Goals

Show by example that COM+ makes it:
• As easy to develop Server Components as it is to develop Client Components
• As easy to deliver enterprise apps as it is to deliver workgroup apps!
Technology Scenario

The caring, sharing environment for your components

COM+ Services:
- Events
- Security
- Load Balancing
- Queued Components
- In-Memory Database
- Compensating Resource Manager
- Administration

MTS Services:
- Transactions
- Resource Pooling
- Security
- Administration

COM
- The Model
- Tools Support
- Multi-Language Discovery (QI)

Application Scenario

- Online service like MSN or AOL
- User login causes two databases to be updated
  - Customer Billing Log
  - Customer Login Status
- Total connect time is tracked
- User is notified of various activities while they are logged in (i.e. chat rooms)
Application Requirements

- Ultimate customer response and availability
- Always accept Logon
- No freebies! Always bill for entire time online
- Use existing billing system
- Use existing dev resources

Many others…

- Keep billing, Customer Online and other systems consistent
- Real-time two way info on who is on
- Real-time info on what is going on
- Flexible administration
- Maximize use of platform, minimize development time

Tackling The Application Requirements

- Ultimate customer response and availability
- Always accept legitimate login
- No freebies! Always bill for entire time online
- Use existing billing system
- Use existing dev resources
- Others…

- Keep billing, Customer Online and other systems consistent
- Real-time two way info on who is on
- Real-time info on what is going on
- Flexible Administration
- Maximize use of platform, minimize development time
Requirement: Simplify Server Side Development

• Issues:
  – Threading
  – Synchronization
  – Data Base Connection pooling
  – Communications - front and back ends

Requirement: Simplify Server Side Development

• Solution: core COM+ Services
  – You write a “single user” component as though it were the only user of DB resources
  – COM+ provides the rich service environment which makes your component multi-user, multi-threaded and resource pooled
Core Application Services

- COM+ provides:
  - Administrator controlled process placement
  - The context
  - Sharing automatically managed by COM+
  - Auto Completion (or explicit SetComplete, SetAbort)

- All you have to do:
  - Stick with the model (or be aware of consequences of leaving model)
  - (Basic component walkthrough)

Tackling The Application Requirements

- Ultimate customer response and availability
- Always accept legitimate login
- No freebies! Always bill for entire time online
- Use existing billing system
- Use existing dev resources
- Others…
- Keep billing, Customer Online and other systems consistent
- Real-time two way info on who is on
- Real-time info on what is going on
- Flexible administration
- Maximize use of platform, minimize development time
Flexible Administration

• Issues
  – Different Administration dimensions
  – Application Distribution
  – Connectivity

Now!

Flexible Administration

• Solution: COM+ Administration
  – Features: extends administration to all COM Components, supports administration of new services and provides a complete programmatic interface for installation, configuration and deployment.
Administrative Model

• Applications are defined and components are installed in the applications that they are to run in
• Attributes are set declaratively on the components
  • Transactions
  • Security roles
  • Activation properties
• SDK - complete automation capabilities

Administration Example

• Walkthrough of Billing system deployment
• Billing components are updated on the dev machine
• The Application can be exported manually:
  – Application information and the DLLs to be moved to the accounting server and…
  – ActiveX® Control is downloaded to the client using the CODEBASE tag and a CLSID
  – Or…
• Auto download of components on demand
  • Requires Windows 2000 Directory Service
Tackling The Application Requirements

- Ultimate customer response and availability
- Always accept legitimate login
- No freebies! Always bill for entire time online
- Use existing billing system
- Use existing dev resources
- Other...

- Keep billing, Customer Online and other systems consistent
- Real-time two way info on who is on
- Real-time info on what is going on
- Flexible administration
- Maximize use of platform, minimize development time

Database Consistency

- Issues:
  - Updating multiple Resource Managers
    - (DB, IMDB, MSMQ)
  - Avoiding conflicts
  - Multiple simultaneous users
  - Indeterminate numbers of users

Diagram:
- BusCustomer Login
- dbCustomer update
- Billing Database
- Other Data Sources
- dbCustomer Online
- Customer Database

17
18
Database Consistency

- Solution: COM+ Transaction Services
  - Features:
    - System provided transactions on activation, commit on deactivation, abort on errors
    - System enforced serialization
    - Manual (SetComplete, SetAbort) or Auto completion

Consistency Example

- COM+ Transaction Services
  - Transaction code and attribute walkthrough
  - Bus_Customer calls db_CustomerBilling and db_CustOnlineStatus
  - Distributed transaction occurs (transparently) when user logs on
  - Without Transactions customer could be billed for time even if they failed to complete the logon or…
  - They might not get billed even though they successfully logged on
  - Auto Completion (not in the Preview)
  - Transaction code and attribute walkthrough
Tackling The Application Requirements

- Ultimate customer response and availability
- Always accept legitimate logon
- No freebies! Always bill for entire time online
- Use existing billing system
- Use existing dev resources
- Other...

- Keep billing, Customer Online and other systems consistent
- Real-time two way info on who is on
- Real-time info on what is going on
- Flexible administration
- Maximize use of platform, minimize development time

Application Security

- Issues:
  - Security Dimensions
  - Distributed application
  - Components being Compounded
Application Security

• Solution: COM+ Security Services:
  – Role-based access control on classes, interfaces and METHODS,
  – mapping roles to Windows 2000 domain accounts

Security Example

• Application context
  – Customer billing information needs to be protected
  – When a user logs on the bus_Customer component running on the Web server will call the accounting server
  – Using roles the Web server will be allowed to call the accounting server but you may not call the component directly
  – Roles established in package definition
  – Roles established in the IDE
  – Code and Attribute walkthrough
Tackling The Application Requirements -
The Story So Far

• Ultimate customer response and availability
• Always accept legitimate logon
• No freebies! Always bill for entire time online
• Use existing billing system
• Use existing dev resources
• Other…

• Keep billing, Customer Online and other systems consistent
• Real-time two way info on who is on
• Real-time info on what is going on
• Flexible administration
• Maximize use of platform, minimize development time

Tackling Other Application Requirements

• Ultimate customer response and availability
• Always accept legitimate logon
• No freebies! Always bill for entire time online
• Use existing billing system
• Use existing dev resources
• Other…

• Keep billing, Customer Online and other systems consistent
• Real-time two way info on who is on
• Real-time info on what is going on
• Flexible administration
• Maximize use of platform, minimize development time
Even Easier!

• Coming soon
  – Direct manipulation of attributes from IDE
  – Direct manipulation of COM+ administration from IDE
• The COM+ Wizard

Conclusion

COM+ makes it:
• As easy to develop Server Components as it is to develop Client Components
• As easy to deliver enterprise apps as it is to deliver workgroup apps!
Call To Action

• Architecting, designing, developing:
  – Assume COM+ level of service and use Windows 2000 now

• Deploying and in production:
  – Use Microsoft Transaction Server now - your components will be uplifted when COM+ is installed

• Everything discussed is a standard feature of Windows 2000

References on COM+

• Books
  – Chappell. Understanding ActiveX and OLE
  – Eddon. Inside DCOM
  – Grimes. Professional DCOM
  – Box. Essential COM

• COM Home Page
  – http://www.microsoft.com/com/

• Windows 2000 Platform SDK
  – NT 5.0 Beta 2 or later
Questions?

Addendum
COM+ Technology
More Details on COM+

- Review: COM and MTS Today
- COM+ Architecture

Review

COM Architecture

- Object Creation
  - In-process
  - Remote
- Calling an Interface Method
Scenario
COM Architecture

main() {
    IAS15Demo* pInt;
    CoInitialize(NULL);
    CoCreateInstance(CLSID_AS15Demo, NULL,
                     CLSCTX_SERVER,
                     IID_IAS15Demo, &pInt);
    pInt->DoSomething();
    ...
}

Object Creation
In-Process

Client Process

- Client
- COM Runtime
- SCM

Component

- object

Registry

- {CLSID}
- Server.DLL

Numbers:
1. Client to SCM
2. SCM to COM Runtime
3. Registry to Component
4. Component to Object
5. Object to Client
Object Creation (Remote)

1. Client requests a component from the COM Runtime.
2. The COM Runtime looks up the component in the SCM.
3. The SCM returns the location of the surrogate process.
4. The COM Runtime requests the surrogate object from the surrogate process.
5. The surrogate process returns the surrogate object to the COM Runtime.
6. The COM Runtime requests the component from the surrogate object.
7. The surrogate object returns the component to the COM Runtime.

Calling An Interface Method

1. Client App requests a method from the In-Process Object through the In-Process Server.
2. The In-Process Server makes a RPC call to the Local Object Proxy.
6. The Local Object on the Local Server Process makes a RPC call to the COM on the Local Server Machine.
10. The Local Object on the Local Server Process makes a RPC call to the COM on the Local Server Machine.
13. The Local Object on the Local Server Process makes a RPC call to the COM on the Local Server Machine.
18. The Local Object on the Local Server Process returns the result to the COM on the Local Server Machine.
19. The COM on the Local Server Machine returns the result to the Local Object Proxy.
20. The Local Object Proxy returns the result to the In-Process Server.
21. The In-Process Server returns the result to the Client App.
Review
MTS Architecture

• Object Creation
  – From base client
  – From component
• Calling an Interface Method

Scenario: MTS Architecture
Base Client

```c
Main() {
    IAS15Demo* pInt;
    CoInitialize(NULL);
    CoCreateInstance(CLSID_AS15Demo, NULL,
                     CLSCTX_SERVER,
                     IID_IAS15Demo, &pInt);
    pInt->CreateSubObject();
    ...
}
```
Scenario: MTS Architecture
Component

CAS15DemoImpl::CreateSubObject() {
    IObjectContext* pCtx;
    IAS15DemoSub* pSub;
    GetObjectContext(&pCtx);
    pCtx->CreateInstance(CLSID_AS15DemoSub,
                          IID_IAS15DemoSub, &pSub);
    pSub->DoSomethingElse();
    ...
}

Object Creation
From Base Client

1. Client
2. COM Runtime
3. Registry
4. MTS Surrogate
5. Component
6. Class factory wrapper
7. Object context
8. Context wrapper
9. MTS Executive

{CLSID} MTX {package}
Object Creation
More about Context

MTS Surrogate

Object context
- Creator ID
- Activity
- Transaction
- Security Properties

MTS Catalog

MTS Surrogate Process

Component
- Object
- Object Context

MTS Executive
- Context wrapper
- Object context

Sub Component
- Class object

Object Creation
From Component
Calling An Interface Method

COM+ Architecture

- What is COM+?
- Concepts
- Activation
- Interception
- Services
What Is COM+?

• Unification of COM and MTS
• Plus...
  – Extensions to base COM
  – Extensions to MTS services
  – New Services
• Minimal impact on how you write components
• Remoting architecture doesn’t change (DCOM)

Applications And Classes

• COM+ Applications
  – Evolution of MTS Packages
  – Library applications run in-process
  – Server applications run in COM+ surrogate
• COM+ Classes
  – Configured classes
    • Installed in application
    • Can use COM+ Services
  – Unconfigured classes
    • Everything else...
COM+ Thread Types

• Single-Threaded Apartment (STA)
  – Synchronized using windows messages
  – Call that leaves thread causes message loop to execute
• Multi-Threaded Apartment (MTA)
  – No synchronization
  – Call leaving thread blocks further execution on thread
  – RPC worker threads are this type

COM+ Apartment Types

• Apartment
  – Collection of objects and their contexts
• Single-Threaded
  – Objects execute on STA thread where they were created
• Multi-Threaded
  – Objects execute on any MTA thread
• Neutral (New!)
  – Objects execute on caller’s thread type
COM+ Domains

- Independent aspects of an execution environment
- Objects with same aspect share a domain
  - Locality (machine, process, apartment)
  - Service specific (e.g. transaction T1,...)
- Arbitrarily intersect and nest

COM+ Context

- An intersection of domains
  - Boundary around objects sharing the same properties
- Every object lives in a context
  - Non-default context
    - Objects from configured classes
  - Default context
    - Objects from unconfigured classes
- The basis for COM+ Services
Scenario: COM+ Class
Base Client

Main() {  
IAS15Demo* pInt;
CoInitialize(NULL);
CoCreateInstance(CLSID_IAS15Demo, NULL,
CLSCTX_SERVER,
IID_IAS15Demo, &pInt);
pInt->CreateSubObject();
...  
}
Scenario: COM+ Class Component

```cpp
CAS15DemoImpl::CreateSubObject() {
    // IObjectContext* pCtx;
    IAS15DemoSub* pSub;
    // GetObjectContext(&pCtx);
    // pCtx->CreateInstance(CLSID_AS15DemoSub,
    //     IID_IAS15DemoSub, &pSub);
    CoCreateInstance(CLSID_AS15DemoSub, NULL,
    CLSCTX_SERVER,
    IID_IAS15DemoSub, &pSub);
    pSub->DoSomethingElse();
    ...
}
```

No special creation API required. Context flows automatically

Activation
From Base Client, In-Process
**COM+ Catalog**

- Unifies and extends COM, MTS registration
- Activator queries Catalog
- Catalog supports two data stores
  - Registry (HKCR) for unconfigured classes
  - RegDB for configured classes
- RegDB populated from…
  - Meta-data in self-describing components
  - Administrative tools
Self-Describing Component

- Type library (TLB)
  - Class, interface, method signatures
- Component Library (CLB)
  - Attributes and registration information
    - Transaction, Synchronization, Queueable, Loadbalanceable, Roles
    - ThreadingModel, Component Categories, FileExt
- Requires Developer Tool Support
  - IMetaDataEmit/IMetaDataImport

Activators

- Extensible architecture for object creation
- Activator chain determines...
  - Object location
  - Context properties
- COM+ provides system activators and defines chain order
Context Revisited

- Context contains a set of context properties
- Context properties can flow from client to new object’s context
- Once initialized, context is immutable
Activation
From Component

Component → COM+ Runtime
1. Activator
2. Catalog
3. Prototype Context Properties
4. Identical Sub Component

Activation
From Component

Component → COM+ Runtime
1. Activator
2. Catalog
3. Context Identical
4. Sub Component

63

64
Activation
From Component

Interception
Calling an interface method
COM+ Services

- Build on unified architecture
- Lots of PDC talks on Services…
  - AS12, AS13 -- Services Overview
  - AS09 -- Security
  - AS16 -- Object Pooling
  - AS07 -- Events
  - AS08 -- Load Balancing
  - AS10 -- Queued Components
  - AS11 -- In-Memory Database
  - AS06 -- Deployment, Administration

Summary

- COM+ unifies COM and MTS
  - Extensible activation architecture
  - Extensible interception architecture
- COM+ extends COM and MTS
  - Services build on unified architecture
  - Additional calling models available
Questions?