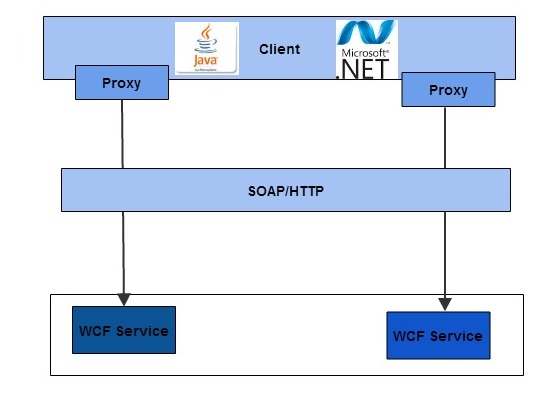
**WCF’s SOA Implementation**

****

**Rameshkartik.RS**

**Table of Contents**

[Introduction 2](#_Toc386275661)

[Why WCF has SOA implementation? 3](#_Toc386275662)

[How WCF implements SOA Patterns? 4](#_Toc386275663)

[Summary 8](#_Toc386275664)

**WHAT’s IN THIS ARTicle?**

* Why WCF has SOA’s Implementation?
* How WCF implements SOA’s patterns?
* Summary

# 

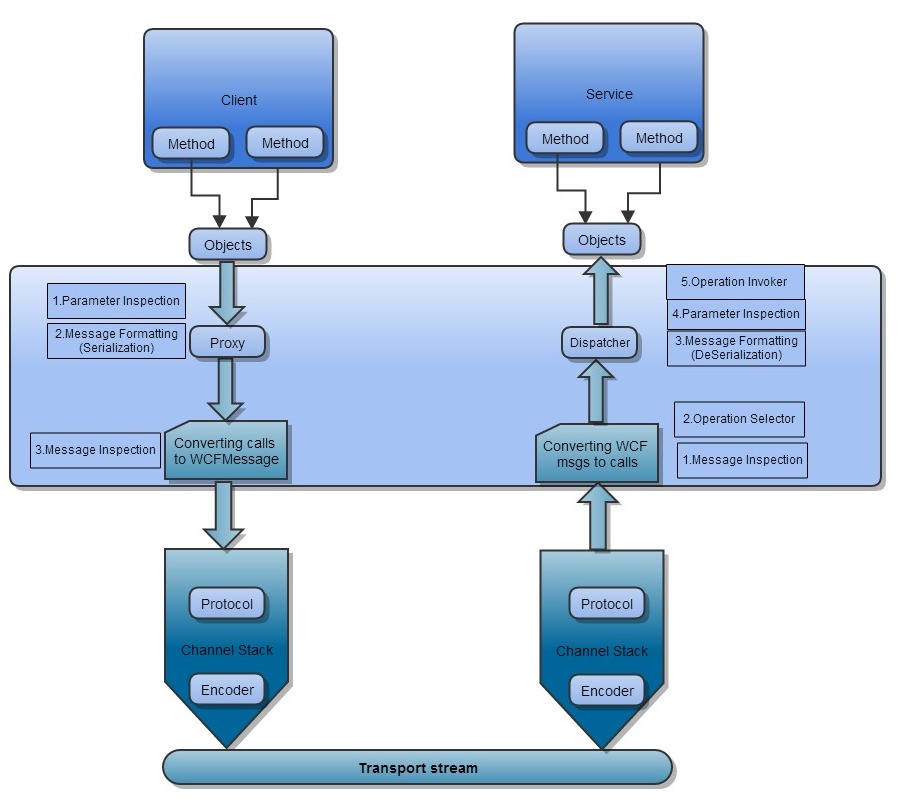
# Introduction

In this article you will come to how the messages travel from the client application to the service application, what are all the various steps in it? In which ways the developer can extend the steps during the message travel. So Happy Journey ☺

# WCF run time Architecture

Am very much interested to discuss/write something about the WCF run time architecture, It is actually a 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.