COM+

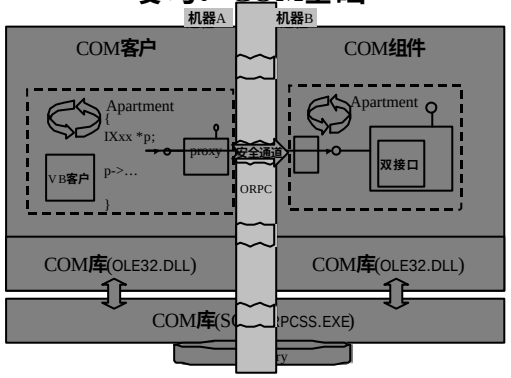
潘爱民

http://www.icst.pku.edu.cn/compcourse

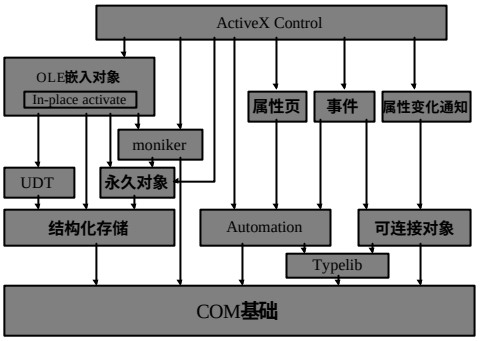
内容

- COM**复习**
- COM+**介绍**
- COM+**结构**
- COM+**服务**
- COM+**开发**

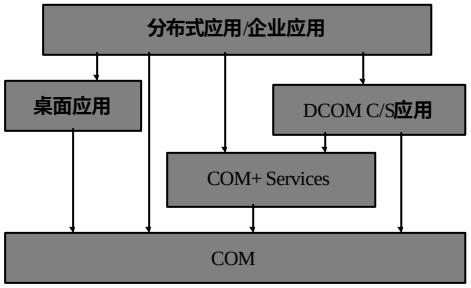
复习：COM基础



复习：COM技术总体结构



COM和COM+



COM+：企业应用

- **从桌面应用发展到企业应用、Web应用**
  - **企业应用特点：**
    - **大型，分布式，实现企业的关键业务**
- **企业应用的挑战**
  - **关键业务：业务逻辑**
  - **基础设施：涉及到系统方方面面**
    - **开发，调试，测试，配置，维护**
    - **如**transaction，security，event**等**

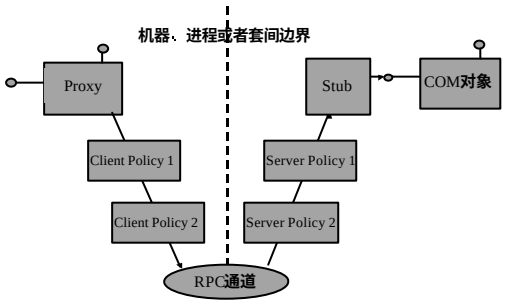
COM+是什么?

- 面向企业应用
  - 提供了企业应用所需要的通用基础设施
  - 设计企业应用要求有思想上的根本变化
- COM+对COM的增强和改进
  - 增强了COM runtime library
  - 增强了MTS(Microsoft Transaction Server)
    - 事务服务，安全服务
  - 提供了新的runtime服务
    - 队列服务，事件服务，负载平衡，(内存数据库)
- COM+兼容COM，只是提供了更多的功能

COM+的设计思想、目标

- 使程序员和用户把注意力和精力集中在业务逻辑上，而不是基础设施上
- 使组件尽可能地通用
  - 编译时刻特性与运行时刻特性分离
- 与以前的COM组件兼容
- 客户和组件的透明性

COM+：interception技术



COM+组件结构

- COM+组件结构
  - COM+ Application —— 管理基本单元
  - COM+ Components —— 功能基本单元
  - COM+ Objects
- COM+ Application
  - 分为两类：
    - Server Application，DllHost.exe
    - Library Application，与客户共享进程，仍然截取
  - 共享同一个进程，同一组属性设置
- COM+ Components为COM进程内组件
- COM+ Objects为COM进程内对象

COM+管理工具

- Component Services snap-in



客户创建COM+对象

- 对客户透明，使用增强了的COM库
- 客户创建COM对象
  - COM+ runtime services会检查COM+的配置信息
  - 启动指定的服务进程
  - 安装interceptor，截取器
  - 在客户方创建代理对象，并返回给客户
- COM+根据指定的应用属性管理服务进程的生命周期

COM+对于组件的要求

- 要求DLL形式的组件
- 要求自注册
  - DllRegisterServer，DllUnregisterServer
- 要求有typelib
  - typelib既可以绑在dll上，也可以单独的文件
  - typelib ——》 component library

COM+ Catalog(COM+目录)

- COM+管理信息，系统数据库，包含两部分
  - COM+ Registration Database
  - Windows Registry (HKEY_CLASSES_ROOT)
- 通过一个对象层次结构管理COM+ Catalog
  - COM+ Administration Library，支持脚本
- 组件服务snap-in管理工具就是该对象模型的一个功能缩影
- 用户可以通过COMAdminCatalog组件进入到对象模型中
  - ProgID：COMAdmin.COMAdminCatalog

COMAdminCatalog组件使用举例

```
Dim Catalog As COMAdminCatalog
Dim Applications As COMAdminCatalogCollection
Dim Components As COMAdminCatalogCollection
Dim AppObject As COMAdminCatalogObject

' get the Applications collection and populate it
Set Applications = Catalog.GetCollection("Applications")
Applications.Populate

' get the correct application, My Application
For Each AppObject in Applicationss
    If AppObject.Name = "My Application" Then Exit For
Next

' get the Components collection for My Application
Set Components = Applications.GetCollection( "Components" , AppObject.Key )
```

COM+ Context(COM+环境)

- 对象所生存的COM+状态
  - COM激活对象时，创建context
  - 对象的环境与对象紧紧绑在一起
  - 根据组件的属性决定环境的状态
  - 环境属性也是对象与客户之间的约定
- 如何访问“环境对象”
  - CoGetObjectContext
  - GetObjectContext

COM+环境(续)

- 环境对象支持接口
  - IObjectContext
  - IGetContextProperties
  - IObjectContextInfo
  - IContextState
  - ISecurityCallContext
- 环境可以扩充
  - 比如ASP对象

COM+安全性

- 安全性基础设施对于分布式应用至关重要
- Authentication —— 认证(鉴定)
  - Authentication Level
  - SSP/SSPI ——允许扩展
  - NTLM/Kerberos
  - CoInitializeSecurity
  - 安全与性能的平衡



COM+安全性(续一)

- 授权(Authorization)
- Role-based
  - 仅限于通过截取器的调用
- 通过程序来控制 ISecurityCallContext 和IObjectControl提供了
  - IsSecurityEnabled
  - IsCallerInRole
- 进一步有ISecurityProperty

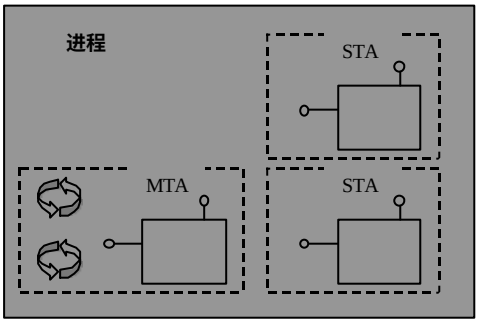


COM+安全性(续二)

- 服务进程的身份(identity)
  - DCOM方案：登录到客户机上的用户 ——不再有效
  - 当前交互用户或者每个指定的用户
- 三层结构的两种安全模型
  - trusted server model
  - impersonation/delegation model



COM+线程模型：STA和MTA



COM+线程模型：TNA

- TNA(Thread neutral apartment)
  - 不包含线程，只有对象，所有的对象与线程无关
- TNA中的对象
  - ThreadingModel=Neutral
  - 创建线程总是接收到一个轻量级的代理
  - 调用时，直接在调用线程中执行
- TNA有希望替代“Free”和“Both”类型
- STA继续生存，特别是包含UI的对象

同步(synchronization)

- 或者serialization
  - STA对象不需要同步，MTA，TNA对象需要同步
- COM+提供了“activity -based synchronization”
  - activity：代表单个客户执行任务的一组对象
  - 相当于一个逻辑线程
  - 每个activity都包含一个进程范围内的锁
  - 可以处理跨越机器边界的嵌套调用



Transaction(事务)

- 分布式应用最困难的一个基础设施
- 定义：由一组相互关联的操作构成的整体行为
- 难点所在：保持系统状态一致性
  - 在操作过程中，发生意外或者失败
  - 操作的中间状态要被隔离开
  - 组件设计思想：把大的操作分解开，并可能包装到小的组件中，这加剧了事务的困难，因为：在编写组件时需要处理由其他组件引起的所有问题

事务的技术要点

- DTC(Distributed Transaction Coordinator)
  - 每个组件对DTC负责，DTC知道所有的组件
- 两阶段提交
  - 准备和提交
  - 对象表态：committing，aborting

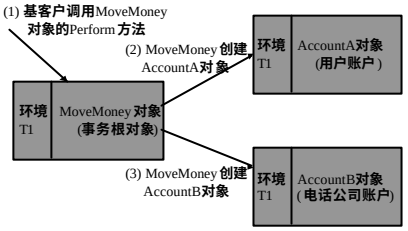
COM+中的事务

- COM+ runtime提供了免费的DTC
- 组件与DTC的通讯
  - IObjectContext::SetComplete
  - IObjectContext::SetAbortComplete
  - 组件的属性设置

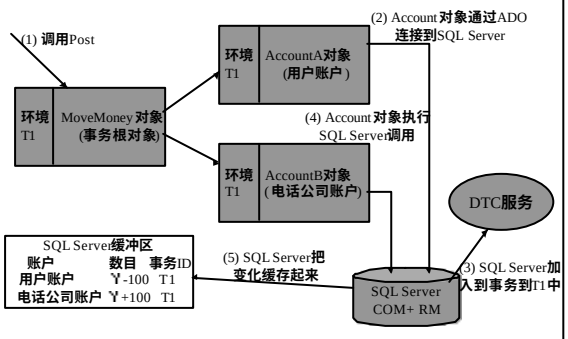
Bank例子



事务与对象



事务过程



COM+ RM

- COM+ RM(Resource Manager)
  - 知道当参与事务的对象改变它的状态时，该如何把资源的变化缓存起来
  - 如果事务被提交，RM知道如何改变资源状态
  - 如果事务被放弃了，RM知道如何把变化丢掉
  - 例如：各种数据库系统，MSMQ
- DTC两阶段提交过程
  - 对象不参与提交过程，只是表态
  - 准备阶段：RM把状态信息保存起来，保证不会失败
  - 提交阶段：DTC通过RM提交事务

事务的特性(ACID)

- Atomicity
  - 要么所有的变化都有效，要么都无效
- Consistency
  - 状态一致性
- Isolation
  - 事务外部看不到事务的中间结果
- Durability
  - 缓存的中间结果要能“持久”

COM+事务的用法(一)



COM+事务的用法(二)



COM+事务的用法(三)

```
[
        uuid(04CF0B76-1989-11D0-B917-0080C7394688),
        helpstring("Account Class"),
        TRANSACTION_REQUIRED
]
coclass CAccount
{
        [default] interface IAccount;
};

[
        uuid(04CF0B7B-1989-11D0-B917-0080C7394688),
        helpstring("MoveMoney  Class"),
        TRANSACTION_REQUIRED
]
coclass CMoveMoney
{
        [default] interface IMoveMoney;
};
```

事务属性

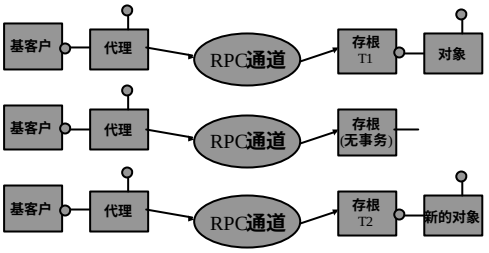
- 不支持
- 支持
- 需要
- 需要新建
- 禁用

对象如何控制事务

- 两个控制位
  - transaction-vote bit：表态位
  - deactivate-on-return bit：返回之后，对象即无效
- 通过环境对象控制这两位
  - IObjectContext
  - IContextState

Transaction and Just-In-Time activation

- 事务组件必须使用JIT activation特性



JIT activation的意义

- 资源利用 —— 可伸缩性
  - 系统帮助回收资源
  - 回收的价值：部分回收所占整体的比例，以及资源本身的重要性(比如数据库连接)
- 保证事务的正确性
  - 一致性和隔离性
- 与对象池的结合
- 状态管理新概念

事务与对象池

- 池组件
- 对象必须实现 IObjectControl接口
  - Activate
  - Deactivate
  - CanBePooled
- 池组件满足以下条件：
  - 线程模型不能为STA
  - 可被聚合
  - 不能聚合其他的池组件



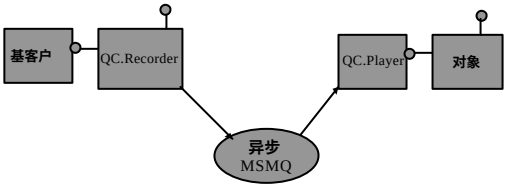
补偿资源管理器 (compensating resource manager)

- 补偿事务
  - Undo，一个对等的反操作
  - 缺点：破坏隔离性
- RM之于事务
- CRM
  - 用到了COM+提供的关于CRM支持的对象
  - 模型：
    - CRM Worker要求一个事务，连接到CRM Clerk
    - CRM Clerk负责与各方面的通信，包括DTC
    - CRM compensator完成两阶段提交的工作
    - DTC控制整个事务过程

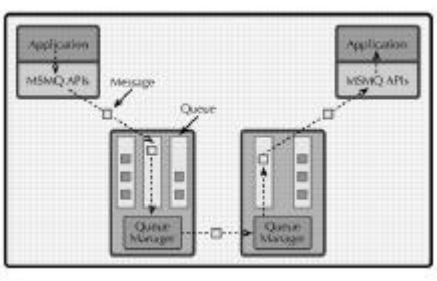
队列组件(Queued Components)

- 在分布式系统中，
  - 同步，异步并存
  - 企业活动中，电话，电子邮件，电话留言
- 通讯双方在时间点上的分离
  - 例如：在off-line下写email，然后放到outbox中，一旦on-line，邮件就被发送出去
  - 资源分配更为合理(通过设置优先级别)，从而提高资源利用率和可伸缩能力
  - 不必要求所有的应用系统都在同一个时刻运行，从而简化分布式应用的开发
- 队列组件
  - COM+为这种通讯方式提供了一套基础设施

QC模型



MSMQ基础



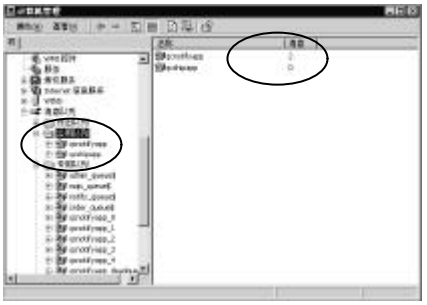
QC的意义

- 对于组件透明
  - 同样的组件既可以与Queue结合，也可以不结合
  - 对于客户不完全透明
- 避免了MSMQ编程的复杂性
  - 编写QC无需任何编写MQ的经验
  - 缺点：缺少控制能力

QC的使用



QC的消息队列



客户代码

```
Dim Ship As Object
Set Ship = GetObject("queue:/new:VBShipProj.VBShip")

Ship.CustomerID = 1111
Ship.OrderID = 2222
Ship.LineItem 12, "Drummers drumming"
Ship.LineItem 11, "Pipers piping"
Ship.LineItem 10, "Lords a leaping"
Ship.LineItem 9, "Ladies dancing"
Ship.LineItem 8, "Maids a milking"
Ship.LineItem 7, "Swans a swimming"
Ship.LineItem 6, "Geese a laying"
Ship.LineItem 5, "Gold rings"
Ship.LineItem 4, "Calling birds"
Ship.LineItem 3, "French hens"
Ship.LineItem 2, "Turtledoves"
Ship.LineItem 1, "A partridge in a pear tree"

Ship.Process
```

QC的设计

- 接口设计
  - 禁止使用[out]，[in,out]，[retval]
  - 必须使用按值传递参数方式
  - 客户对一系列组件方法的调用必须能够满足自己功能的需要，例如，不能使用枚举器对象
  - 不能返回与应用相关的错误码，返回值只表示QC.Recorder是否记录成功。
- 如何接收输出
  - 没有输出
  - 对某些调用要求有应答
    - 用同样的异步机制QC
    - 传递一个也具有QC特性的回调对象
  - 在设计软件的时候应该尽量避免输出的必要性

QC的设计(续)

- QC与事务的结合
  - MSMQ本身就是一个RM
  - 在客户端QC与事务的结合
  - 在服务器端QC与事务的结合
- QC的安全性
  - 与同步方式的认证过程不同
  - 每个消息包含数字签名

QC与Moniker

- 使用Moniker创建QC对象，例如

  GetObject("queue:/new: VBShipProj.VBShip")
  - 封装了组件的创建过程，使组件对于两种方式透明
- queue名字对象

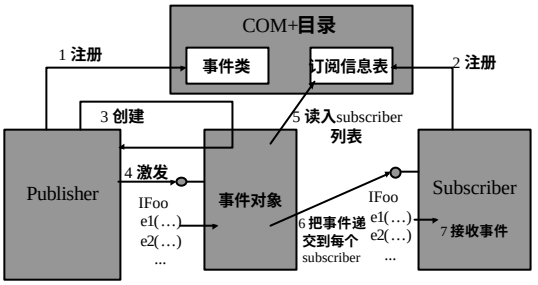
  GetObject("queue:ComputerName =SomeMachine/new: VBShipProj. VBShip")
- new名字对象

  GetObject("new:VBShipProj.VBShip")

COM+事件模型

- 事件模型中用到的两个概念
  - publisher：发布者
  - subscriber：订阅者
- 变化通知
  - 轮循法
  - 可连接对象 —— tightly coupled event
  - COM+的方案：loosely coupled event model

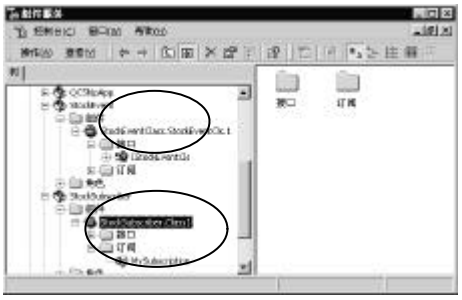
COM+事件模型结构



安装新的事件类



Subscriber创建一个subscription



Publisher示例代码

```
' Create the event class object

Dim foo As New StockEventCls

' Call the event class object's method that
' fires the StockPriceChanged event.

Private Sub Command1_Click()
   Call foo.StockPriceChanged(Text1.Text, CCur(Text2.Text))
End Sub

' Call the event class object's method that
' fires the NewStockListed event.

Private Sub Command2_Click()
   Call foo.NewStockListed(Text1.Text)
End Sub
```

Subscriber示例代码

```
' The subscriber must implement the interface on
' which it wants to receive calls from the event object
Implements StockEventCls

' Pop up message box telling user of
' NewStockListed event

Private Sub StockEventCls_NewStockListed(ByVal Symbol As String)
   MsgBox "New Stock " + Symbol + " has started trading", vbOKOnly, _
          "VB Event Subscriber"
End Sub

' Pop up message box telling user of
' StockPriceChanged event

Private Sub StockEventCls_StockPriceChanged(ByVal Symbol As String, _
                  ByVal Price As Currency)
   MsgBox "The market price of stock " + Symbol + " is now " _
                  + Str(Price), vbOKOnly, "VB Event Subscriber"
End Sub
```

subscription

- **两种形式**
  - persistent
  - transient

```
Set oCOMAdminCatalog = CreateObject("COMAdmin.COMAdminCatalog.1")
'Get the TRANSIENTSUBSCRIPTIONS collection

Set oTSCol = oCOMAdminCatalog.GetCollection("TransientSubscriptions")
Set oSubscription = oTSCol.Add

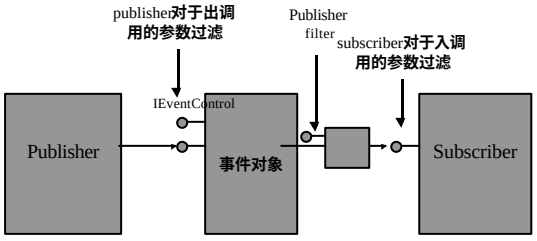
Set objvar = objref
oSubscription.Value("SubscriberInterface") = objref
oSubscription.Value("EventCLSID") = clsid
oSubscription.Value("Name") = "TransientSubscription"

oTSCol.SaveChanges
```

关于事件

- **事件方法调用返回**
  - **对于非QC组件，表明事件方法已经被调用**
  - **对于QC组件，表明事件调用已经被送到队列中**
  - **返回值仅表明该事件提交的大致情况，不针对任何一个订阅者**
- **对于多个订阅者的情形**
  - **提交顺序不确定**
  - **可以设置并发提交**

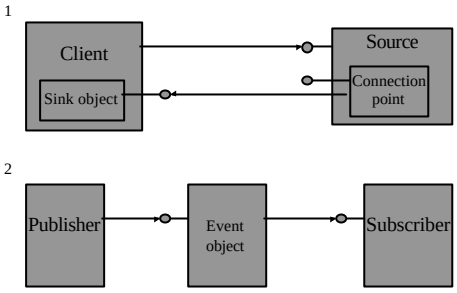
事件的过滤机制



事件服务与其他服务的结合

- **与队列组件的结合**
- **与事务的结合**
  - **如果有多个订阅者，有的支持事务，有的不支持事务，则不支持事务的订阅者的动作无法回滚**
- **事件的安全性**
  - **标准的COM+安全机制**
  - **订阅者可以以库应用方式运行**

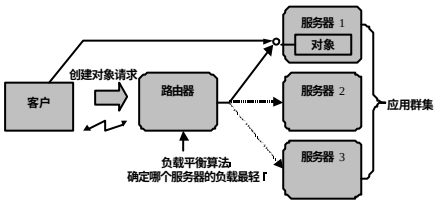
COM+事件模型与源对象模型对照



COM+的其他服务

- IMDB(In-Memory Database)
  - 内存缓冲，以空间换时间策略
- CLB(Component Load Balancing)
  - 已移到“Application Center Server”中
  - Windows 2000 Advanced Server中保留
  - 与clustering service一起提供high-availability and scalability

CLB结构示意图



COM+与其他系统服务

- COM+与Active Directory Service
- COM+与Security Sevices
  - Kerberos和PKI
- COM+与Data Access Services
- COM+与DTC
- COM+与MSMQ
- COM+与Web Services

COM+开发

- 在现有工具下，基本没有变化
- 新的工具会有新的支持
  - 底层原理不变，但是包装会做得更好
- 分两个方面
  - COM+组件的开发，针对每一种服务有一些新的要求，大多数保持与原来开发模式的兼容性，新的特性多数可以通过配置获得
  - 与COM+系统提供的各种服务和管理工具打交道

参考资料

书：《Understanding COM+》(配有例子)

书：即《深入理解COM+》

书：《Understanding Microsoft Windows 2000 Distributed Services》

电子资源：MSDN-Library