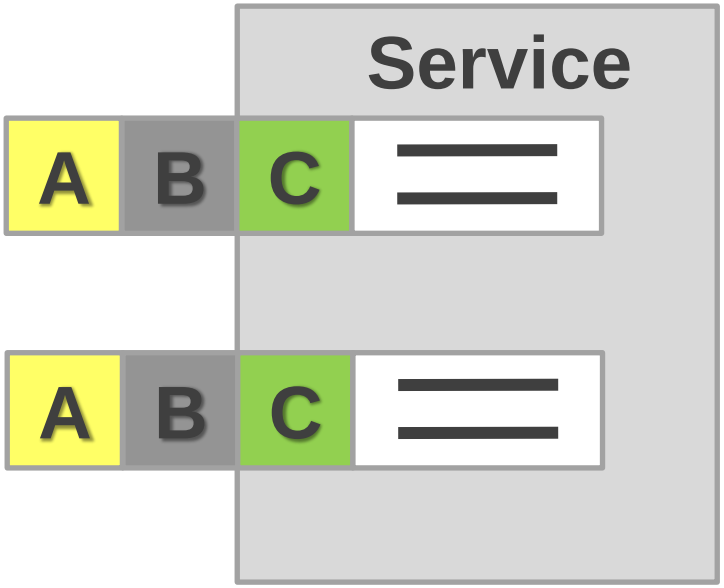
WCF Extensibility

- **In WCF there are two ways to extend**
- **The channel layer (the hard way)**
  - Only time you need to use the channel layer is when you want a single message to result in several outgoing messages
- **The service model layer (the easy way)**
  - Uses behaviors to plug in bits of code on one side or the other
  - Behaviors are like extending your cars engine

Steps to extend

- **Implement the functionality**
- **Implement a behavior**
  - to plug the functionality into WCF
- **Apply the behavior**

WCF Behaviors: Where to apply



WCF Behaviors: How to apply

- **Attributes**
  - Design time (Operation or Service)
- **Config**
  - Deploy time (Endpoint or Service)
- **Programmatically**
  - Run time (Operation, Endpoint, or Service)

Aspecting WCF Services

- From the service side you might want to:

- Log the calls (and their results)
- Record/Transform exceptions
- Time the calls
- Add performance counters
- Add custom authorization logic

- Only one way in WCF to do *all* of these…

IOperationInvoker

- **Functionality to invoke an operation**
  - At the operation level
  - Applied via an operation behavior

Demo

- Building an Operation Invoker to put whatever logic we want around calling a service method

- And now over to the client side…

Aspecting Entity Framework

- **When performing a database call you might want to:**

- **Log the call (with SQL)**
- **Figure out what to optimize**
  - Keep track of total number of each type of call
  - Time the call
- **Insert TOP (so you aren't getting back 6000 records)**
- **Return from the cache instead of actually calling**
  - (and if not in cache, cache the results after calling)

Entity Framework

- **Client side calls can be difficult for frameworks**
- **EF Not really designed for this sort of thing**
  - Even in 4.0
- **There is a workaround, but it is really complicated**

- **http://blogs.msdn.com/b/adonet/archive/2010/09/13/ef-caching-with-jarek-kowalski-s-provider.aspx**

Demo

- Hooking a new EF provider into the existing context

Aspecting WCF clients

- When making a call to a service you might want to:

- Make sure a channel has been opened
- Access the channel in a thread safe way
- Time the call (add performance counters)
- Catch exceptions and log them
- If the exception was just a communication glitch, try again
- If it is a GUI program, display an error
- Caching results

WCF client calls

- Are really just method calls
- How do you aspect arbitrary methods from an interface?

Demo

- Creating a ServiceCaller (really a method caller)

- Allowing nested ServiceCallers

Thank You

Scott Reed

Brain Hz Consulting, Inc.

scott@brainhzconsulting.com

www.brainhzconsulting.com