Blazor

Ricordate come funzionava (e funziona ancora…) **ASP.NET Web Forms**? Una pagina HTML, all’interno di una grande form, con un campo nascosto (il famoso **ViewState**) contenente una stringa crittografata rappresentante lo stato della pagina, cioè il valore corrente di tutti i controlli server nella pagina. Questa form, mandata al server ogni volta che l’interazione dell’utente richiedesse un aggiornamento dell’interfaccia, scatenava l’esecuzione del motore di ASP.NET Web Forms, che aggiornava l’HTML e il valore del ViewState in base agli handler del code behind.

Web Forms cercava solo di automatizzare questo processo, per nascondere i dettagli di funzionamento del protocollo ai programmatori Microsoft che venivano dal mondo Desktop, che erano abituati a ragionare in termini di Form, controlli e Handler di eventi nel code behind.

Questa astrazione portava con se però una grossa inefficenza in termini di scambio dati e di esperienza utente.

La cosa fu mitigata con l’arrivo dell’ **Update Panel**, che rendeva asincrono lo scambio utilizzando AJAX in maniera trasparente al programmatore, risolvendo il problema del refresh.

Quando questa astrazione è diventata ingombrante e abbiamo accettato l’idea di doverci scrivere l’HTML (anzichè generarlo dai controlli server) e controllare l’interazione con il server, utilizzando JavaScript dove necessario, siamo passati a **ASP.NET MVC**

A un certo punto però, per ragioni che avremmo compreso fino in fondo solo anni dopo, il legame tra ASP.NET e IIS è cominciato ad andare un po’ stretto.

Questo portò all’introduzione di [**OWIN**](http://owin.org/), uno standard aperto che definisce una interfaccia tra un web server .NET e una applicazione web.

Con la versione 4 di MVC, fornendo di fatto la possibilità di far girare applicazioni ASP.NET MVC su web server diversi da IIS.

 Ma la vera ragione che portò a un momento di rottura fu il fatto che ASP.NET non era stato creato pensando agli scenari Cloud, e Microsoft stava per puntare tutto sul suo Cloud: **Microsoft Azure**.

Nel 2016 fu rilasciata la prima versione di **ASP.NET Core**.