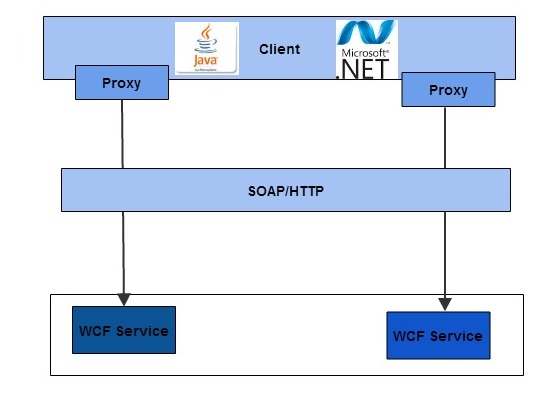
**WCF’s Run Time Architecture and SOA Implementation**

****

**Rameshkartik.RS**

**Table of Contents**

[Introduction 2](#_Toc392828000)

[WCF runtime Architecture 3](#_Toc392828001)

[Why SOA patterns are implemented in WCF? 6](#_Toc392828002)

[How SOA Patterns are implemented in WCF? 7](#_Toc392828003)

[Summary 10](#_Toc392828004)

# Introduction

In this article you will come to how the messages travel from the client application

the service application, what are all the various steps in it? In which way the developer

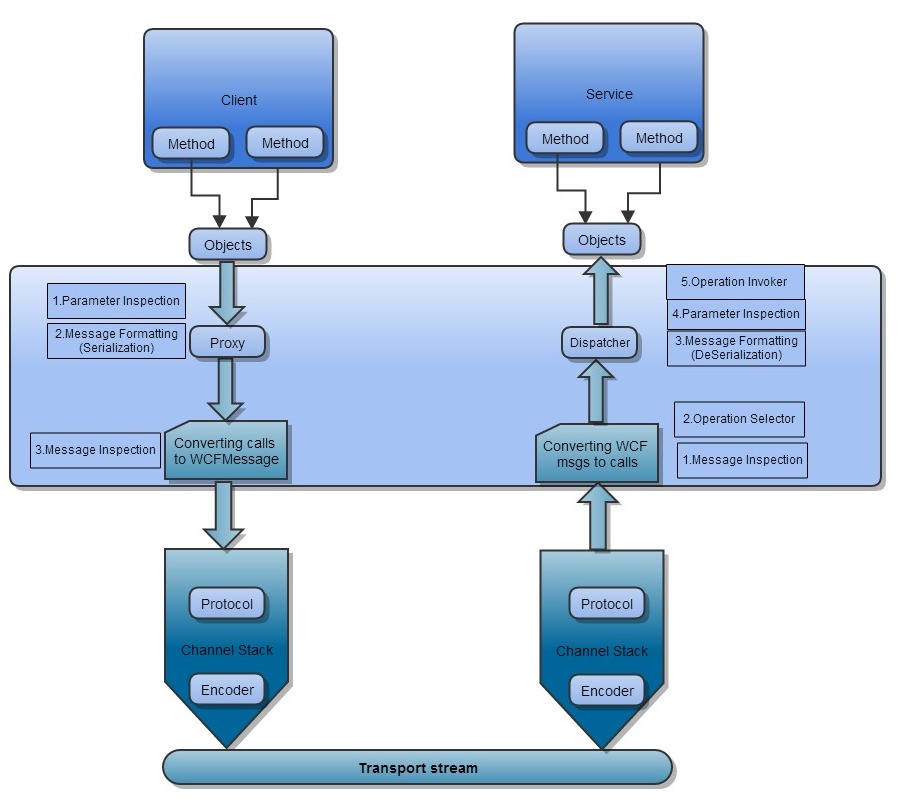
can extend the steps during the message travel.

Also, I would like to tell you something about the SOA patterns,Why the SOA Patterns are implemented in WCF? and How the SOA patterns are implemented in WCF? Ok, Lets talk about it

# WCF runtime Architecture

Am very much interested to discuss/write something about the WCF run time architecture, It is actually an important and must known concept for everyone who are trying to learn about the messaging patterns. It’s a multilayered architecture allowing different types of messaging patterns to be implemented. The data in the multilayered architecture travels from the client application to a stream transporting the message.

It travels up to the implementation of the service.



In the above figure, you can find the calls passing the proxy layer which actually converts the calls into the messages and sends them through the channel,and the dispatcher will gets back the calls by converting from the messages.

Proxy layer will keep pushing the messages on the channel, it is passing the protocol and the encoder layers before it is reaching the wires.On the services side, it passes through the encoder and protocol layers and it is received by the dispatcher. Finally, it’s the dispatcher’s responsibility to examine the messages and decide it which method to call. In the client side there are four extensible points before the data travels to the transport stream, The first one is the parameter inspection where you can hook to perform custom validation, value modification, or special filtering. Next extensible point the proxy leverages is message formatting(serializing) where you convert the calls into the WCF objects. You can also customize the serialization process by using a custom formatter object. And the final extension point is the message inspection and extension.

In the service side there are five extensible points, here the steps are completely reversed against the client side steps.

The first extensible point here is the message inspector where it can be used to dispatch runtime programmatically. Then the next step is operation selector where the dispatcher will select the operation. Once the target has been identified ,the dispatcher will deserializing the WCF messages into the objects that can be supplied as parameters when invoking the target method, At this point dispatcher provides the extension points for the message deserialization and the parameter inspection. The final step for the dispatcher is to invoke the target method, supplying the prepared parameters.

# 

# Why SOA patterns are implemented in WCF?

Actually the WCF team at Microsoft was trying to deliver by considering the following main points

* Interoperability across platforms
* Unification of existing technologies
* Service oriented programming

Interoperability across platforms – Microsoft wants the application running on windows should be communicated with legacy applications, MAC OS X Machines, Linux machines, windows clients, solaris machines and anyone else who abides by the WS-I (Web services interoperability Organization Specification).

Unification of existing technologies - WCF takes all the capabilities of the

Distributed technologies and overlay a simplified API called System. Service Model,

In which you can find all things of ASMX,WSE,System.Messaging,. Net Remoting

In one roof of WCF.

Service Oriented Programming – To increase the flexibility in the programming

Ease the business orientation of modern software projects, WCF takes us

Service oriented programming from the object oriented programming

# How SOA Patterns are implemented in WCF?

**SOA Design Patterns**

Establishing a robust service oriented architecture requires the business to take

In the account of Design Patterns, Security and API management. SOA design

Patterns allow organizations to solve their problem by using the proven solutions.

SOA patterns have the solutions for the commonly recurring problems within

the enterprise. The implementation environment for SOA design patterns must

allow loose coupling and reuse of integration solutions. We will see in WCF how

these patterns are used and why?

**Decoupled Contract**

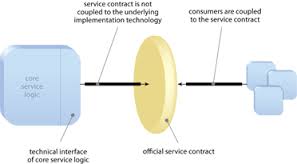
The reason why this ‘Decoupled Contract’ has been invented is “How can a service,

express its capabilities independently of its technical implementation?”In other words,

To provide an effective enterprise service, the technical contract should be completely

Independent from its logical implementation which is yet to align with the other

services. So only the interface details will be visible to the client not its implementation.



Diagram

By Decoupling the service contract ,the service implementation can be evolved without directly impacting service consumers, This can increase the amount of refactoring opportunities and reusability

**Proxy Pattern**

Proxy pattern is used to provide a surrogate object, which references to the other

object. Also called, it is a proxy class, it represents the functionality of another class.

public interface iAction

{

void PerformAction();

}

public class ServiceClass : iAction

{

public void PerformAction()

{

Console.WriteLine("ServiceClass action performed.");

}

}

public class Proxy : iAction

{

private ServiceClass \_servClass;

public void PerformAction()

{

if (\_servClass == null)

\_servClass = new ServiceClass();

\_servClass.PerformAction();

}

}

Let's discuss how the proxy pattern is applied in the WCF. In WCF as soon we create a

service reference for a wcf service, a proxy object will be created.

WCF provides the ClientBase<T> for the proxy classes to inherit, The base class has the

capability to set up the communication and other operation calls. The consumers send a

call against the proxy for which the SOAP messages will be send them to the service

using defined binding. AS I mentioned in the above source code example, the proxy

object provides a surrogate object servClass which represents the functionality of

another class called ServiceClass. The same concept being used in the WCF Service and

the proxy.

**Operation Context Pattern**

**It decouples the functional input parameters** from the technical information the method needs to execute. WCF provides a method called System.ServiceModel.OperationContext , this class will take over the responsibility of providing information about the call,session id,incoming message header in the SOAP

envelope,and information about the identity of the caller. The context contains lots of information about the non functional and therefore it should not be the part of the DataContract. When working with the duplex mode, operation context provides the channel to be used to call back the client during the method execution. Also the operation context provides an additional context called WebOperationcontext that provides the method a more information about the request in terms of HTTP properties.

# Summary

The main aim of WCF is to create interoperability across platforms,

unification among the legacy systems. Also the WCF supports the SOA patterns

Decoupled, Façade, proxy, operational context patterns to define the system

as SOA oriented.