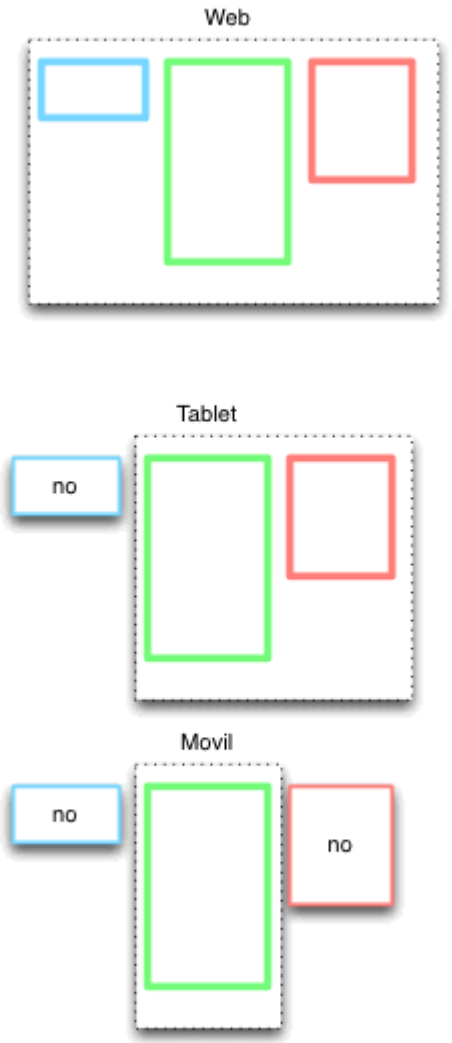
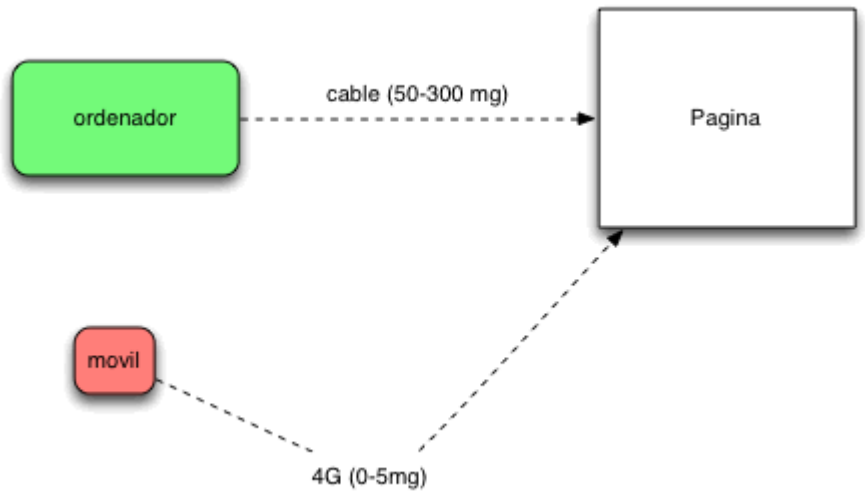
¿Qué es Mobile RESS? . Habitualmente cuando los desarrolladores necesitan adaptar sus aplicaciones a dispositivos móviles optan por utilizar técnicas de Responsive Web Design . Estas técnicas permiten que nuestra aplicación web se muestre de una forma diferente dependiendo si el entorno es tablet, móvil o web. Para ello normalmente nos apoyamos en CSS y Media Queries que obrarán el milagro.



El resultado suele ser bueno , pero hay que pagar un precio y es que la página web sigue

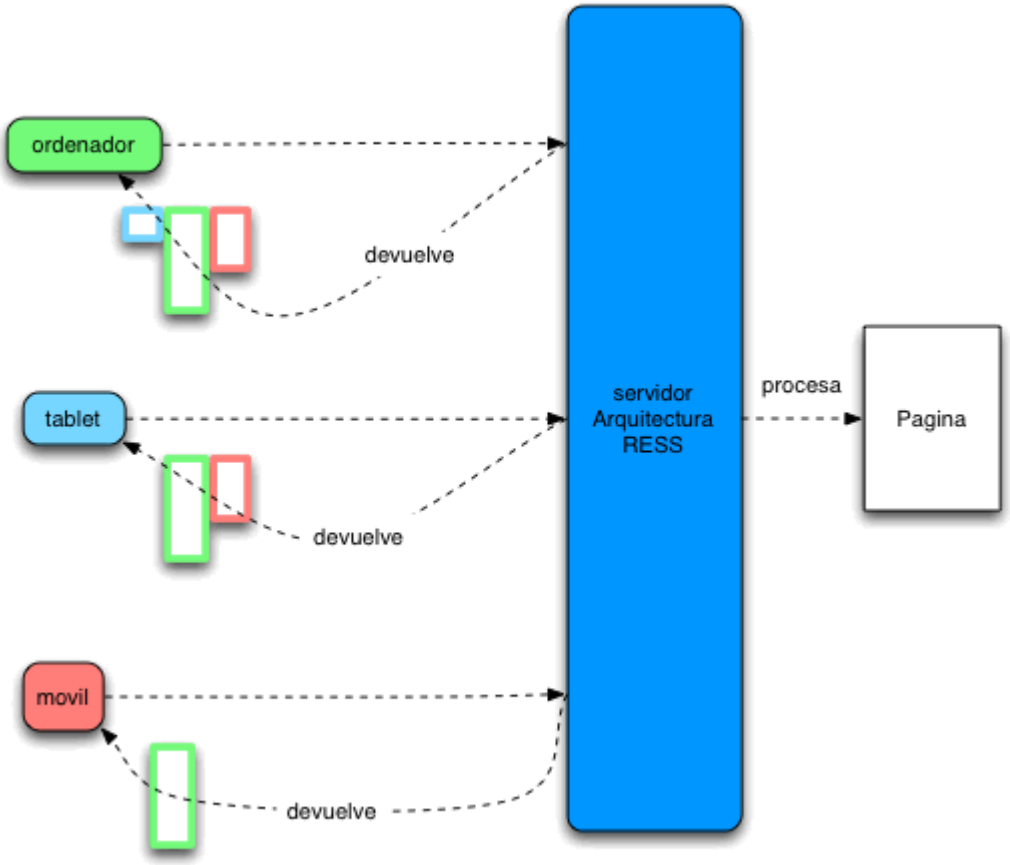
siendo la misma aunque partes de ella no se muestren. Esto puede ser un problema, cuando nuestra página web es accedida desde un ordenador estamos conectados a internet con una buena línea. Sin embargo esto no tiene porque ser cierto cuando hablamos de un dispositivo móvil.



Para solventar este tipo de problemas necesitamos otro enfoque a nivel de Arquitectura. No es suficiente con utilizar Responsive Web Design. El problema se volverá más seguro cuanto más usuarios accedan a la aplicación desde localizaciones remotas o con mala cobertura móvil.

Mobile RESS (Responsive Web Design Server Side)

La solución viene a partir de un enfoque del lado del servidor. Si queremos mejorar el rendimiento de las aplicaciones móviles el servidor deberá enviar a cada dispositivo la información que le encaja y no toda la página.



Mobile RESS y librerías

Implementar este tipo de soluciones no es excesivamente complejo y existen clases y librerías en todas las plataformas que nos ayudan a implementarla de forma rápida. Una de las más habituales sin complicarnos demasiado la vida es MobileESP que tiene un gran soporte multiplaforma y aporta una clase denominada UAgentInfo que dispone de un amplio conjunto de métodos para decirnos que dispositivo se esta conectando en estos momentos a nuestra web. Vamos a ver un ejemplo de código a través de un JSP sencillo.

```
&lt;%@ page language=&quot;java&quot; contentType=&quot;text/html;
```

```
charset=UTF-8&quot;
 pageEncoding=&quot;UTF-8&quot;%&gt;
 &lt;%@page import=&quot;com.handinteractive.mobile.*&quot; %&gt;
&lt;!DOCTYPE html PUBLIC &quot;-//W3C//DTD HTML 4.01
Transitional//EN&quot;
&quot;http://www.w3.org/TR/html4/loose.dtd&quot;&gt;
&lt;html&gt;
&lt;head&gt;
&lt;meta http-equiv=&quot;Content-Type&quot; content=&quot;text/html;
charset=UTF-8&quot;&gt;
&lt;title&gt;Insert title here&lt;/title&gt;
&lt;/head&gt;
&lt;body&gt;

&lt;%
UAgentInfo agente=new UAgentInfo(request.getHeader(&quot;User-
Agent&quot;),request.getHeader(&quot;Accept&quot;));
if (agente.detectTierTablet()) {

out.println(&quot;estamos en una tableta&quot;);
}
else if (agente.detectTierIphone()) {

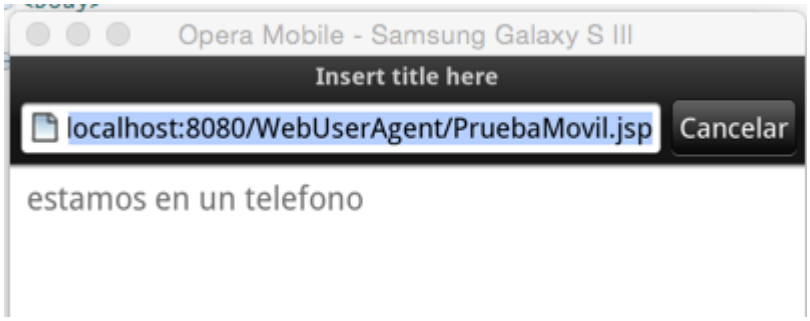
 out.println(&quot;estamos en un telefono&quot;);
}else {
 out.println(&quot;estamos en la web&quot;);
}

 %&gt;

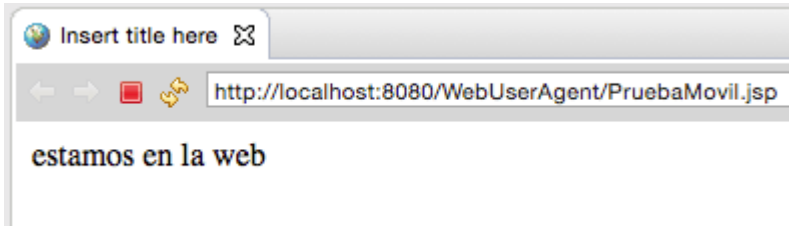
&lt;/body&gt;
```

```
&lt;/html&gt;
```

Dependiendo del dispositivo con el que se cargue la web la página JSP generará una estructura de información u otra. En el caso de un emulador de telefono como Opera Mobile Emulator:



En el caso de la web clásica:



Otros artículos relacionados: Arquitecturas SPA ,HTML5 History API