el estado del recurso fuera. Internamente el estado del recurso puede ser cualquier cosa desde una base de datos relacional a un fichero de texto.

- Mensajes autodescriptivos. REST dicta que los mensajes HTTP deberían ser tan descriptivos como sea posible. Esto hace posible que los intermediarios interpreten los mensajes y ejecuten servicios en nombre del usuario. Uno de los modos que HTTP logra esto es por medio del uso de varios métodos estándares, muchos encabezamientos y un mecanismo de direccionamiento. Por ejemplo, las cachés Web saben que por defecto el comando GET es cacheable (ya que es side-effect-free) en cambio POST no lo es. Además saben como consultar las cabeceras para controlar la caducidad de la información. HTTP es un protocolo sin estado y cuando se utiliza adecuadamente, es posible es posible interpretar cada mensaje sin ningún conocimiento de los mensajes precedentes. Por ejemplo, en vez de logearse del modo que lo hace el protocolo FTP, HTTP envía esta información en cada mensaje.

- Hipermedia como un mecanismo del estado de la aplicación. El estado actual de una aplicación Web debería ser capturada en uno o más documentos de hipertexto, residiendo tanto en el cliente como en el servidor. El servidor conoce sobre le estado de sus recursos, aunque no intenta seguirle la pista a las sesiones individuales de los clientes. Esta es la misión del navegador, el sabe como navegar de recurso a recurso, recogiendo información que el necesita o cambiar el estado que el necesita cambiar.

En la actualidad existen millones de aplicaciones Web que implícitamente heredan estas restricciones de HTTP. Hay una disciplina detrás del diseño de sitios Web escalables que puede ser aprendida de los documentos de arquitectura Web o de varios estándares. Por otra parte, también es verdad que muchos sitios Web comprometen uno más de estos principios, como por ejemplo, seguir la pista de los usuarios moviéndose a través de un sitio. Esto es posible dentro de la infraestructura de la Web, pero daña la escalabilidad, volviendo un medio sin conexión en todo lo contrario.

Los defensores de REST han creído que estas ideas son tan aplicables a los problemas de integración de aplicaciones como los problemas de integración de hipertexto. Fielding es bastante claro diciendo que REST no es la cura para todo. Algunas de estas características de diseño no serán apropiadas para otras aplicaciones. Sin embargo, aquellos que han decidido adoptar REST como un modelo de servicio Web sienten que al menos articula una filosofía de diseño con fortaleza, debilidades y áreas de aplicabilidad documentada.

¿Cómo sería un ejemplo de diseño basado en REST?

De nuevo tomaremos como ejemplo a la Web. La Web evidentemente es un ejemplo clave de diseño basado en REST, ya que muchos principios son la base de REST. Posteriormente mostraremos un posible ejemplo real aplicado a Servicios Web.