## **Windows API 进程间通信**

**摘要：** 1、动态数据交换(DDE)和网络动态数据交换(NDDE)[1,2] DDE可能是最为强大的进程间通信方式，实现也非常复杂。通信的两端分为服务端与客户端，两端都需要具有窗口。 一个DDE的通信过程大致如下： DDE会话由客户端发起，先初始化一个会话。

1、动态数据交换(DDE)和网络动态数据交换(NDDE)[1,2]<?xml:namespace prefix = o />

DDE可能是最为强大的进程间通信方式，实现也非常复杂。通信的两端分为服务端与客户端，两端都需要具有窗口。

一个DDE的通信过程大致如下：

DDE会话由客户端发起，先初始化一个会话。DDE客户端会向服务端发送一个WM\_DDE\_INITIATE窗口消息。服务窗口收到此消息后，根据条件判断是否需要响应此消息，如果服务端确定可以与客户端建立会话，那么会返回WM\_DDE\_ACK消息。如果客户端成功收到了服务端发送的WM\_DDE\_ACK窗口消息，那么会话就已经建立，之后就进行数据的传输。

数据传输分为很多种方式，可以只传输一个单独的数据项，也可以再建立一个永久数据连接( Permanent Data Link)。

如果进行独立数据项传输，数据传输的单位是数据项。客户端使用WM\_DDE\_REQUEST和WM\_DDE\_POKE消息向服务端接收和发送数据项。数据的传输使用WM\_DDE\_DATA消息。

而永久数据连接的建立也是通过客户端发起，客户端向服务端发送WM\_DDE\_ADVISE消息，服务端根据情况做出回应，根据是否能够建立连接而返回不同的WM\_DDE\_ACK消息。终止数据连接使用消息WM\_DDE\_UNADVISE。

客户端还可以使用WM\_DDE\_EXECUTE请服务端发送命令。

终止一个DDE会话使用WM\_DDE\_TERMINATE消息。

几乎所有的过程都是由客户端发起请求，而服务端响应，服务端响应的方向大多是发送一个WM\_DDE\_ACK消息。

DDE的通信需要操作系统ATOM表的帮助，用于索引数据。操作ATOM表有专门的API。

2、通过File Mapping在进程间共享数据[3]

    共享文件是一种高效的进程间通信方式，因为直接共享内存，所以访问速度非常快。而且允许大量数据传输。

    共享内存的进程间通信与其他进程间通信方式相比，限制条件要少。比如邮槽对数据大小和数据流向有限制，管道只能采用半双工通信形式等，剪贴板容易受其他进程影响等。

使用File Mapping共享数据也有缺点。最大的缺点是无法实现实时性，一个进程无论是写入还是读出数据后，另一个进程都不知道对方的操作是否已经完成。因此在要求实时数据传输时，File Mapping通常与其他进程通信方式配合使用，用于传输大量的数据。

3、Windows Socket

    利用Windows Socket实现的进程间通信借用了Socket通信的原理。其实任何可以在两台主机间进行网络数据通信的机制都可以用于进程间通信。

    如果在一台主机上既使用Socket服务端又实现Socket客户端，那么客户端在连接服务端时将IP指定为本机，就可以实现同一主机上的Socket服务端进程和Socket客户端进程间的通信。

    使用Socket进行进程间通信既可以实现实时性，又可以传输大数据量信息，且数据传输的速度也很快，因此应用范围很广泛。

    在进程间实现Socket通信的原理和程序实现方法与实现不同主机间的Socket通信没有区别。

4.剪切板通信模式。Win\_COPY

5.WIndows消息机制(WIN\_MESSAGE)通信

6.进程间除TCP通信，还有在TCP协议之上的例如IOT-HTT，Redis，MQTT等消息中间件

7.图形和文字识别通信

8.dll注入

9.管道通信

**1、简介**

　    DDE是一种动态数据交换机制（Dynamic Data Exchange，DDE）。使用DDE通讯需要两个Windows应用程序，其中一个作为服务器处理信息，另外一个作为客户机从服务器获得信息。客户机应用程序向当前所激活的服务器应用程序发送一条消息请求信息，服务器应用程序根据该信息作出应答，从而实现两个程序之间的数据交换。

　　DDE（Dynamic data exchange）的工作原理是：

　　甲方申请一块全局内存，然后把内存指针postmessage到乙方,乙方根据收到的指针访问那块全局内存。有几个API函数是做这种事的，你在MSDN里查一下 Dde 打头的函数全都出来了。因为是已经淘汰的技术,连MFC都没对他进行封装。很难保证它同样会出现在以后的Windows API中。

**2、交换方式**

　　（1）冷连接（CoolLink）：数据交换是一次性数据传输，与剪贴板相同。当服务器中的数据发生变化后不通知客户，但客户可以随时从服务器读写数据；

　　（2）温连接（WarmLink）：当服务器中的数据发生变化后马上通知客户，客户得到通知后将数据取回；

　　（3）热连接（HotLink）：当服务器中的数据发生变化后马上通知客户，同时将变化的数据直接送给客户。

　　DDE客户程序向DDE服务器程序请求数据时，它必须首先知道服务器的名称（即DDEService名）、DDE主题名称（Topics名），还要知道请求哪一个数据项的项目名称（Items名）。DDEService名应该具有唯一性，否则容易产生混乱。通常DDEService就是服务器的程序名称，但不是绝对的，它是由程序设计人员在程序内部设定好的，并不是通过修改程序名称就可以改变的。Topics名和Items名也是由DDEService在其内部设定好的，所有服务程序的Service名、Topics名都是注册在系统中，当一个客户向一个服务器请求数据时，客户必须向系统报告服务器的Service名和Topics名。只有当Service名、Topics名与服务器内部设定的名称一致时，系统才将客户的请求传达给服务器。

　　当服务名和Topics名相符时，服务器马上判断Items名是否合法。如果请求的Item名是服务器中的合法数据项，服务器即建立此项连接，建立连接的数据发生数值变化后，服务器会及时通知客户。一个服务器可以有多个Topics名，Items名的数量也不受限制。

　　DDE交换可以发生在单机或网络中不同计算机的应用程序之间。开发者还可以定义定制的DDE数据格式，进行应用程序之间特别目的IPC，它们有更紧密耦合的通信要求。大多数基于Windows的应用程序都支持DDE。但DDE有个明显的缺点就是，通信效率低下，当通信量较大时数据刷新速度慢，在数据较少时DDE较实用。

比如MT4就运用DDE让自己成为服务端，程序可利用DDE获取MT4的外汇报价数据实现实时更新。

using NDde.Client;

private void MainForm\_Load(object sender, EventArgs e)

{

try

{

*//申明并实例化一个DdeClient对象*

client = new DdeClient("MT4", "BID", this);

client.Advise += client\_Advise;

*//连接到DDE服务器*

client.Connect();

*//循环获取数据*

client.StartAdvise("USDCHF", 1, true, 60000);

}

catch (Exception ex)

{

displayTextBox.Text = "MainForm\_Load: " + ex.Message;

}

}

private void client\_Advise(object sender, DdeAdviseEventArgs args)

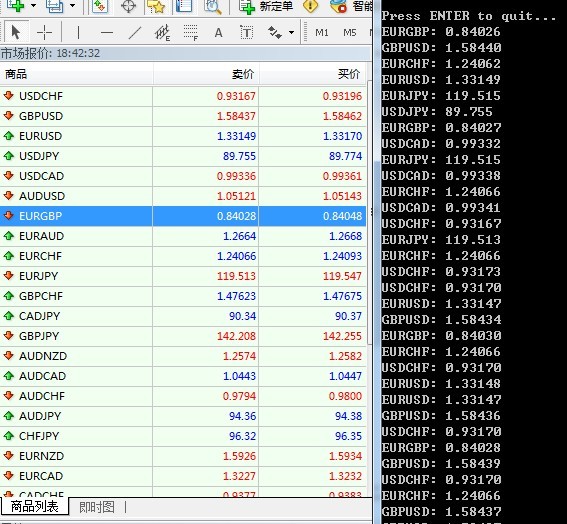
{

*//显示更新数据*

displayTextBox.Text = "OnAdvise: " + args.Text;

}

这样就能实时获取MT4服务器的数据了。



因为MT4支持DDE通讯，在MT4的底层代码里，作者加入了DDE通讯循环广播，所以外部只要是实现了该协议的任何软件和插件，都可以接收到MT4的实时行情源，中间的通讯协议就是DDE，DDE和EXCEL无关，是一种协议，而EXCEL恰也是实现了该协议的软件，所以EXCEL也支持订阅和广播DDE协议数据，所以MT4不仅可以和实现了DDE协议的客户端交互，还可以和实现了该协议的EXCEl交互，本质上，只要该插件或软件支持或者实现了DDE协议，都可以互相交互。

同理，multicharts中文版-英文版都实现在以MC作为客户端----数据接收方的功能，接收数据DDE协议订阅，即无论外部是任何插件或者软件，只要按照DDE协议广播了特定的service和item的广播信心，multciharts作为客户端，只要按DDE协议实现并订阅，都可以收到外部的任何行情注入。此作为数据交换。

另外除了自定义软件程序作为client和服务端MT4DDE对接外，本身multcicharts也可以作为DDE客户端，所以，multicharts可以和MT4实现直接对接，截图如下：



此DDE-SImple为客户端和服务端的双实现，既可以用于发送DDE广播，也可以接受特定订阅数据的服务端和客户端原理解析。

所以对于multichats，因为他只能在QM里面设置一个Universole-DDE数据源，所以，其实可以在外部接收数据，比如股票，期货，外汇，外盘，债券，数字货币等，无论任何数据，然后在中间件软件的接收之后的帮助下，注入到multciahtrs中完成信号分析和分析，其余历史数据可以分品种导入即可，也可以打开图表之后，外部中间件完成一次历史数据广播，将历史数据也注入进去。历史数据又中间件软件去下载获取。

所以，理论上，可以完成无论是金融数据，还是其他任何数据在MC中的分析。全部通过中间件DDE作为服务端和multciahrs进行交互。

而英文版本的multichatys.net版本因为是和visual studio.net无缝对接的，所以在mc.net中理论上可以调用所有windows接口API，完成和外部的直接连接，包括交易，还可以外部行情注入靠DDE，而实盘交易，可以在.net的公式编辑器中调用各个交易源的交易接口完成信号下单，或者外部交互。无论对于交易，还是组合监控，都是非常好的方式，可以无缝享受mc的所有策略回测和行情分析，策略编辑功能，完成分析。

具体亦可见-multicharts官方网站视频对接。

。Net版本还有C++版本已经封装好了交互的dll，用于开发。.net为NDde.