**WCF Service Failover**

**Table of Contents**

[Introduction 2](#_Toc464652623)

[Failover 2](#_Toc464652624)

[Need for Back up 3](#_Toc464652625)

[Routing Configuration 4](#_Toc464652626)

[Summary 7](#_Toc464652627)

# Introduction

Failover – Stand by server takes over when the main server fails.

In this article, I will explain the failover functionality to be achieved using WCF concepts. The fair idea behind going into failover is 24 X 7 availability of the Service to the critical client applications. The Client application should always in a position to consume the Backup Service though Main Service is not available due to some reasons. Let us discuss this in detail.

# 

# Failover

Failover is a method of protecting the applications/communication from failure, in which backup service

Automatically takes over when the main server fails. In the real world scenario, you can see that ticket

Booking websites take more time to book the tickets (to get the server response) during the peak

hours. The Server takes more time due to server’s unavailability or a number of requests in queue are

high or any other reason. Whatever the reason is, the client application should not get an impact,

another server should take it up when the main server is not available.

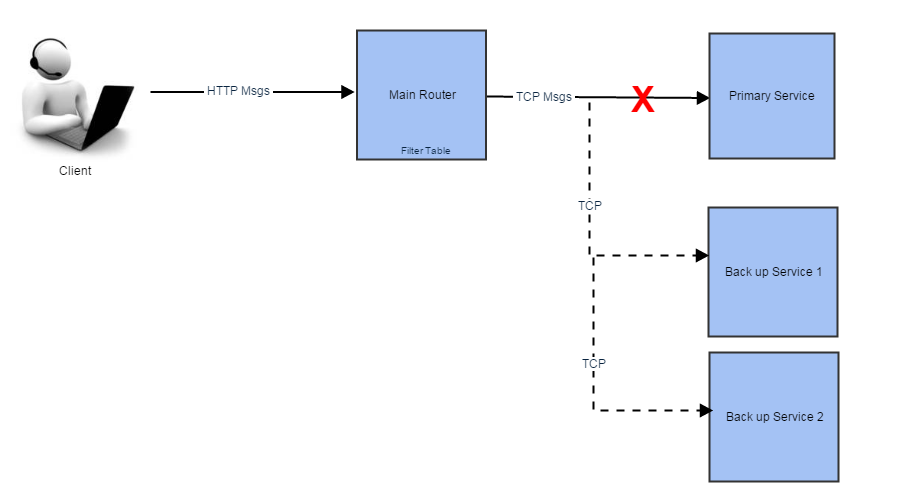


Figure1.0 – Work flow Diagram

In the snapshot, given above, you can find that the Client communicates with the main router. It does

not aware of the primary Service, backup Service. When the router receives the request from the

client, it usually tries to process it with the primary server. If the primary server is not reachable for any

reasons, the main router then automatically tries to connect with the backup Service 1. If the backup

Service 1 is available the request would be processed and send the response to the client. If the backup

Service 1 is also not available then it would send the request to backup Service 2 to get it processed.

The Client does not need to worry about the availability of servers. The main router will re-route the

request to the back-up Services and get it processed if the primary servers are not available.

# Need for Back up

Let us discuss in detail about the need of a back-up Services.

**Critical Applications Availability**

Critical applications are the one where large number of users always wants to perform many

Transactions at a time. For Ex: Bank Sectors, large number of peoples need to do fund transfer and other

Transactions as part of their day-to-day activities, if this kind of application servers gets affected

, the impact will be huge. To overcome these issues, backup server is always recommended to take care

of Client’s transactions if the primary server is not available

**Load Balancing**

There are situations where the Service gets more requests/transactions only during the peak time on

any single day. Let us take the ticket booking websites, a large number of users will try to book the

tickets once the tatkal period gets started. If the server would not capable enough to process all the

requests ,then the clients will not be able to book the tatkal tickets. To overcome this, the main server

should load balance the requests to the backup Services accordingly. Say if the Primary server is already

processing 1000 messages ,the main server would reroute the other 1000 messages to the backup

Services. This would increase the processing time and the client’s requests will not affect.

# Routing Configuration

Let us take the Figure 1.0 as a reference-configuring primary and backup Services

accordingly. As a first step, create the primary Service library and the host application. As usual, in the

Primary Service library you can define the contracts and Service implementation. After that create the

host application and refer the primary Service library, then define the Service, endpoint and host it. In

Our example I have done it in self-hosting. If you need assistance in configuring and hosting the

Services, Please look out my previous articles Configuring WCF Routing, Hosting in WCF.

Primary Service



Figure 1.2 Primary Service

From the Figure 1.2, you can find that primary Service hosted on net.tcp://localhost:8523/PrimaryServiceLib/Service. After that, you need to repeat the same configuration for the Backup Services 1 and 2. In our case, the address of Backup Service 1 and 2 are “net.tcp://localhost:8524/BackupService1/Service/”, net.tcp://localhost:8525/BackupService2/Service/. Now you have successfully hosted the Services Primary, Backup Service 1 and Backup Service 2. Now it is time to define the Routing Service, which will pass the messages to the appropriate Service, based on availability. As I have already mentioned in my previous articles, we shouldn’t forget this simple sentence “Address the filter and filter end” . It is just a sentence I have framed myself to remember the RoutingServer configuration steps, which are Address, Filter table, Filters and endpoint☺, As I shown in the Figure 1.3 snapshot configure the Routing Service accordingly

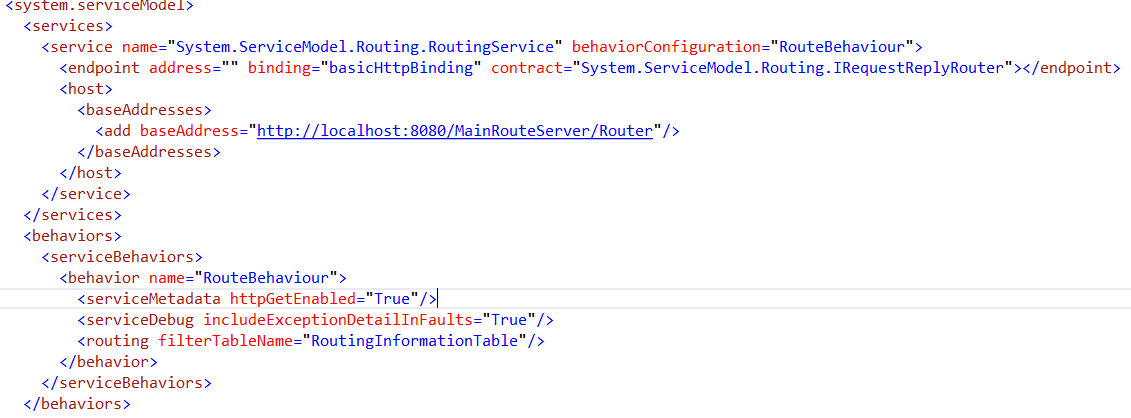




Figure 1.3 Routing Service

In the filter table named “RoutingInformationTable” you can find that MatchAllFilter have been applied and route to the end point named “PrimaryService\_TCPEndPoint”. MatchAllFilter is used to route all the messages from the client to the PrimaryService\_TCPEndpoint. You can find other attribute named “backuplists” in the filter element. If the RoutingService find the PrimaryService\_TCPEndpoint is not available for some reasons, it can route the messages to the endpoints listed in the BackupServices. If the backup Service 1 is also not available, then it re-routes to back up Service 2 endpoint. So the client will not aware of the Service from which Service his request served.

# Summary

Failover functionality is an important feature in the client-server communication.

It is quite natural that Services may stop/not reachable for any reasons. Routing Service will take

the client to the different Service (backup) if any of the primary Services is not available. Backup Service

will process the request and send it back to the client.