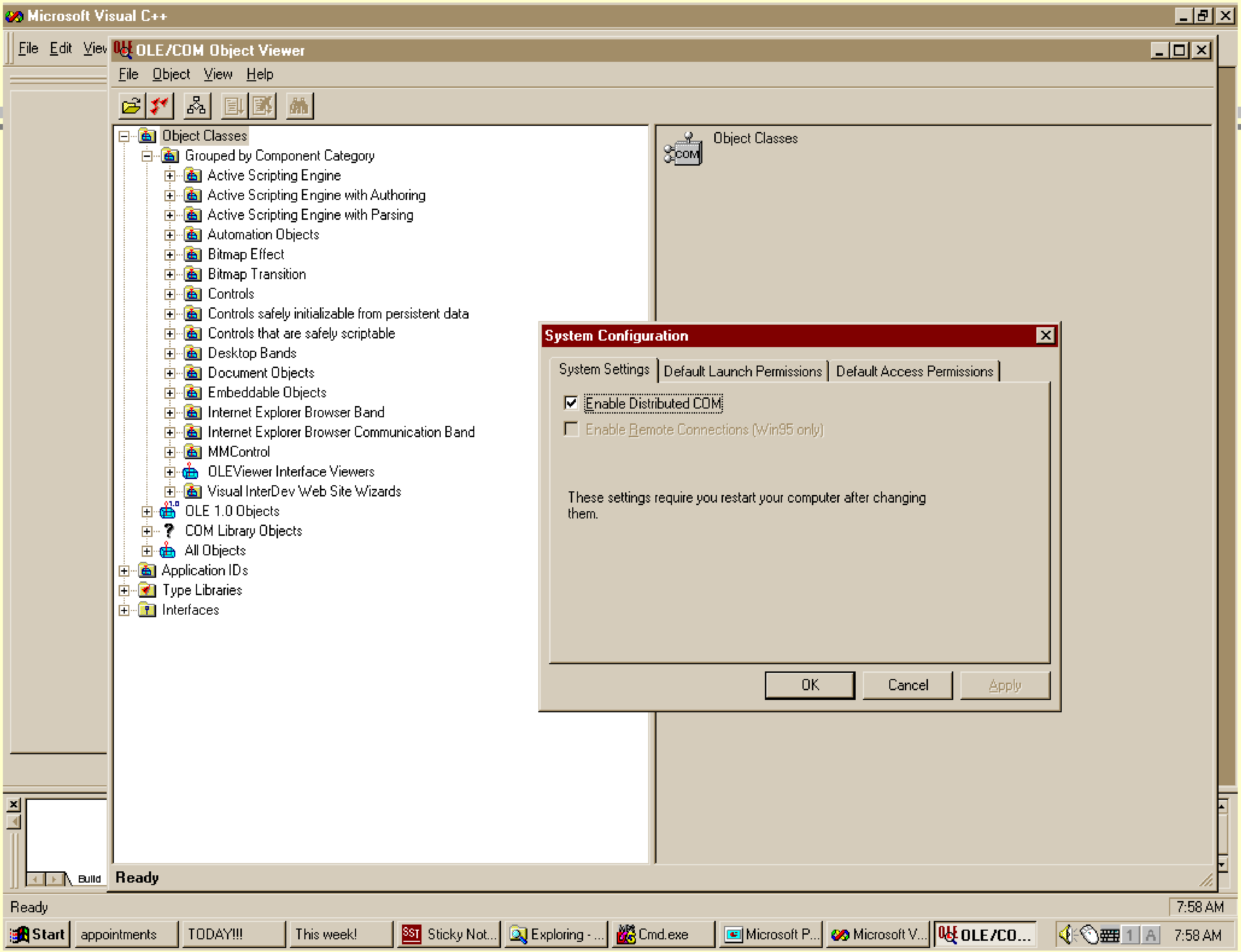
Putting the Distributed into COM

Jim Fawcett
CSE775 - Distributed Objects
Spring 2003

Network Configuration

- Using NT administered domains
  - make sure printer and file sharing are on
  - using OLE/COM Object Viewer from the Visual C++ Tools menu (oleview.exe):
    - check that Distributed COM is enabled
    - check the Default Launch Permissions. If your group, e.g., one of the NT users groups, is not specified then you will have to manually start the server on the remote machine
    - check the Access Permissions. Usually everyone has access permission.
  - using the NT Task Manager check that the service control manager is running - RPCSS.EXE (DCOM Server Process Launcher with XP SP2)


Microsoft Visual C++

OLE/COM Object Viewer

- Object Classes
  - Grouped by Component Category
    - Active Scripting Engine
    - Active Scripting Engine with Authoring
    - Active Scripting Engine with Parsing
    - Automation Objects
    - Bitmap Effect
    - Bitmap Transition
    - Controls
    - Controls safely initializable from persistent data
    - Controls that are safely scriptable
    - Desktop Bands
    - Document Objects
    - Embeddable Objects
    - Internet Explorer Browser Band
    - Internet Explorer Browser Communication Band
    - MMControl
    - OLEViewer Interface Viewers
    - Visual InterDev Web Site Wizards
  - OLE 1.0 Objects
  - COM Library Objects
  - All Objects
- Application IDs
- Type Libraries
- Interfaces

Object Classes

System Configuration

System Settings | Default Launch Permissions | Default Access Permissions

[✓] Enable Distributed COM

[ ] Enable Remote Connections (Win95 only)

These settings require you restart your computer after changing them.

OK | Cancel | Apply

Ready

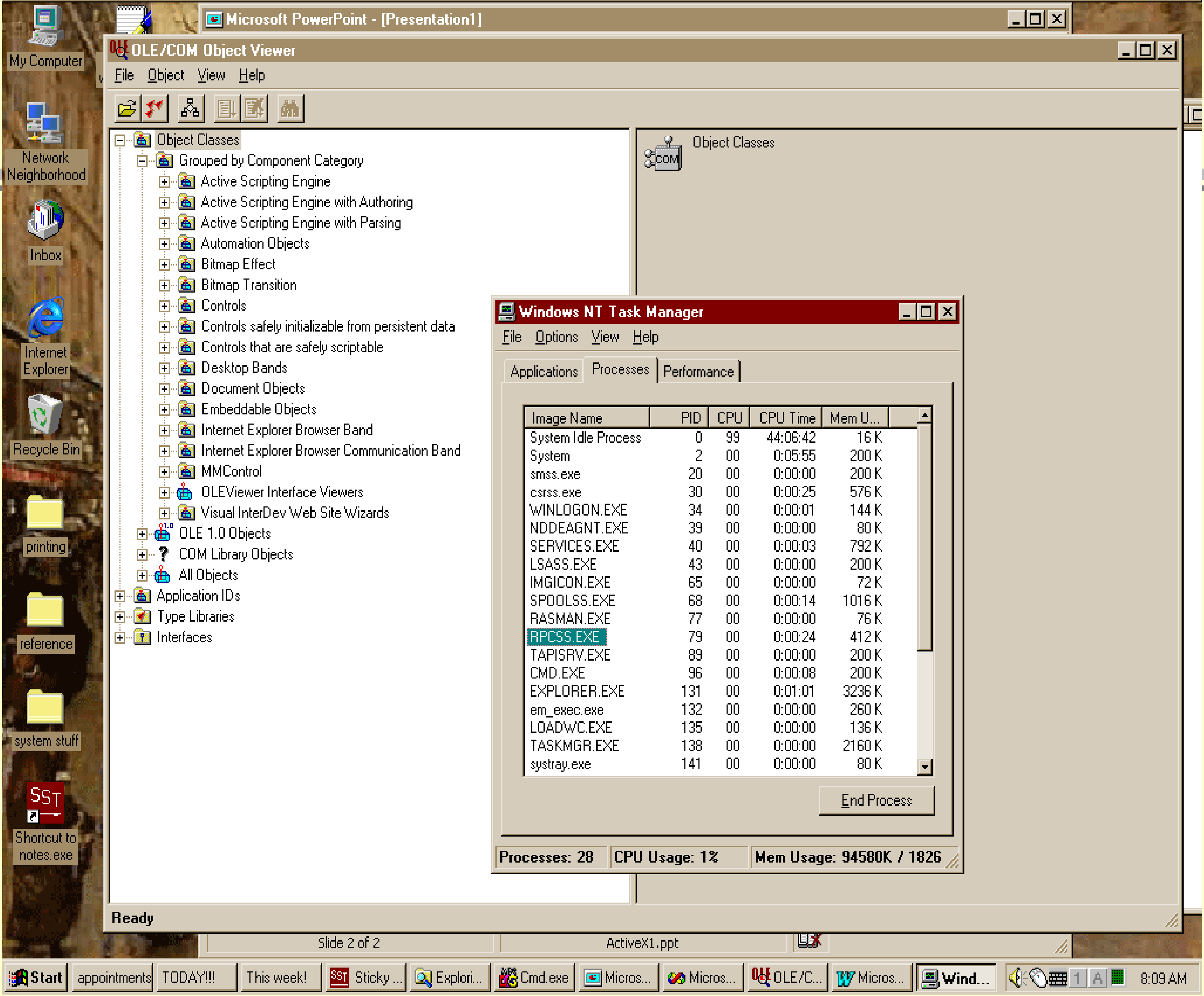

Microsoft PowerPoint - [Presentation1]
OLE/COM Object Viewer
File Object View Help
Object Classes
Grouped by Component Category
Active Scripting Engine
Active Scripting Engine with Authoring
Active Scripting Engine with Parsing
Automation Objects
Bitmap Effect
Bitmap Transition
Controls
Controls safely initializable from persistent data
Controls that are safely scriptable
Desktop Bands
Document Objects
Embeddable Objects
Internet Explorer Browser Band
Internet Explorer Browser Communication Band
MMControl
OLEViewer Interface Viewers
Visual InterDev Web Site Wizards
OLE 1.0 Objects
COM Library Objects
All Objects
Application IDs
Type Libraries
Interfaces
Object Classes
Ready
Windows NT Task Manager
File Options View Help
Applications Processes Performance
End Process
Slide 2 of 2
ActiveX1.ppt
Start appointments TODAY!!! This week! Sticky ... Explori... Cmd.exe Micros... Micros... OLE/C... Micros... Wind...
8:09 AM

Image Name	PID	CPU	CPU Time	Mem U...
System Idle Process	0	99	44:06:42	16 K
System	2	00	0:05:55	200 K
smss.exe	20	00	0:00:00	200 K
csrss.exe	30	00	0:00:25	576 K
WINLOGON.EXE	34	00	0:00:01	144 K
NDDEAGNT.EXE	39	00	0:00:00	80 K
SERVICES.EXE	40	00	0:00:03	792 K
LSASS.EXE	43	00	0:00:00	200 K
IMGICON.EXE	65	00	0:00:00	72 K
SPOOLSS.EXE	68	00	0:00:14	1016 K
RASMAN.EXE	77	00	0:00:00	76 K
RPCSS.EXE	79	00	0:00:24	412 K
TAPISRV.EXE	89	00	0:00:00	200 K
CMD.EXE	96	00	0:00:08	200 K
EXPLORER.EXE	131	00	0:01:01	3236 K
em_exec.exe	132	00	0:00:00	260 K
LOADWC.EXE	135	00	0:00:00	136 K
TASKMGR.EXE	138	00	0:00:00	2160 K
systray.exe	141	00	0:00:00	80 K

Processes: 28 CPU Usage: 1% Mem Usage: 94580K / 1826

Install Components

- Log on to the server machine and:
  - copy all files needed for the server application onto the server machine
    - dlls for components and proxy/stubs
    - type libraries if there are any automation interfaces
    - if you've used type library marshalling there won't be any proxy/stub dll
    - exes for server applications
    - data files needed by the server
  - register dll's for components and proxy/stub using regsvr32
  - register exe's for components using comp /RegServer

- Use ipconfig to determine the IP address of the server machine.

Hooking “Local” clients to Remote Components

- To remotely run components with clients that are not configured for remote operations, we need, on client machine, the registry settings:
  - \HKEY_CLASSES_ROOT\AppID\<component CLSID>\RemoteServerName=“ip address of server node”
  - \HKEY_CLASSES_ROOT\AppID\<component CLSID>\RunAs=“Interactive User”

- We must remove the settings:
  - \HKEY_CLASSES_ROOT\CLSID\<component CLSID>\InProcServer32
  - \HKEY_CLASSES_ROOT\CLSID\<component CLSID>\LocalServer32

Making Clients "Remote" Aware

- To avoid the need for registry settings described on the previous slide, the client must know that it will call a remote server.
- Client uses CoCreateInstanceEx:

```
HRESULT CoCreateInstanceEx(
  REFCLSID rclsid,                // component CLSID
  IUnknown *punkOuter,            // must be NULL
  DWORD *dwClsCtx,                // usually CLSCTX_SERVER
  COSERVERINFO *pServerInfo,      // see next slide
  ULONG cmq,                      // number of MULTI_QIs
  MULTI_QI rgmqResults            // array of MULTI_QIs
);
```

- Returns `S_OK, E_INVALIDARG, E_NOINTERFACE`, or `CO_S_NOTALLINTERFACES`

COSERVERINFO Structure

- This structure identifies the remote machine and may also define security settings

```
typedef struct _COSERVERINFO {
  DWORD dwReserved1;          // set to zero
  LPWSTR pwszName;            // IP address or DNS or UNC name
  COAUTHINFO __RPC_FAR *pSuthInfo;  // see below
  DWORD dwReserved2;          // set to zero
} COSERVERINFO;
```

- If `pSuthInfo` is set to NULL the NT Domain server is used to authenticate a user.

COAUTHINFO Structure

- This structure is used to activate a server based on specified security settings rather than taking the InterActive User settings.

```
 typedef struct _COAUTHINFO {
   DWORD dwAuthnSvc;
   DWORD dwAuthzSvc;
   LPWSTR pwszServerPrincName;
   DWORD dwAuthnLevel;
   DWORD dwImpersonationLevel;
   COAUTHIDENTITY __RPC_FAR *pAuthIdentityData;
   DWORD dwCapabilities;
} COAUTHINFO;
```

MULTI_QI Structure

- We can pass to CoCreateInstanceEx an array of MULTI_QI structures to get back pointers to several interfaces with a single round trip call.

```
typedef struct _MULTI_QI {
  const IID *pIID;  // a desired interface CLSID
  IUnknown *pItf;   // contains interface pointer on return
  HRESULT hr;       // returns S_OK if successful
} MULTI_QI;
```

Downloading Proxy/Stub dlls

- If your client needs to download a proxy/stub dll to communicate with a remote server, it may make its interfaces available without making registry entries locally by using CoRegisterPSClsid:

```
WINOLEAPI CoRegisterPSClsid(
  REFIID riid,       // interface used by client
  REFCLSID rclsid    // proxy/stub CLSID
 );
```

- This will not make permanent changes in the local registry. The interface will be available for the lifetime of the process or until `CoRegisterPSClsid` is called again.

Automated Download and Install

- An activeX control can be embedded in web page using object tag:

```
<object
  ID = “MyControl”
  CLSSID=“clsid::xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx”
  CODEBASE=“http://xxxx/yyyy/…/control.dll#version=1,0,0,121”
  TYPE=”application/x-oleobject”
  WIDTH=150
  HEIGHT=60
  VSPACE=0
  ALIGN=left
>
```

- The control will be downloaded if it hasn’t been installed locally or if the local version number is older than that in the object tag.

CoGetClassObjectFromURL

- To process this object tag the browser (IE3.0 or later) will use:

```
STDAPI CoGetClassObjectFromURL(
  REFCLSID   rclsid,                    // CLSID of object
  LPCWSTR    szCodeURL,                 // URL path w file name
  DWORD      dwFileVersionMS,           // major version number
  DWORD      dwFileVersionLS,           // minor version number
  LPCWSTR    szContentTYPE,             // content type string
  LPBINDCTX pBindCtx,       // client generated bind context
  DWORD      dwClsContext,              // CLSCTX_INPROC_SERVER
  LPVOID     pvReserved,                // NULL
  REFIID     riid                       // UUID of IClassFactory
  VOID **    ppv                        // returned interface
);
```