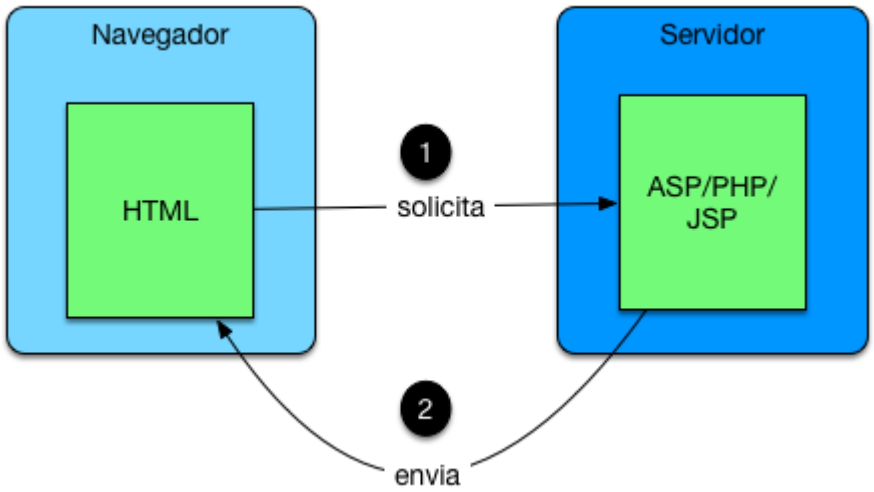
La historia de las Arquitecturas web es ya muy amplia . Hemos pasado por muchos posibles enfoques y soluciones sobre como construir una aplicación web y siempre llegan nuevas ideas que hacen al sector evolucionar. Vamos a echar un vistazo a los diferentes enfoques que se han producido en los últimos 20 años.

El Modelo cero y el código spaguetti

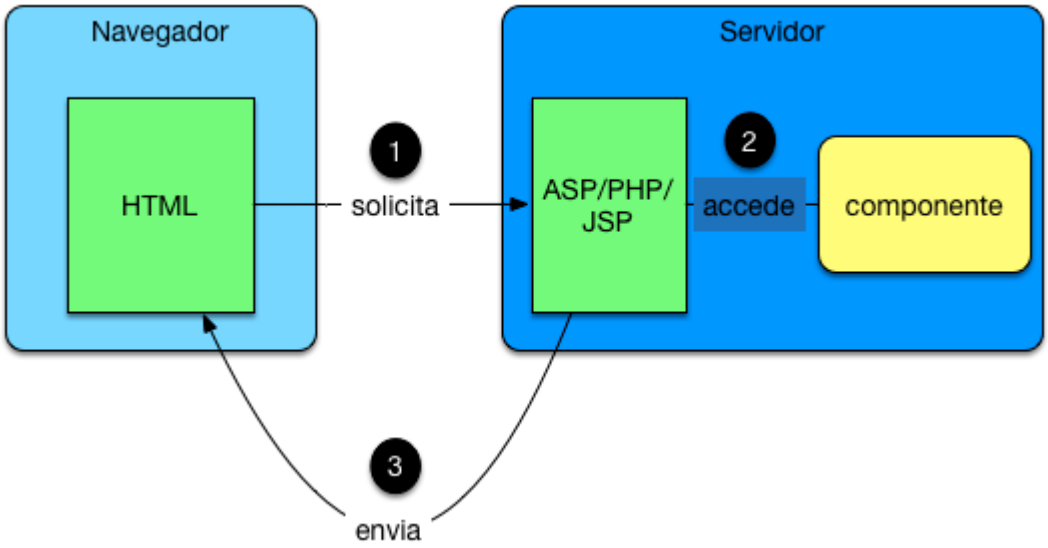
Esta es una forma de llamar a las soluciones iniciales cliente/servidor en las que tenemos una página JSP/ASP/PHP que se conecta a una base de datos y genera un nuevo contenido html. Es código spaguetti, el que nadie quiere tener que pero que en su momento se usó mucho. A día de hoy lo puedes encontrar en más sitios de los que se piensa. Como ventaja fundamental destaca su sencillez a nivel de arquitectura y como desventaja su poca flexibilidad y nula capacidad de reutilización.



El Modelo 1

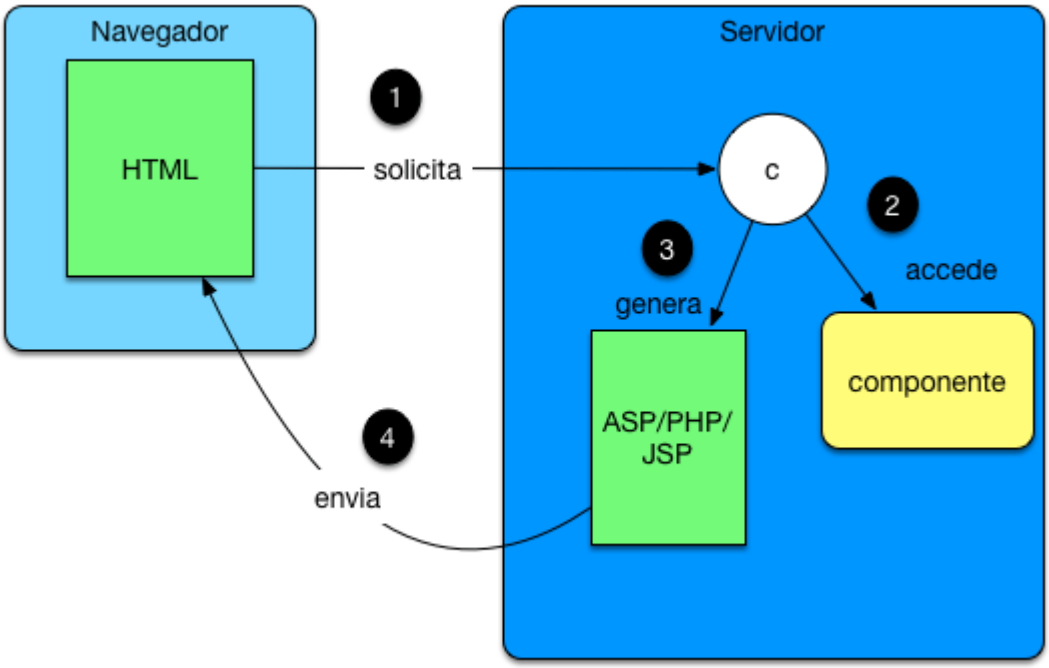
Este fue un salto importante , promueve la modularización y el uso de componentes a través de la programación orientada a objeto. Fue desde mi punto de vista un gran salto. Hay enfoques tipo Rails que se apoyaron más en un patrón tipo Active Record y otros en

Servicios y Repositorios como Spring. La clave fundamental es generar componentes en la capa de backend y aumentar la reutilización de esa parte.



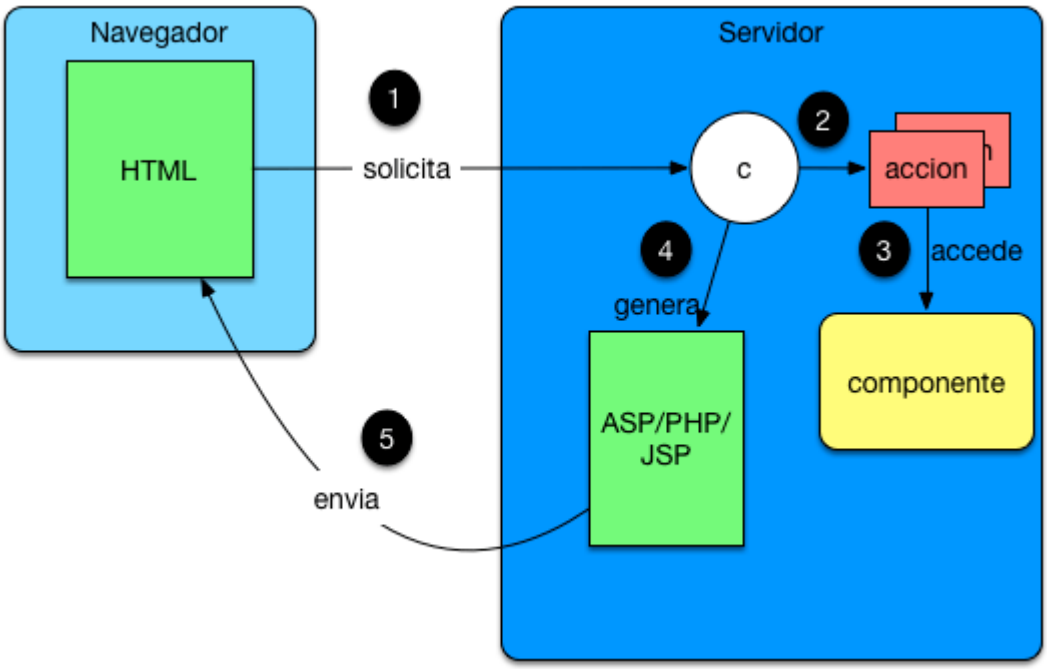
El modelo 2 (MVC)

Quizás para mucha gente el modelo más importante, se apuesta por la separación de responsabilidades entre Vista , Controlador y Modelo. Prácticamente todos los frameworks web han implementado este enfoque de una forma u otra. Ejemplos son: Struts en Java , ASP.NET en Microsoft o Laravel en PHP.



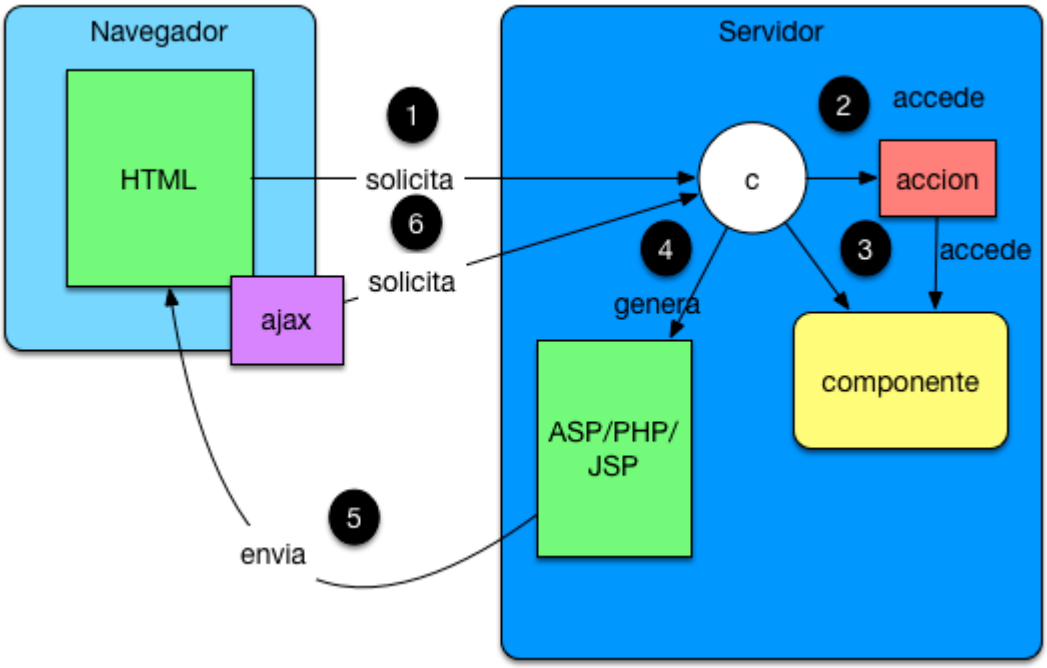
El Modelo MVC 2 FrontController/Enrutador

Una evolución importante del modelo MVC fue el modelo MVC 2 o de FrontController que apuesta por una arquitectura en la que únicamente hay un controlador principal y gestiona todo a través de acciones. Esta es un poco la idea de muchos de los frameworks modernos con el concepto de router en la capa de presentación. El enfoque más clásico puede ser Struts en el lado del servidor o Spring MVC.



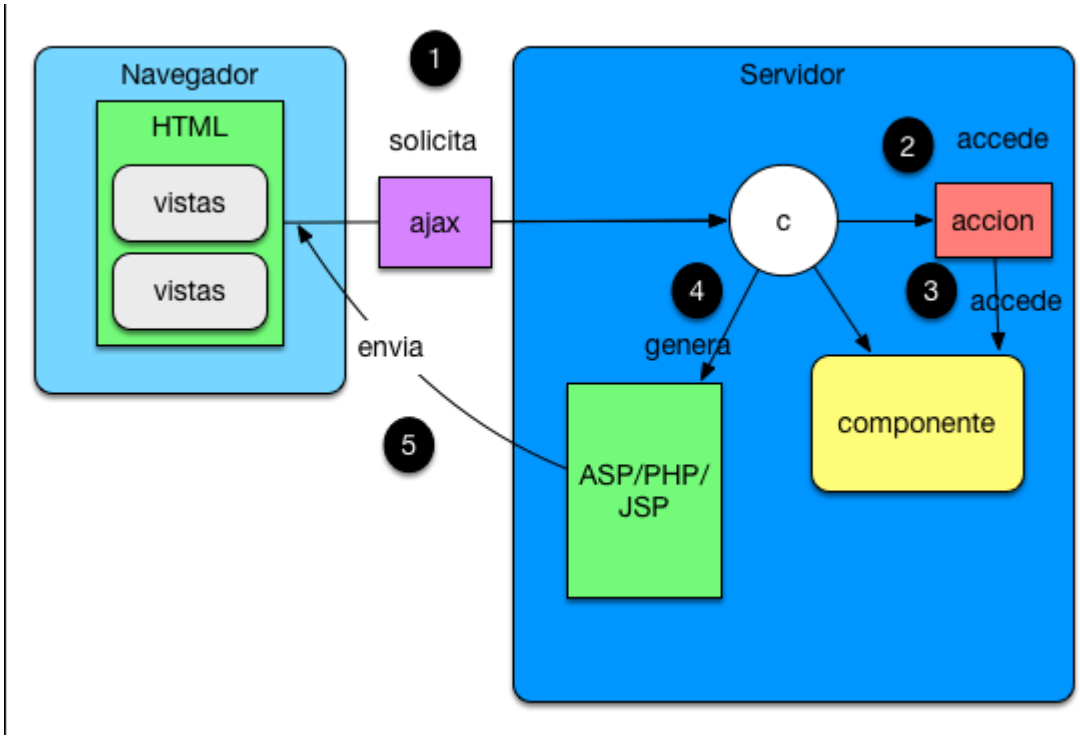
Arquitecturas Web y Ajax

Hasta ese momento todas las evoluciones se produjeron del lado del servidor. El lado cliente tenía pocas novedades. Es en esta situación cuando surge AJAX como tecnología para mejorar el rendimiento entre cliente y servidor. Esto supuso una verdadera revolución a la forma de programar.



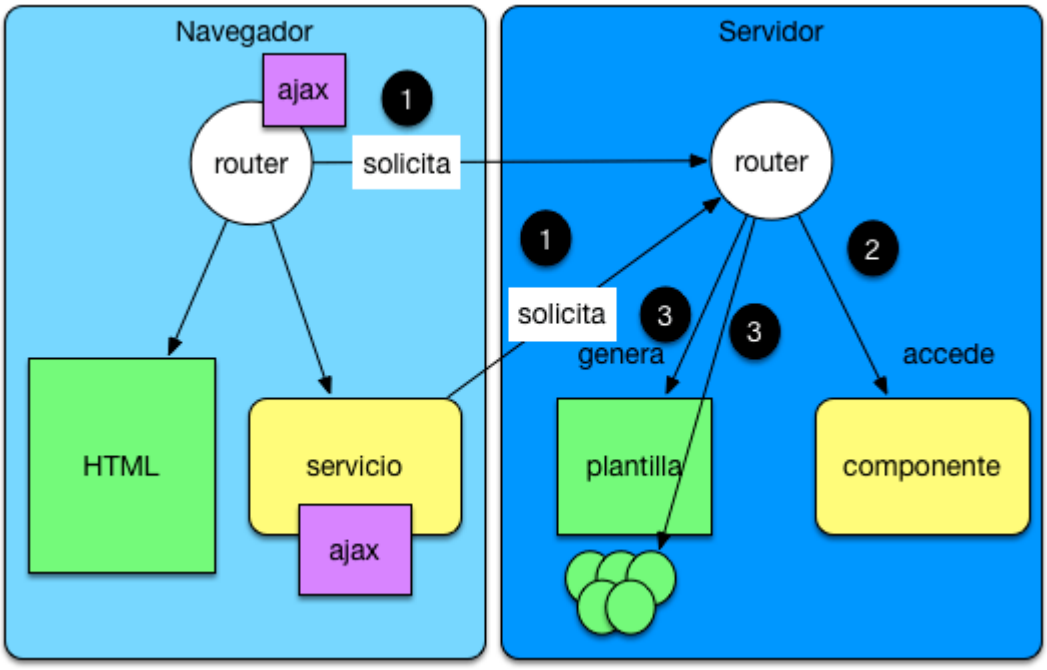
Arquitecturas Web y SPA

Con la llegada del mundo móvil y la necesidad de tener las aplicaciones web cada vez más desconectadas surgen las arquitecturas SPA (Simple Page Application) . Su propuesta principal es dar mayor responsabilidad al navegador y que el se encargue de cargar las vistas y datos utilizando AJAX.



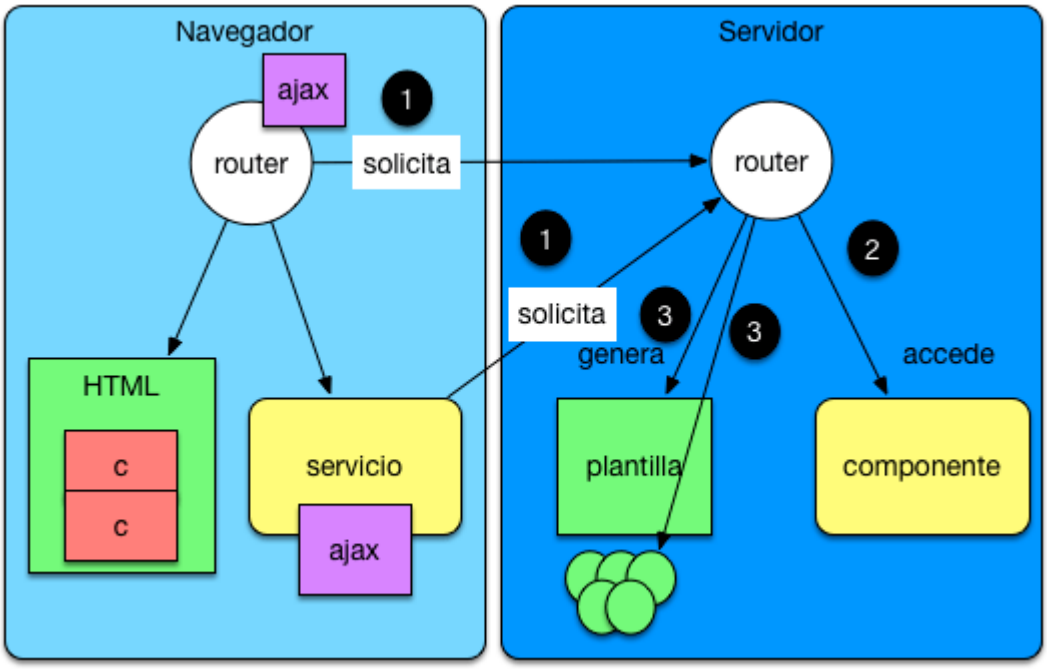
Arquitecturas SPA MVC

Poco a poco el lado cliente comienza a tener más peso en los desarrollos y se necesita organizar mejor el código de JavaScript . Aparecen los primeros frameworks MVC de cliente como Backbone.js que permiten dividir las responsabilidades de la misma forma que en el servidor.



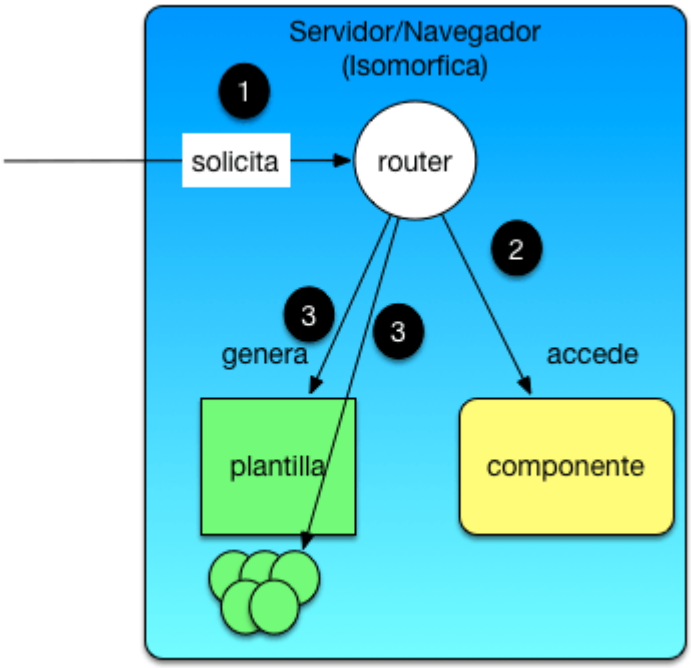
Arquitecturas SPA MVC y uso de componentes

Estas arquitecturas empiezan a madurar rápidamente y aparecen tecnologías como Angular.js que promueve el uso del modelo MVC y la utilización de componentes en capa de presentación . Aparecen librerías complementarias como React que se centran en estos últimos.



Arquitecturas Web Isomórficas

Ahora mismo estamos entrando en otra fase , comienza a llegar el JavaScript Isomórfico . Si nos fijamos en el último diagrama la parte cliente y la parte servidor son muy parecidas. ¿Qué sucedería en el caso de que ambas partes estuvieran implementadas en JavaScript? . Pues que probablemente mucho código se podría compartir y según se ejecutara la aplicación en cliente o en servidor el comportamiento variaría.



Este es un pequeño resumen de las diferentes arquitecturas que hemos estado usando a la hora de desarrollar aplicaciones web y las que usaremos.

Otros artículos relacionados : Java y su evolución , Arquitecturas SPA