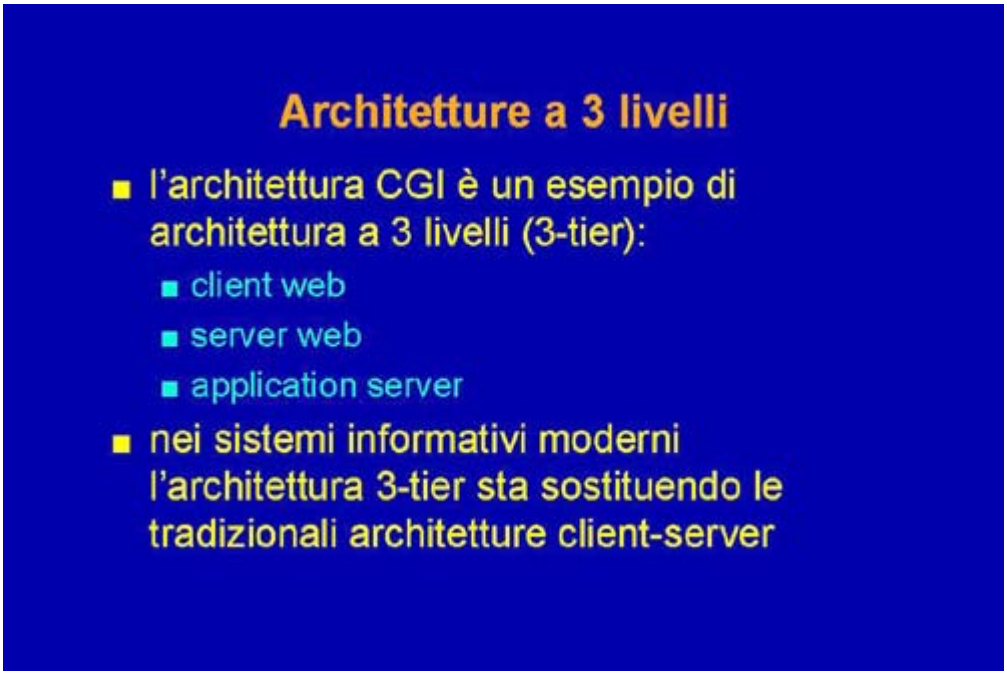
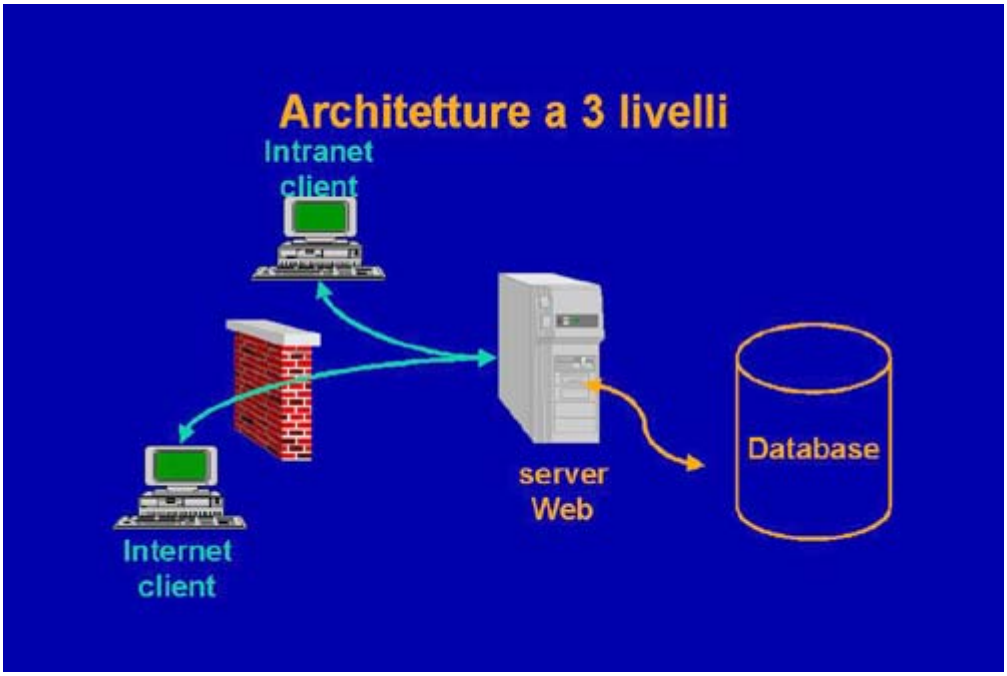
Architetture Web a tre livelli: CGI, SSI, ISAPI e codice mobile
Architetture a 3 livelli (1)



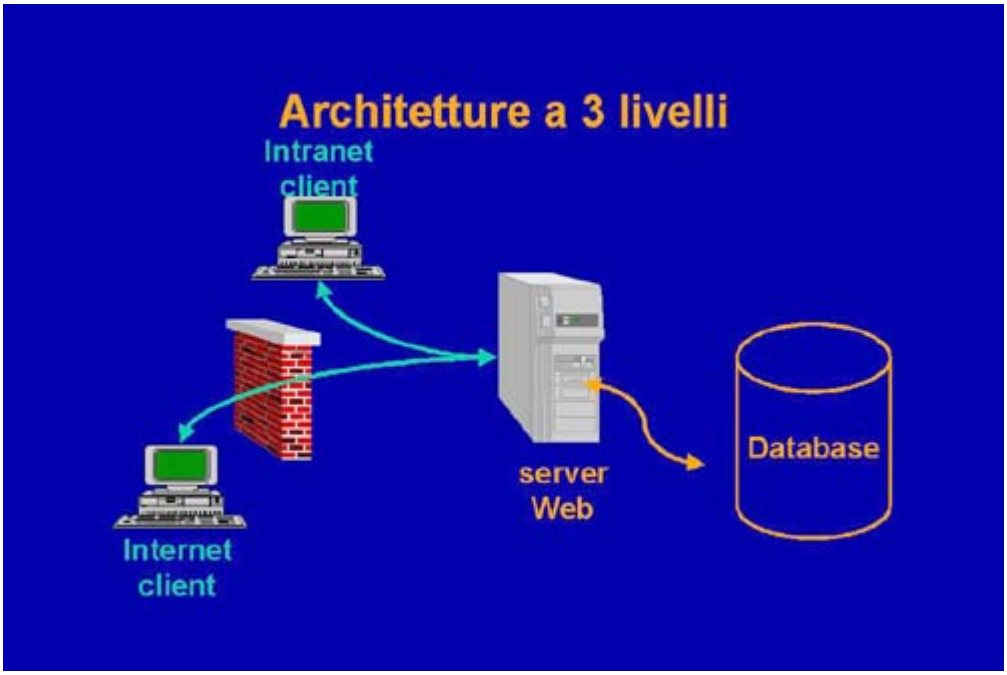
Nel corso della lezione precedente abbiamo analizzato le caratteristiche dell'architettura CGI. L'architettura CGI è un esempio di architettura Web a tre livelli, ma non è l'unico. Nel corso di questa lezione andremo a vedere quali sono le altre architetture a tre livelli e vedremo quali sono le caratteristiche, i vantaggi e gli svantaggi rispetto all'architettura CGI. Si parla di architetture a tre livelli quando abbiamo a che fare con architetture distribuite dove viene utilizzato un client Web, un server Web ed un application server. Questo tipo di architettura sta sostituendo, in quelli che sono i sistemi informativi moderni, la tradizionale architettura client-server, dove l'accesso ad un application server era fatto attraverso un client dedicato, costruito e programmato appositamente per accedere a quel application server.

Architetture a 3 livelli (2)



Un architettura di questo tipo, un architettura di tipo client-server, richiede quindi la costruzione di uno specifico client per ciascuna specifica applicazione. Questo può essere modificato andando ad utilizzare una architettura a tre livelli nella quale, appunto, viene utilizzato, non più un client specifico per accedere all'application server, ma un tradizionale client Web. L'architettura generale è quella indicata in questa figura. Come vedete il client è un normale client Web che accede al data base applicativo non direttamente, ma attraverso un server Web intermedio. Questo fa sì che possano essere utilizzati client Web standard e che non debba essere sviluppato un client specifico per ogni applicazione che intende accedere a questo data base. Questa architettura, quindi, prevede tre livelli: il primo è il client Web, il secondo livello è il server Web e il terzo livello è il data base applicativo.

Architetture a 3 livelli (3)



Uno dei vantaggi di questa architettura è che è un'architettura di tipo aperto: ovvero non ci si lega in particolare ad un tipo di client, ad un tipo di server e ad un tipo di data base applicativo, ma abbiamo la flessibilità necessaria per andare a sostituire un client Web con un diverso client Web, o per andare a modificare il server Web che stiamo utilizzando. Un altro aspetto fondamentale di questo tipo di architettura è la sua scalabilità: la sua apertura verso ambienti Internet. Se sviluppiamo, infatti, un sistema informativo all'interno di una intranet, il cui limite è rappresentato da questo muro che divide la intranet dalla rete Internet esterna, possiamo facilmente estendere i servizi che distribuiamo all'interno della nostra intranet, semplicemente evolvendo quelli che sono i servizi distribuiti dal nostro server Web. Per arrivare su Internet non avremo bisogno di nulla in più di quanto non utilizziamo per arrivare ai nostri utenti intranet, quindi, anche chi utilizzerà da Internet il proprio navigatore Web, potrà accedere a quelli che sono i servizi del nostro sistema informativo. È fondamentale in questo caso avere un meccanismo di sicurezza che consenta di distinguere, da un punto di vista logico, quello che è il traffico Internet da quello che è il traffico intranet ed era rappresentato, nella slide precedente, da quel muro che rappresenta un sistema firewall che protegge la rete intranet dalle richieste provenienti dalla rete Internet.

Architetture a 3 livelli (4)

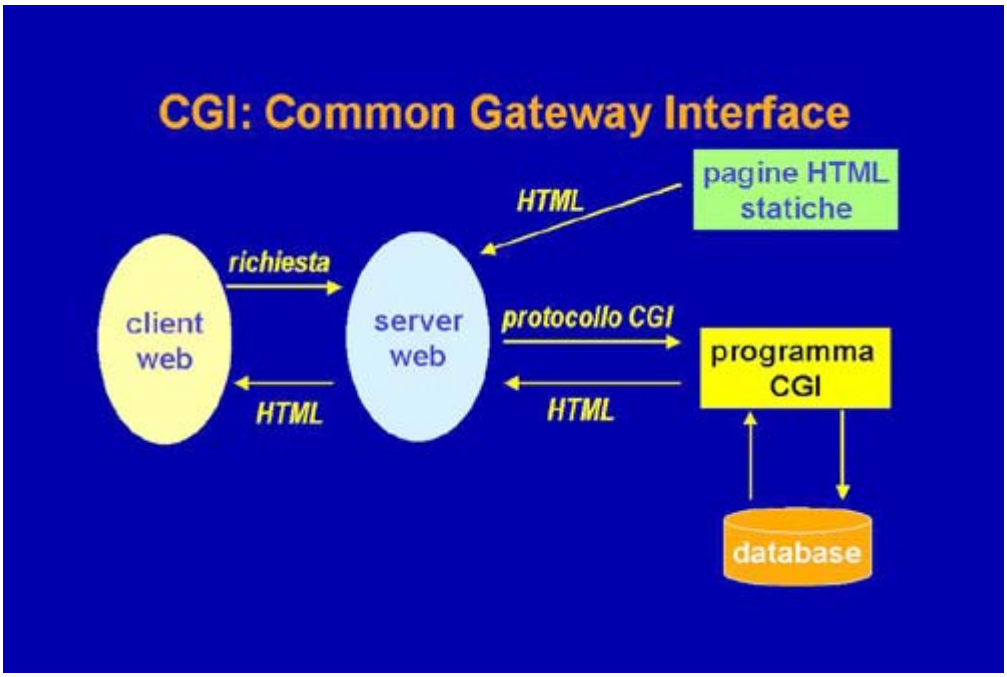
Quali sono i vantaggi principali di una architettura a tre livelli? Li vediamo indicati in questa slide. Abbiamo, innanzitutto, indipendenza dal client, nel senso che il nostro utente potrà utilizzare il client Web di sua preferenza, quindi il suo Netscape o il suo Internet Explorer, a seconda dei suoi gusti e delle sue preferenze personali. Abbiamo indipendenza dal server applicativo, in quanto andiamo ad utilizzare un'interfaccia Web che passa attraverso un server Web. E abbiamo un servizio che è scalabile, in quanto possiamo far crescere il numero di utenti che accedono al nostro servizio, semplicemente, scalando in maniera orizzontale, o verticale, quelle che sono le caratteristiche del server Web che dà accesso al nostro servizio. Infine, come abbiamo detto poco fa, possiamo espandere facilmente i servizi di intranet verso utenti Internet.

CGI: Limitazioni

Abbiamo visto, quindi, che una tipica architettura a tre livelli è la cosiddetta architettura CGI. Questa è stata la prima architettura sviluppatasi per distribuire applicazioni a tre livelli Web-oriented, ma non è l'unica. In particolare, l'architettura CGI ha una serie di applicazioni in quanto la sua semplicità

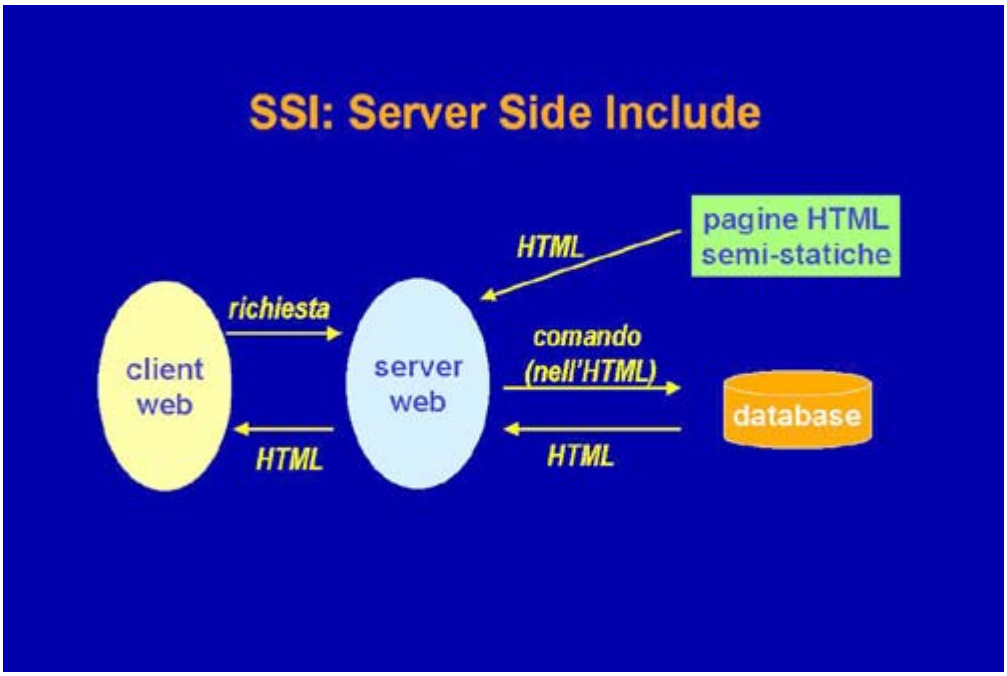
consente un rapido sviluppo delle applicazioni, ma ci sono due grosse limitazioni: la prima è che il processo di attivazione di un processo CGI è molto inefficiente e introduce una certa latenza. Infatti, per ogni richiesta CGI il server Web dovrà lanciare un nuovo processo, separato da quello nel quale sta eseguendo il server Web stesso. Questo è un procedimento che consuma molte risorse e che richiede tempo. L'efficienza, quindi, in condizioni di carico molto elevate, di un programma CGI, non potrà andare oltre una certa soglia. L'altro problema legato all'uso di architetture CGI è dovuto al tipo di linguaggio specifico che si utilizza. Molto spesso per sviluppare applicazioni CGI si utilizzano linguaggi interpretati, questo fa sì che le prestazioni non siano eccezionali. Soprattutto nel caso di applicazioni computation intensive, le prestazioni di un'architettura di tipo CGI possono non raggiungere quelli che sono i requisiti del servizio che intendo offrire. Ovviamente, posso decidere di utilizzare un linguaggio non interpretato, un linguaggio compilato, per sviluppare le mie applicazioni CGI, ma rimane comunque la prima delle due limitazioni indicate, legate al procedimento con cui deve essere lanciato un processo del tutto indipendente dal server Web. Vedremo, quindi, che sono state sviluppate una serie di architetture, anch'esse a tre livelli, che forniscono un'alternativa all'architettura CGI stessa. Parleremo in particolare dell'architettura Server Side Include, parleremo dell'architettura Web API e dell'architettura a codice mobile.

CGI: Common Gateway Interface



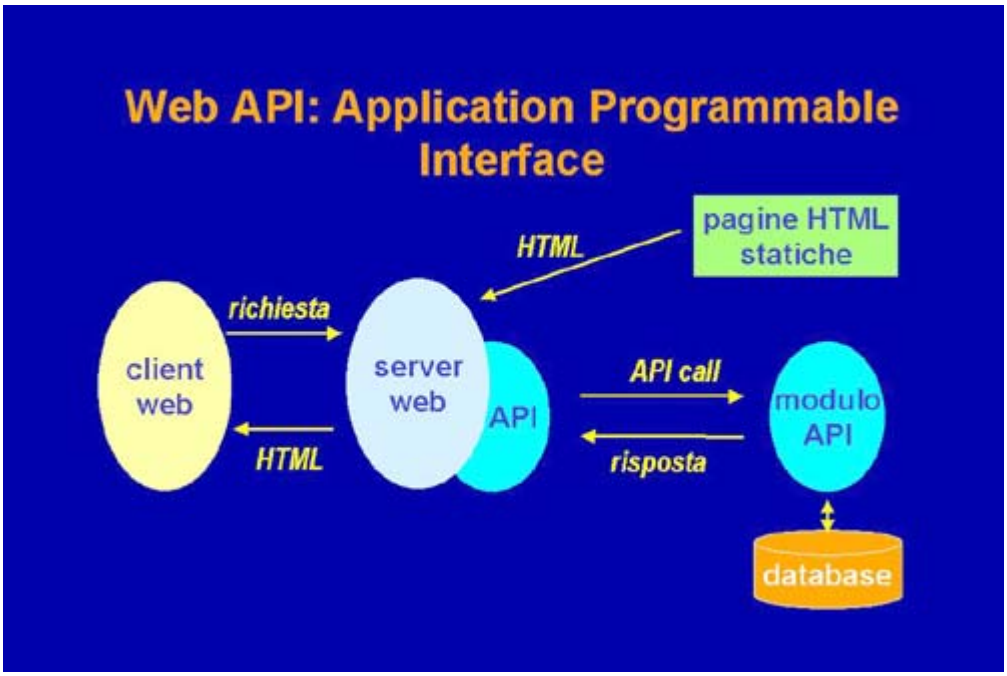
Riprendendo quelli che sono gli elementi di una applicazione CGI, ritroviamo in questa slide che un'architettura CGI fa uso di un client che invia una richiesta HTTP. Questa richiesta HTTP, rivolta al server Web, anziché essere diretta ad ottenere una pagina HTML statica, è diretta ad invocare un programma CGI che, attraverso il protocollo CGI, viene attivato e consente di eseguire delle elaborazioni al volo. Negli esempi che abbiamo visto nella lezione precedente abbiamo visto delle semplici elaborazioni che coinvolgevano soltanto il programma CGI, ma questa architettura, generalmente, può essere estesa per andare ad interrogare dal nostro programma CGI un sistema informativo più complesso, come ad esempio un data base. Il programma CGI deve occuparsi, quindi, di formattare l'output in formato HTML in modo da poterlo passare al server Web, che si occuperà semplicemente di girarlo al client Web. In questo modo riusciamo ad invocare un processo, il programma CGI, che sta sul server Web e che esegue delle elaborazioni in tempo reale a partire da un normale client Web.

SSI: Server Side Include



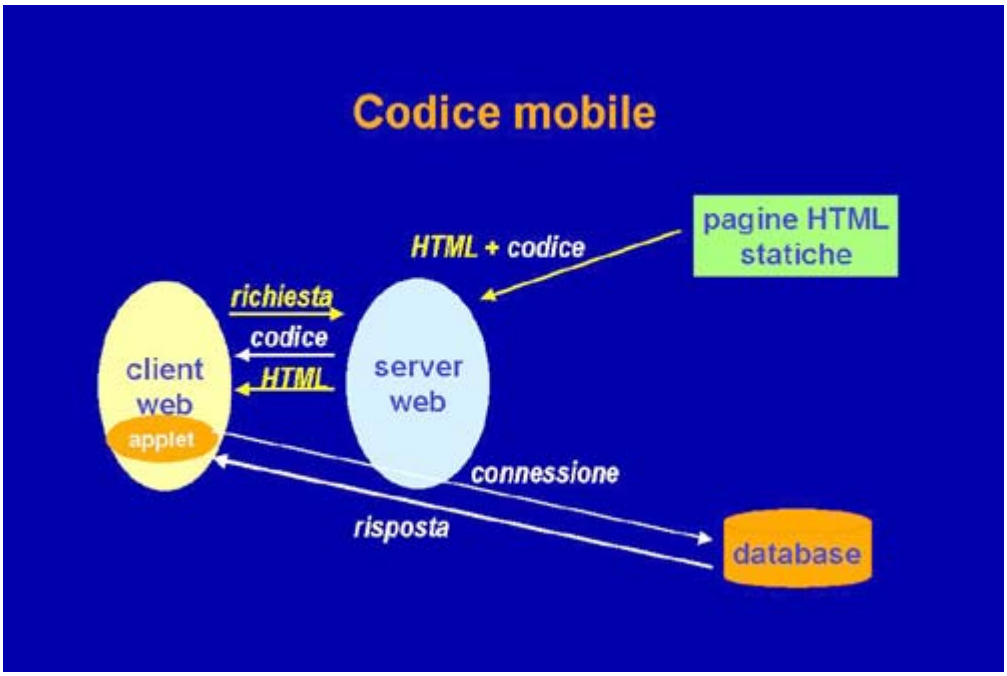
Questa, abbiamo detto, non è l'unica architettura che abbiamo a disposizione, ma ne esistono altre con i loro vantaggi e i loro svantaggi. Cominciamo dall'architettura cosiddetta Server Side Include. In questo caso si cerca di ovviare alla complessità, nel caso CGI, di avere separate le pagine dinamiche e le pagine statiche. In questo caso, in particolare, le pagine HTML contengono delle direttive, dei comandi, che vengono eseguiti dal server prima di distribuire la pagina richiesta. Per questo motivo questa architettura viene detta Server Side Include, perché le pagine HTML sul lato server contengono delle direttive che vengono eseguite sul lato server prima che la pagina venga distribuita. Il procedimento, in questo caso, prevede ancora che il client Web invii la sua richiesta HTTP, questa richiesta va ad invocare una pagina HTML semi-statica la quale, prima di esser distribuita dal server Web, verrà eseguita dal server Web stesso e gli eventuali comandi contenuti nell'HTML consentiranno, ad esempio come indicato in questa slide, di accedere al nostro sistema informativo. L'output di questi comandi, che sarà ancora in formato HTML, verrà sostituito nell'esatto punto in cui quella direttiva semi-statica era contenuta. Il server, a questo punto, avrà confezionato una pagina completamente HTML da spedire al client. Notate che per il client non c'è nessuna differenza rispetto a quanto avveniva nel meccanismo CGI, in quanto ad una richiesta HTTP fa eco una risposta HTTP che contiene una pagina HTML. Il client non percepisce in alcun modo il fatto che quella pagina HTML sia stata creata al volo con un meccanismo di Server Side Include piuttosto che con un meccanismo di tipo CGI.

Web API: Application Programmable Interface



Un terzo tipo di approccio è quello che va sotto il nome di Web API (Application Programmable Interface). Che cosa significa? Significa che, come mostrato in questa slide, i server Web vengono forniti con un'interfaccia con la quale possono essere estesi. Ovvero possono essere definiti dei moduli, i cosiddetti moduli API, che estendono quelle che sono le funzionalità di base del server Web. Se vogliamo sviluppare una particolare applicazione che andrà ad elaborare delle richieste eseguite sul nostro server Web, possiamo andare a costruire un modulo API che risponde alle chiamate previste dall'interfaccia API di quel server. Questo modulo dovrà essere, ovviamente, una libreria dinamica collegata al server Web. Questo ne consentirà delle caratteristiche molto interessanti in termini di prestazioni e di performance, in quanto il modulo API diventa un estensione strettamente collegata del server Web stesso. In particolare vedremo, nella prossima lezione, un esempio di questo tipo di architettura e in particolare vedremo l'esempio Microsoft delle cosiddette ISAPI (Information Server API) che sono il meccanismo API previsto dalla architettura Microsoft. Le caratteristiche di questo tipo di architettura sono innanzi tutto le elevate prestazioni. Si tratta in effetti di un estensione del server Web stesso, strettamente collegata al server, che non richiede, come nel caso del programma CGI, un processo separato da attivare ad ogni invocazione. Le performance di una soluzione di questo tipo, quindi, saranno molto più elevate delle performance offerte da una soluzione basata su architetture CGI o su architetture di tipo Server Side Include.

Codice mobile



Esiste, infine, un quarto modello di architettura a tre livelli ed è il cosiddetto modello a codice mobile. In questo caso, vedremo, oltre ad inviare sul client del codice HTML, si provvede ad inviare del codice che verrà eseguito sul client stesso: dei piccoli programmi che sfrutteranno le capacità di elaborazione del client. Questo fa sì che per interazioni con l'utente non sia necessario accedere nuovamente alla rete, ma queste possano essere fatte in locale dal pezzo di codice inviato dal server sul client stesso. L'architettura è quella indicata in questa slide e, come vedete, in questo caso ad una richiesta HTTP proveniente dal client corrisponde una risposta del server contenete non solo HTML ma anche del codice, che verrà inviato dal server verso il client. Questo codice verrà, a questo punto, eseguito sul client stesso. In questo caso abbiamo l'esempio di una applet, una applicazione proveniente dalla rete, che verrà eseguito sulla macchina virtuale del client stesso. Questo codice potrà, a sua volta, effettuare ulteriori connessioni verso la rete ed accedere a risorse che si trovano, ad esempio, su un sistema informativo remoto. Il sistema informativo potrà, a questo punto, rispondere a questa richiesta di connessione, inviando i dati richiesti. In questo caso, notate, che l'architettura prevede che il client sia in grado di eseguire delle applicazioni provenienti dalla rete. Questo tipo di architettura richiederà quindi dei client particolari, in grado di eseguire questo tipo di codice. Vedremo che le architetture principali che prevedono questo meccanismo sono le architetture basate su Java e quelle basate su ActiveX, in ambiente Microsoft. Vedremo anche che ci sono una serie di linguaggi di script Web che possono essere utilizzati per spedire applicazioni dal server verso il client.

SSI per Apache Web Server

Proseguiamo in questa analisi dell'architettura a tre livelli e cominciamo ad analizzare alcune specifiche soluzioni che prevedono l'utilizzo di una di queste architetture. Nel corso di questa lezione vedremo, ancora, come è organizzata la soluzione Server Side Include per il Web server Apache. Nella lezione successiva ci occuperemo invece di soluzioni tipicamente Microsoft; vedremo in particolare l'architettura ASP che è anch'essa una soluzione di tipo Server Side Include e vedremo la soluzione ISAPI, che invece è una soluzione di tipo Web API. Quasi tutti i principali server Web consentono di utilizzare Server Side Include nelle pagine HTML. Andiamo a considerare un server di esempio, il server Apache, che è uno dei server più diffusi e tra i più utilizzati in Internet, e andiamo a vedere in che modo è possibile inserire delle Server Side Include in questo tipo di server. Questo ci consentirà di comprendere meglio quali sono gli aspetti architetturali di questo tipo di soluzione. In particolare, il server Apache ha una direttiva include che consente di inserire all'interno di una pagina HTML un altro file, oppure di eseguire una serie di comandi built-in (comandi supportati dal server stesso), oppure che consente di eseguire dei processi che si trovano sul server Web stesso.

Esempio: SSI per Apache (1)

Andiamo ad analizzare come questa soluzione, di tipo Server Side Include, sia supportata nel server Web Apache e cerchiamo di comprendere bene quali sono i concetti strutturali e procedurali. Nell'esempio, molto semplice, che andiamo a vedere, abbiamo un documento HTML che contiene una direttiva Server Side Include. In questo caso la sintassi mostrata è quella prevista dai server Web Apache. Altri server Web avranno sintassi simili che consentono di realizzare funzionalità equivalenti. Notate che la pagina HTML è quasi statica, nel senso che contiene informazioni HTML tradizionali, come ad esempio questa linea dove viene scritto: la data di oggi è, ma contiene anche dei comandi che devono essere eseguiti ed interpretati dal server prima di distribuire la pagina. In particolare, notate il comando echo, che esegue la visualizzazione della variabile specificata nell'attributo var, che in questo caso si chiama DATE_LOCAL. Questa variabile conterrà, quindi, la data di sistema del server al quale ci colleghiamo. Notate che queste direttive di Server Side Include sono in mezzo all'HTML, occorrerà quindi utilizzare dei meccanismi per farle riconoscere al server Web nel momento in cui questa pagina viene distribuita. In questo caso non è un vero e proprio commento, ma è un commento HTML che inizia con questa sequenza di 4 caratteri a cui segue immediatamente un carattere #. Questo è il segnale che il server utilizza per capire che si tratta di una direttiva di Server Side Include, e quindi, per eseguire il comando built_in echo che visualizza il valore di questa variabile.

Esempio: SSI per Apache (2)

Spostiamoci quindi sul PC e andiamo a vedere come funziona questa pagina HTML. Dobbiamo andare a specificare, ovviamente, l'URL di questo documento e in particolare, in questo caso, utilizziamo l'URL testssi.shtml che individua il file. Dicendo al nostro browser di caricare questa pagina, la pagina verrà caricata e il server, prima di spedire questa pagina al client, andrà a sostituire la direttiva Server Side Include con l'output di quel comando. In questo caso si trattava del contenuto della variabile DATE_LOCAL, che specifica qual è la data locale e qual è il fuso orario corrente sul nostro server Web. Notate, e lo possiamo vedere andando ad analizzare il sorgente di questa pagina HTML ricevuta, che il contenuto di questa pagina è un contenuto puramente HTML. Il server, prima di spedire questa pagina, ha sostituito l'output del comando echo e ha consentito quindi di passare un parametro generato in tempo reale dal server stesso. Questo esempio chiude questa prima lezione sulle architetture a tre livelli e in particolare ci mostra un esempio della architettura Server Side Include per il server Web Apache. Nella prossima lezione andremo a vedere quali sono le soluzioni a tre livelli per i server Microsoft, in particolare andremo a vedere l'architettura ASP e l'architettura ISAPI.