# **Webservice Lifecycle**

WCF中有三种实例模式：PerSession,PerCall,Single

PerCall模式:

　　客户端创建代理对象(Proxy).

　　客户端调用代理对象的一个Contract,代理对象将其传递给服务宿主.

　　宿主应用程序创建一新的ServiceContract，并且执行请求操作.

　　在执行完请求操作后，如果要求有应答，那么ServiceContract会给代理对象一个应答，然后销毁自己（如果实现了IDisposable，则调用Dispose()）

PerSession模式:

客户端创建代理对象(Proxy).

　 客户端第一次调用代理对象的一个Contract操作，代理对象将其调用请求传递给服务宿主.

　 宿主程序创建新的服务对象，并执行请求操作，如果有必要，返回客户端应答.

　 客户端再次发出调用操作的请求，宿主会先判断是否已有建立好的会话，如果存在，则不需要再创建新的服务对象，直接使用老对象即可。

　 在时间达到指定要求或者因一些特殊原因，会话会过期，此时服务对象销毁。

Single模式:

服务端启动，同时创建服务对象.

　客户端通过代理调用契约操作.

　第一步中创建的服务对象接受请求，并执行操作，进行必要的应答.

　第一步创建的服务对象将一直保留.

　服务关闭，第一步创建的对象销毁.

在msdn找到一篇对比ASP.NET Web Services or .NET Remoting的文章，

其中提到了状态管理这一节，可惜的是对于最后一句话，也是没有提到具体的解决方案。。

ASP.NET Web Services or .NET Remoting: How to Choose

http://msdn.microsoft.com/en-us/library/ms978420.aspx

The ASP.NET Web Services model assumes stateless service architecture by default; it does not inherently correlate multiple calls from the same user. In addition, each time a client invokes an ASP.NET Web service, a new object is created to service the request. The object is destroyed after the method call completes. To maintain state between requests, you can either use the same techniques used by ASP.NET pages, i.e., the Session and Application property bags, or you can implement your own custom solution.

