**粘包与分包问题概述：**

**指发送方发送的若干包数据到接收方接收时粘成一包，从接收缓冲区看，后一包数据的头紧接着前一包数据的尾。出现粘包现象的原因是多方面的，它既可能由发送方造成，也可能由接收方造成。发送方引起的粘包是由TCP协议本身造成的，TCP为提高传输效率，发送方往往要收集到足够多的数据后才发送一包数据。若连续几次发送的数据都很少，通常TCP会根据优化算法把这些数据合成一包后一次发送出去，这样接收方就收到了粘包数据。接收方引起的粘包是由于接收方用户进程不及时接收数据，从而导致粘包现象。这是因为接收方先把收到的数据放在系统接收缓冲区，用户进程从该缓冲区取数据，若下一包数据到达时前一包数据尚未被用户进程取走，则下一包数据放到系统接收缓冲区时就接到前一包数据之后，而用户进程根据预先设定的缓冲区大小从系统接收缓冲区取数据，这样就一次取到了多包数据。分包是指在出现粘包的时候我们的接收方要进行分包处理。**

**解决方法：自己构建一种数据格式，数据的开头是本段数据的数据长度的int（byte数组表示），每次发送数据时候计算本次消息的数据长度，并且添加到要发送的消息中，解析数据的时候先判断接收到的消息是否满足byte数据数组的开头的那个int所代表的数据长度，>=的情况下再进行分处理数据，否则继续接收数据。**

**接受解析数据处理方法：**

namespace GameServer.Servers

{

class Message

{

private byte[] data = new byte[2048];

private int startIndex = 0;//代表我们已经存储了多少数据

public byte[] Data//data的属性

{

get { return data; }

}

public int StartIndex//startIndex属性

{

get { return startIndex; }

}

public int RemainSize//计算剩余的数组大小

{

get { return data.Length - startIndex; }

}

public void AddCount(int count)

{

startIndex += count;

}

public void ReadMessage(Action<RequestCode,ActionCode,string> processDataCallBack)//这个方法用来解析数据，解决了粘包分包问题

{

while (true)

{

if (startIndex <= 4)

{

return;

}

int l = BitConverter.ToInt32(data, 0);//读取这个data中前四个字节，看看要传过来的信息的长度是多少（他只会读取前四个字节，因为只有四个字节是一个int32）

if ((startIndex - 4) >= l)//表示这个data里面的数据已经是完整的了，或者还多了

{

RequestCode rc = (RequestCode)BitConverter.ToInt32(data, 4);//这一步是把RequestCode解析出来，枚举类型，4字节长度

ActionCode ac = (ActionCode)BitConverter.ToInt32(data, 8);//这一步是把ActionCode解析出来，枚举类型，4字节长度

string s = Encoding.UTF8.GetString(data,12, l-8);//从第12个索引开始后面就都是数据了

processDataCallBack(rc, ac, s);

Array.Copy(data, l + 4, data, 0, startIndex - l - 4);//将读取过的数据删掉

startIndex -= (l + 4);//把startIndex恢复到现在剩余数据的末尾

}

else

{

break;

}

}

}

**发送数据的处理方法：**

public static byte[] PackData(ActionCode ac, string data)//将要返回的数据打包成（数据长度）+（ActionCode）+（返回的消息）这种格式的字节数组

{

byte[] rcBytes = BitConverter.GetBytes((int)ac);

byte[] dataBytes = Encoding.UTF8.GetBytes(data);

int l = rcBytes.Length + dataBytes.Length;

byte[] lBytes = BitConverter.GetBytes(l);

return (lBytes.Concat(rcBytes).Concat(dataBytes)).ToArray();

}