COM e tecnologie Web

Enrico Lodolo
e.lodolo@bo.nettuno.it

Tecnologie Microsoft per Inter/Intranet

- **COM rappresenta la base di tutte le tecnologie Microsoft in ambito Intranet ed Internet (Web)**
- **Possiamo in particolare distinguere 3 grandi aree:**
  - **Active scripting**
  - **Strumenti *client-side* (Internet explorer)**
  - **Strumenti s*erver-side* (IIS e ASP)**

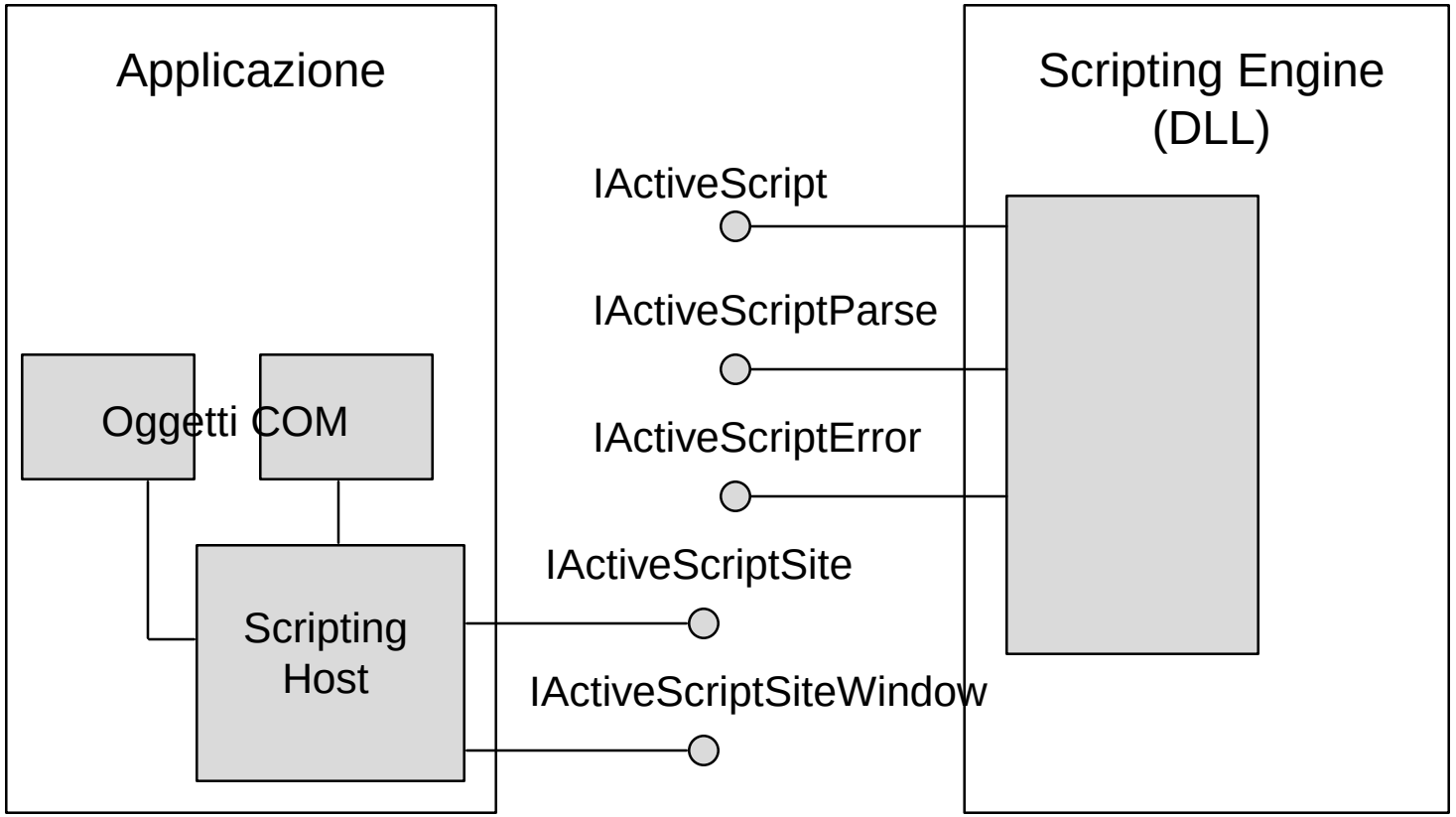
Automation e scripting

- **La tecnologia Automation costituisce uno strumento ideale per realizzare sistemi di scripting estensibili**
- **Infatti praticamente tutti i linguaggi di script più diffusi sono basati su un modello ad oggetti (Javascript, VBScript/VBA , Python, Perl ecc.), quindi si adattano bene al modello COM**
- **Inoltre un interprete riesce ad accedere facilmente ad oggetti che implementano l'interfaccia IDispatch**
- **Questi ultimi infatti possiedono le due caratteristiche di base per essere comandati da uno script-engine:**
  - **Introspezione** (type libraries): l'interprete è in grado di ricavare tutte le informazioni necessarie riguardo ai metodi esposti e ai parametri necessari
  - **Very-late binding** (dispatching): l'interprete è in grado di invocare dinamicamente i vari metodi utilizzando IDispatch.Invoke

Active scripting: concetti di base

- La tecnologia Active Scripting rappresenta un modo standard per consentire ad un'applicazione di utilizzare un motore di scripting per automatizzare alcune funzioni
- In pratica si tratta di una collezione di interfacce COM (object model) che vengono in parte implementate dall'applicazione e in parte dallo script engine
- I vari script engine che supportano questo modello sono intercambiabili fra di loro
- Oltre ai due motori forniti dalla Microsoft (VBScript e JScript) ne sono già disponibili altri, prodotti da terze parti: per esempio PythonWin è un implementazione freeware del motore di scripting per Python
- Quindi esiste una netta indipendenza fra la capacità di un applicazione di essere "automatizzata" e il motore di scripting che utilizza questa capacità:l'unico legame è l'aderenza al modello Active Scripting

Active scripting: schema generale



Active scripting: funzionamento

- **Lo script-engine è un oggetto COM e come tale viene registrato nel sistema**
- **L'applicazione può attivarlo con CoCreateObject e ottenere (con QueryInterface) le sue due interfacce principali**
- **IActiveScript comprende metodi per**
  - attivare e disattivare l'engine
  - passare al motore l'interfaccia IActiveScriptSite implementata dall'applicazione
  - Indicare i nomi degli oggetti COM Automation esposti dall'applicazione
  - Controllare il thread che esegue uno script
- **IActiveScriptParse consente invece di attivare uno script passato sotto forma di stream di testo**
- **Tutte le subroutine contenute nello script vengono esposte come metodi di una *dispinterface* e quindi possono essere invocati**
- **IActiveScriptError viene invece passata all'applicazione quando si verifica un errore e consente di ricevere informazioni sul tipo di errore, la sua posizione ecc.**

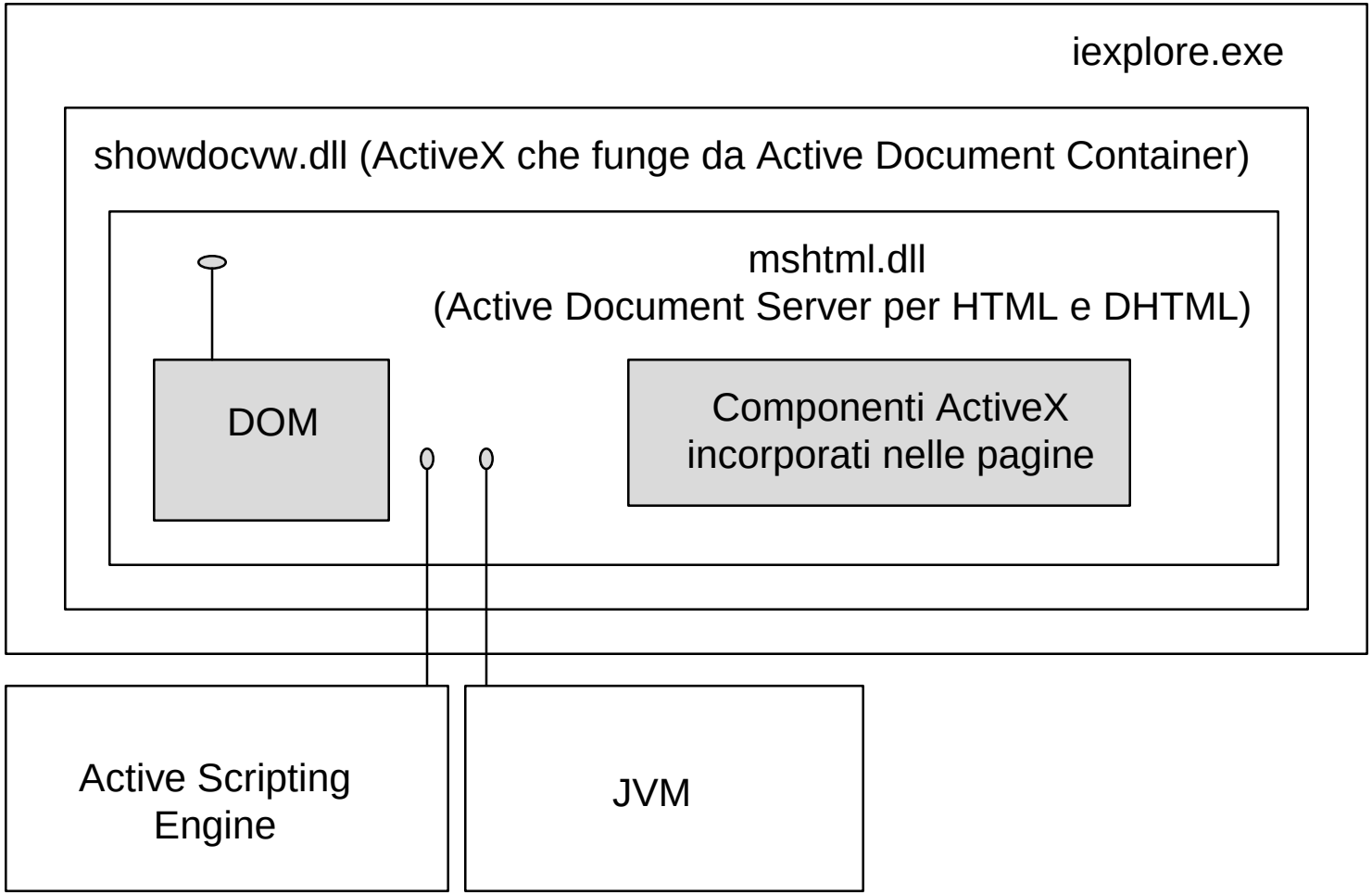
Active Scripting: funzionamento

- L’applicazione espone un’interfaccia IActiveScriptSite che funziona da “callback”
- L’applicazione deve creare inoltre una classe COM Automation per ogni oggetto interno che vuole esporre
- I nomi di queste classi vengono registrate nell’engine attraverso IActiveScript.AddNamedItem
- Quando l’engine incontra nello script un’istruzione del tipo CreateObject(“ClassName”) chiama IActiveScriptSite.GetItemInfo per istanziare nell’applicazione l’oggetto opportuno
- GetItemInfo restituisce l’interfaccia IUnknown dell’oggetto in questione e quindi - tramite QueryInterface - l’interfaccia IDispatch
- Ogni volta che l’engine incontra un’istruzione del tipo ClassName.DoSomething utilizza IDispatch per invocare il metodo
- La seconda interfaccia esposta dall’appliczione verso l’engine - IActiveScriptSiteWindow - permette all’engine di ricavare l’handle della finestra da usare come parent per le finestre eventualmente aperte dallo script

Client-side: Internet Explorer e DHTML

- **La strategia Microsoft sul lato client delle applicazioni Web poggia su due pilastri: il browser Internet Explorer e DHTML**
- **Internet Explorer ha una struttura fortemente componentizzata e rappresenta l'esempio più completo ed interessante di applicazione delle tecnologie basate su COM**
- **DHTML è una versione dinamica di HTML, proposta dalla Microsoft, che è stata recentemente accettata come base per lo standard DHTML dal W3C consortium**
- **DHTML si basa su un modello ad oggetti denominato DOM (Document Object Model) che può essere manipolato mediante interfacce COM**

Struttura di Internet Explorer



Struttura di IE

- **L'eseguibile IEXPLORE.EXE è semplicemente un "telaio" che funziona da ActiveX container e gestisce menu e toolbar**
- **Il vero motore di Internet Explorer è un componente ActiveX contenuto nella DLL SHOWDOCVW.DLL, che espone un'interfaccia per la naigazione e funziona da visualizzatore universale di documenti**
- **Infatti questo componente è in grado di ospitare, e quindi di visualizzare, qualunque tipo di documento per cui sia disponibile un server Active Documents**
- **Grazie a SHOWDOCVW.DLL è possibile creare applicazioni che incorporano funzionalità di navigazione Web**
- **Quindi SHOWDOCVW.DLL è nel contempo un componente ActiveX e un container Active Documents**
- **MSHTML.DLL non è altro che uno di questi server, specializzato nella gestione di documenti HTML e DHTML**
- **MSHTML espone un'interfaccia che consente di accedere al Document Object Model contenuto al suo interno e quindi di pilotare le pagine dinamiche di DHTML**

IE: incorporamento di ActiveX

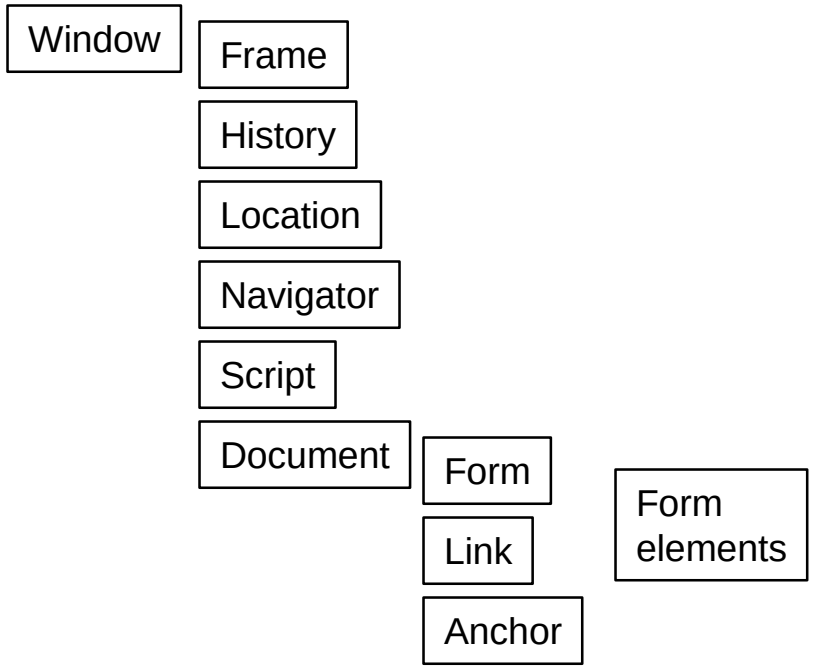
- **Inoltre Internet Explorer consente di incorporare componenti ActiveX all'interno delle pagine HTML mediante una sintassi di questo tipo:**

```
<OBJECT ID="ClassName" WIDTH=300 HEIGHT=32
  CLASSID="CLSID:D7053240-CE69-11CD-A777-00D01143C57">
  <PARAM NAME="Size" VALUE="2540;846"
  <PARAM NAME="FontCharSet" VALUE="0"
  ...
</OBJECT>
```

- **Questi componenti si comportano in pratica in modo molto simile alle applet Java**
- **Trattandosi di file binari non sono però interpiattaforma e pongono problemi di sicurezza**
- **Dopo una grossa enfasi iniziale sono stati un po' abbandonati a favore di DHTML**

IE: scripting

- **Le capacità di scripting di Internet Explorer sono relizzate mediante la tecnologia Active Scripting descritta in precedenza**
- **I vari componenti dell'explorer espongono una serie di oggetti Automation che costituiscono l'object model su cui JScript (la versione Microsoft di JavaScript) e VBScript possono operare:**

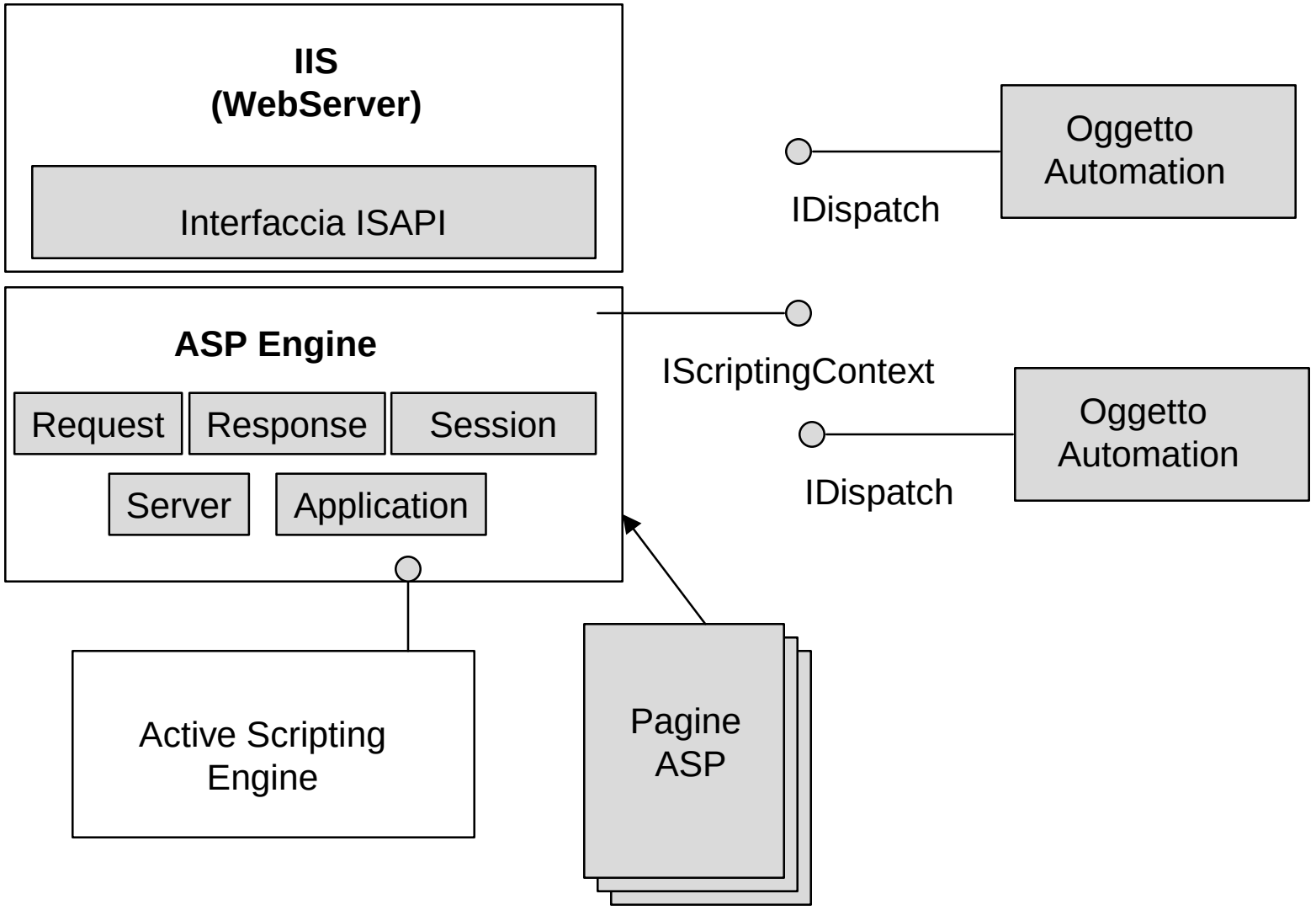


- **Oltre agli oggetti tradizionali legati ad HTML gli script possono operare anche sugli oggetti DOM per gestire le pagine dinamiche**

Server-side: IIS, ISAPI e ASP

- **Il perno della strategia Microsoft sul lato server è il server Web di NT, chiamato Internet Information Server (IIS)**
- **IIS prevede un'API denominata ISAPI che consente di estenderne le funzionalità mediante DLL**
- **Le DLL ISAPI sono comparabili alle applicazioni CGI ma sono più efficienti perchè non richiedono l'attivazione di un nuovo processo ad ogni richiesta di esecuzione**
- **Però il fatto che le estensioni ISAPI risiedono nello stesso spazio di processo del Web Server è un elemento di fragilità: un errore all'interno della DLL è in grado di mandare in crash il server**
- **ASP (Active Server Pages) rappresenta la tecnologia SSS (Server Side Scripting) proposta dalla Microsoft**
- **Le active server pages sono pagine che contengono sia parti HTML che parti di script (VBScript, JScript o altro). Gli script vengono eseguiti dal motore ASP sul lato server e al browser arrivano pagine HTML "pulite"**
- **Il motore ASP è realizzato sotto forma di estensione ISAPI (ASP.DLL)**

ASP: schema generale



ASP: principi di funzionamento

- **Quando il Web server riceve un URL di questo tipo:**

  www.myserver.com/apage.asp?par1&par2&par3

  sa che la pagina apage.asp deve essere passata al motore ASP per essere processata
- **In una pagina ASP abbiamo elementi di HTML inframmezzati da script.**
- **Le righe di script sono identificate dai tag <% e %>**
- **Gli elementi HTML vengono passati direttamente al web server mentre le parti di script vengono interpretate utilizzando un motore di scipting di tipo Active Scripting (VBScript, JeScript, Python o altro)**
- **Se uno script inizia con = il risultato della valutazione viene inserito all'interno della pagina HTML al posto dello script stesso**
- **Per esempio la sequenza**

  <H3> Ultima visita <%= Request.Cookies("LastVisit")> </H3>

  viene tradotta e passata al web server così:

  <H3>Ultima visita 10/5/1999<H3>

ASP: modello ad oggetti/1

- **Similmente a quanto avviene sul lato client, gli script si basano su un modello ad oggetti**
- **ASP mette a disposizione 5 oggetti fondamentali: Request, Response, Server, Session e Application**
- **Questi oggetti sono implementati in ASP.DLL sotto forma di oggetti COM Automation e passati al motore Active Scripting con le modalità descritte in precedenza**
- **Request mette a disposizione sotto forma di metodi e proprietà tutti gli elementi passati dal browser al web server nella richiesta HTTP:**
  - URL (in particolare i parametri dopo il ?)
  - Sequenze di POST o GET nel caso di form
  - Cookies
  - ...
- **ASP fa il parsing della richiesta e permette di accedere comodamente ai vari elementi. Per esempio Request.QueryString("Name") restituisce Mario se l'URL era:**

  www.myserver.com/mypage.asp?Name=Mario&ID=35

ASP: modello ad oggetti/2

- **L'oggetto Response rappresenta la pagina HTML che viene passata al Web server e quindi restituita al browser**
- **Permette allo script di intervenire nella costruzione di questa pagina e di accedere agli elementi HTTP che vengono spediti assieme ad essa:**
  - Header
  - ContentType: text/html, image/gif, image/jpeg ecc.
  - Cookies
  - Status: "200 Ok" o messaggio di errore
  - Expires: validità della pagina in minuti (evita ricaricamenti inutili)
- **La pagina viene normalmente costruita sequenzialmente ma Response espone la property Buffer che permette di gestire una bufferizzazione e quindi di operare in modo non sequenziale**
- **Il metodo Response.Write("xxxx") permette di inserire righe HTML all'interno della pagina in costruzione, convertendo i caratteri speciali nelle sequenze opportune (p.es > viene convertito in >)**
- **Con Response.WriteBinary() possiamo invece scrivere sezioni binarie (p.es immagini GIF)**

ASP: modello ad oggetti/3

- L’oggetto Server mette a disposizione una serie di funzioni di utilità: per esempio URLDecode e HTMLDecode
- Il metodo più importante è Server.CreateObject(ProgID) che consente di agganciare oggetti COM automation allo script
- L’oggetto Session permette di ovviare ai problemi connessi al fatto che HTTP è un protocollo “stateless” e quindi non mantiene alcun stato tra una transazione e l’altra
- Session usa i cookies dietro le quinte per fornire alle applicazioni ASP un meccanismo di persistenza fra una transazione e l’altra
- Oltre a ciò Session fornisce una serie di metodi per riconoscere l’utente e per gestire i timeout
- L’oggetto Application viene utilizzato per memorizzare informazioni globali, condivise fra tutte le sessioni e gli utenti
- In ASP un’applicazione è l’insieme di tutte le pagine conenute in una directory e nelle sue sottodirectory

ASP e oggetti COM

- Come abbiamo il motore ASP consente di creare e utilizzare oggetti COM Automation negli script
- Gli oggetti vengono istanziati con Server.CreateObject(ProgID) ed è poi possibile invocare tutti i metodi esposti via IDispatch
- Gli oggetti utilizzati possono essere normali oggetti Automation oppure essere “ASP aware”
- Questi ultimi sono in grado di accedere all’object model esposto dal motore ASP (Request, Response, Server, Session e Application)
- Quando viene invocato Server.CreateObject ASP verifica (leggendo nella type library) se l’oggetto creato espone il metodo OnStartPage
- In caso positivo chiama StartPage passando come parametro un interfaccia specifice di ASP: IScriptingContext
- IScriptingContext espone 4 proprietà: Request, Response, Server, Session e Application, ovvero l’object model ASP.
- A questo punto l’oggetto COM è collegato con l’applicazione ASP ed è in grado di accedere e tutte le sue funzionalità

Altre informazioni

- **Il linguaggio di default per le pagine ASP e VBScript ma è possibile definire un altro linguaggio attraverso la direttiva @LANGUAGE:**
- **Per esempio inserendo il tag:**

  <% @ LANGUAGE “JScript” %>

  la pagina ASP in cui è contenuto utilizza JScript
- **E’ anche possibile mescolare sezioni di script in diversi linguaggi utilizzando una versone estesa del tag <SCRIPT>:**

  <SCRIPT LANGUAGE = “NomeLinguaggio” RUNAT=“Server”>

  ...

  </SCRIPT>
- **Con ASP è possibile accedere ai database attraverso ADO (Active Data Object), una tecnologia COM che permette l’accesso ai database via ODBC o OLE DB**
- **Esiste una implementazione di ASP realizzata da Chili!Soft (Chili!ASP) che gira su altri web server: Netscape Fasttrack, Lotus Domino, IBM CSS ecc. e in diversi sistemi operativi (NT, Solaris...)**