COM, ActiveX, OLE Automation, TLB

스크래핑센터

최진영

COM에 대해

- COM(Component Object Model)
  - Re Use (재사용) 즉 OPP(객체 지향 프로그래밍)을 하기위해서 COM을 한다.
  - 모듈화 구현
- OPP의 한계 극복
  - C++에서 클래스 작성 해봐야 Delphi 같은 파스칼을 쓰는 환경에선 소용없음
    -> COM의 Binanry 수준에서 객체 분류,모듈 분류
  - Compile 다시해야한다. OPP프로그래밍은 매커니즘은 단지 소스 이므로 실제 적용 시 다시 컴파일 해야함
    -> COM으로 작성된 모듈이면 COM모듈만 바뀌면 그 COM을 사용하는 응용프로그램도 적용 받음
- COM / DLL 비교
  - 위치에 영향을 받지 않는다. 로컬 pc에 설치된 *.dll파일의 위치 뿐만 아니라 리모트pc(서버)에 설치된 모듈까지 구분이 없음
  - 자체적으로 버전정보를 제공한다. 구식 dll은 자신이 포함하고 있는 함수를 노출해야만 한다(sum.dll->sum2.dll->sum3.dll 이런식으로 버전 업 그레이드를 하여도 같은 기능을 하는 dll을 계속써야함. 하지만 COM은 자체적으로 버전정보를 지원하여 이러한 문제를 해결

COM(Component Object Model)

- 자신의 고유한 기능을 제공하는 단위 어플리케이션(=컴포넌트)의 통합 및 커뮤니케이션방법에 대한 표준을 정의한 사양
- 핵심적인 MS의 기반 기술 역할 제공
  - OLE,ActiveX,ADO,OLE DB,Active Directory 모두 COM을 기반으로 작성
- Component규약을 따르는 MS 사가 주체가 되어 만든 표준
  - Component규약
    - 컴파일러의 종류에 관계없이 이 Component 규약에 따라 제작한 Component는 어떤 컴파일러에서도 그 파일을 사용할 수 있도록 하는 규약 ex)COM,CORBA,EJP,.NET
- COM의 특징
  - 언어 독립성 - Interface 라는 방식으로 COM에서 지원하는 어떠한 언어로 작성되어도 서버 프로그램 작성 가능
  - 위치 투명성 - 컴포넌트가 물리적 위치에 관계없이 다른 컴포넌트 혹은 컴포넌트 클라이언트에 의해 사용될 수 있다는 것
  - Binary Standard - 개발자는 exe,ocx,dll 등과 같은 binary만 있으면 컴포넌트를 사용 가능

COM의 Interface

- COM 객체가 자신의 기능을 노출시키는 기본적인 방법
- COM 객체와 클라이언트 사이의 계약
- 각 COM 객체는 반드시 IUnkown 인터페이스
- COM 객체 고유의 기능을 노출하는 하나 이상의 인터페이스를 제공

- COM 인터페이스 정의
  - IDL로 COM인터페이스 정의
  - IDL(Interface Definition Language)
    - 인터페이스를 정의하는 표준 개발 도구
    - MIDL 컴파일 제공
    - 언어 독립성 제공 (형식 라이브러리)
    - 위치투명서 제공(프록시/스텁코드)

COM의 Interface

- IUnknown 인터페이스
  - 모든 COM 인터페이스는 Iunknown을 상속
  - 모든 COM 객체가 갖추어야 할 기본적인 서비스 제공
  - QueryInterface :
    - 특정 인터페이스에 관하여 그 개체에 질의를 던지는 역할
    - COM개체가 제공하는 인터페이스 구하는 유일한 방법
    - QueryInterface를 통하여 COM객체가 지원하는 다른 인터페이스를 요청
  - AddRef/Release : : 개체의 생성,소멸을 관리
    - 인터페이스를 리턴하기전 AddRef한다.
    - 인터페이스의 사용이 끝나면 Release 한다.
    - 인터페이스 포인터를 다른 인터페이스 포인터에 대입할 때 AddRef한다.
- IDispatch 인터페이스
  - 후기의 바인딩을 사용하여 개체에 접근하고자 할 때 사용
  - 자동화(Automation)을 지원하기 위해 IUnknown 인터페이스로부터 파생된 인터페이스
- Custom Interface
  - Script언어에서 지원 불가능,컴파일 언어 지원가능 ex) Visual Basic Script
- Automation Interface
  - Script 언어 지원, 컴파일 언어 불가능
- Dual Interface
  - 모두 지원 가능

COM(Component Object Model) 접근 방식

- COM 컴포넌트 : 자신의 고유한 서비스 제공
  - COM 서버
    - In-Process-Sever(대부분 사용)
      - DLL파일로 구현
    - Out-Process-Sever
      - EXE파일로 구현
      - 로컬 서버
      - 리모트 서버
  - COM 객체 : COM 인터페이스를 구현한 클래스의 인스턴스
- COM 클라이언트 : COM 컴포넌트의 서비스를 사용

- In-Process-Sever & Out-Process-Sever
  - In-Process-Sever의 경우 같은 PC에서 Sever와 Client를 제작할 경우 Client의 문제에 의해 Server에서도 그 문제가 파급될 수 있음.
  - Out-Process-Server의 경우 Client의 문제가 Server로 파급되지 않는다.
  - 만약 Client 프로그램이 다른 pc에 위치할 때 In-Process Server의 경우 dllHost.exe라는 대리 프로세스에 의해 지원됨 (이 경우 Client의 문제가 Server로 파급되지 않는다.)

COM 컴포넌트 사용

- 레지스트리에 반드시 등록

서버유형	In-process-server	Out-of-process-server
레지 등록	Regsvr32 DLL 파일명	Exe파일명 /RegServer
레지 해제	Rgsvr32 /u DLL파일명	Exe파일명 /UnregServer

- COM client 프로그램 제작 순서

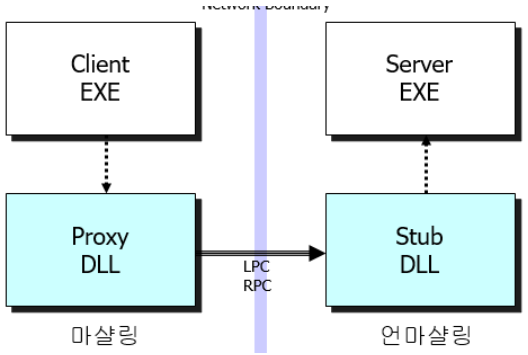
COM 라이브러리 초기화	CoInitialize 함수 사용
CLSID 얻음	CLSID From ProgID 함수 사용
Instance 생성	CoCreateInstance함수 사용(서버를 실행하고, 그 서버의 Interface 포인터 얻음)
COM object 사용	Method 및 Property 사용
COM 라이브러리 삭제	CoUninitialize();

GUID의 다른이름

- UUID(Universal Unique Indentifier)
- GUID(Globally Unique Indentifier)
- CLSID(Class Indentifier)
- IID(Interface Indentifier) GUIDGEN
- LIBID(Type Library Indentifier)

COM(Component Object Model) 용어 정리

- Marshaling
  - 클라이언트와 컴포넌트에 공유되는 모든 데이터는 포장(packaged)되어서 어떤 방법으로 전송 되어야함. 이 프로세스를 마셰링이라고 하며 '순서로 정돈한다'라는 의미이다.
- Dispinterface
  - IDispatch 인터페이스에만 완전히 기반한 인터페이서를 dispinterface라고 한다.
  - IDispatch = Dispinterface + v-table
- 프록시/스텁



COM 객체 구현

- 인터페이스 포함
  - COM 객체 클래스 안에 인터페이스 구현 클래스를 포함
  - 구문은 복잡하지만 디버깅이 쉽다.
  - MFC에서 사용
- 인터페이스 상속
  - COM객체 클래스를 인터페이스에서 상속
  - 간단하다.
  - ATL에서 사용
- COM 객체 클래스 정의
  - 다중 인터페이스 구현을 위해 다중 상속 이용
  - COM 객체 구현 클래스에서는 모든 인터페이스 메서드를 제정의
  - IUnknown

ActiveX

- COM 기술에 Automation을 지원하고 GUI를 추가로 지원한 것이 ActiveX이다. (ActiveX=OLE+COM)
- Internet Explorer에 결합하여 웹 브라우저에서 실행 파일을 실행
- 작동 시 윈도우 또는 윈도우 에뮬레이터를 써야함
- 웹 표준을 지키지 않아 퇴출위기, MS가 사용 자제를 '권고'

- ActiveX를 작성하는 방식
  - MFC
    - WinInet 클래스를 사용하면 쉽게 HTTP,FTP 프로토콜을 사용하여 TCP/IP 및 WinSock 프로토콜을 추상화 하여 데이터 전송
  - ATL
    - MFC를 이용하는 방식보다 다소 어렵기는 하나 작고 빠른 코드를 생산할 수 있는 장점이 있어 더 많이 사용되고 있다.

ActiveX 기능

- 컴퓨터 내부에 파일을 생성
- 삭제 및 존재 여부 확인
- 사용자가 관리자 권한을 허락하면 윈도우 폴더 내부에 까지 파일 생성
- 레지스트리도 엑티브엑스 오비젝트를 마음대로 수정 가능
- OLE2.0의 복잡함 해소 및 MFC의 한계 극복
- Internet Explore 3.0과 결합하여 HTML 안에 엑티브 X 컨트롤을 관리
- 브라우저가 Object 태그를 통하여 엑티브X 컨트롤을 발견하면 사용자의 간섭 없이도 엑티브X 컨트롤을 자동으로 내려 받아 설치

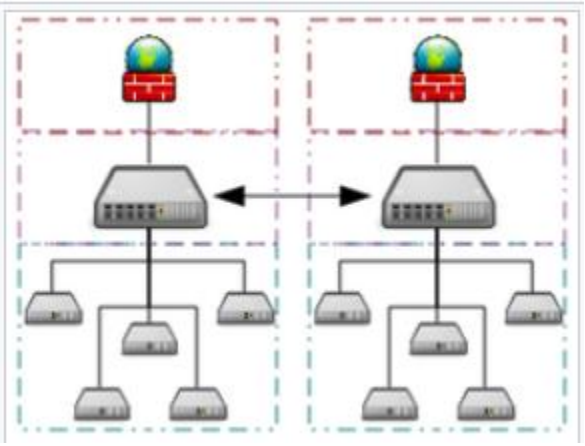
ActiveX 문제점

- 사용자의 간섭 없이도 자동으로 설치되어서 보안의 허점
  - 윈도우 10에 와서야 MS는 Edge를 만들어 ActiveX의 위의 문제점 해결
- Internet Explore8에서만 거의 정상적으로 작동하고 윈도우가 아닌 다른 운영체제에서는 실행이 불가능하다. 특히 Mac OS용 IE에서는 엑티브X를 실행시킬 수 없음.
- Google의 Chrome에서 지원하다가 45(버전)부터는 지원을 공식적으로 지원 종료

OLE(Object Linking Embedding) : 개체연결 및 삽입

- 다양한 개체 자료들을 독립적으로 사용하는 것이 아니라 서로 유기적으로 엮어 하나의 응용프로그램에서 사용할 수 있는 기능
  - Ex)한글문서에 도표,수식을 포함시키는 것
- 대상이 되는 파일이나 프로그램을 연결하여 끼워 넣는 방법을 가리킨다. Ex)문서,동영상,소리,만화 등을 담는 바탕화면 같은 것
- OCX(OLE Control Extexsions) : OLE를 확장 한 것. MS 윈도우95 이상에 대한 OCX표준안이 ActiveX이다.

OLE Automation



IPC

- MS가 개발한 프로세스 간 통신(IPC) 매커니즘
  - IPC : 프로세스들 사이에 서로 데이터를 주고받는 행위,방법,경로를 뜻한다.
- COM의 하위 집합
- 모든 자동화 오브젝트들은 IDispatch 인터페이스의 구현
- 동적 데이터 교환(DDE)를 대체
- 반드시 MS OLE를 사용할 필요는 없으나 자동화 오브젝트들 가운데 일부는 OLE 환경에서 사용 가능.

OLE Automation 지원 언어

- ABAP
- C
- C++
- C#
- 비주얼 베이직
- 닷넷

TLB

- 사용자 인터페이스 데이터는 TLB 파일에 저장되고 필요할 때 OLE 지원 프로그램에 의해 참조
- ActiveX 서버 또는 OLE 서버가 개발된 경우 TLB파일이 생성

- COM의 가장 큰 장점인 언어 독립적을 위해 TLB 파일이 필요하다.
  - Type Library : .dll 파일을 만들었다면 이 안의 메서드를 이용하기 위해서는 .h파일이 있어야 함. .h 파일은 C# 이나 VB에서 읽지 못함 따라서 tlb 파일이 필요하다.