REST Representational State Transfer

REST

- REST es el acrónimo de **Representational State Transfer** y se refiere a un estilo de arquitectura web que tiene muchas características y especifica el comportamiento de clientes y servidores.
- REST define un set de principios arquitectónicos por los cuales se diseñan servicios web haciendo foco en los recursos del sistema, incluyendo cómo se accede al estado de dichos recursos y cómo se transfieren por HTTP hacia clientes escritos en diversos lenguajes

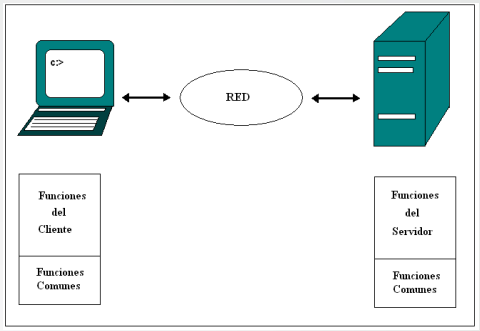
Características

- Origen Roy Thomas Fielding, ámbito académico
- Estilo de arquitectura
- Describe como debería comportarse la Web
- Se apoya en el uso de URI y HTTP
- REST evoluciona en la red

Restricciones / Principios

- Cliente Servidor
- Sin estado
- Caché
- Sistema de capas
- Interfaz Uniforme

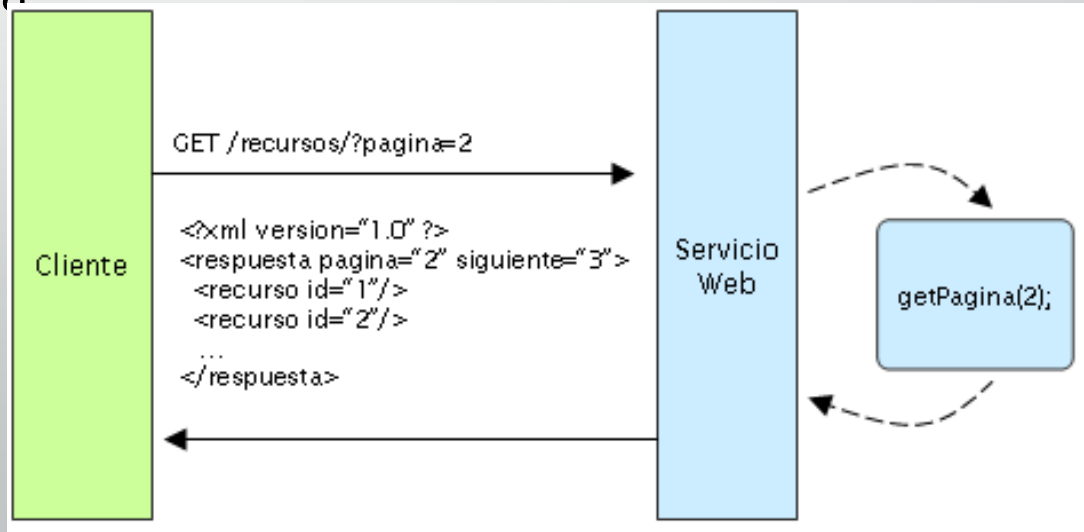
Cliente Servidor



- **Servidor**
- Genera respuestas que incluyen enlaces a otros recursos para permitirle a la aplicación navegar entre los recursos relacionados
- Genera respuestas que indican si son susceptibles de caché o no, para mejorar el rendimiento al reducir la cantidad de peticiones
- **Cliente**
- Utiliza el atributo Cache-Control del encabezado de la respuesta para determinar si debe cachear el recurso
- Envía peticiones completas que pueden ser serviciadas en forma independiente a otras peticiones

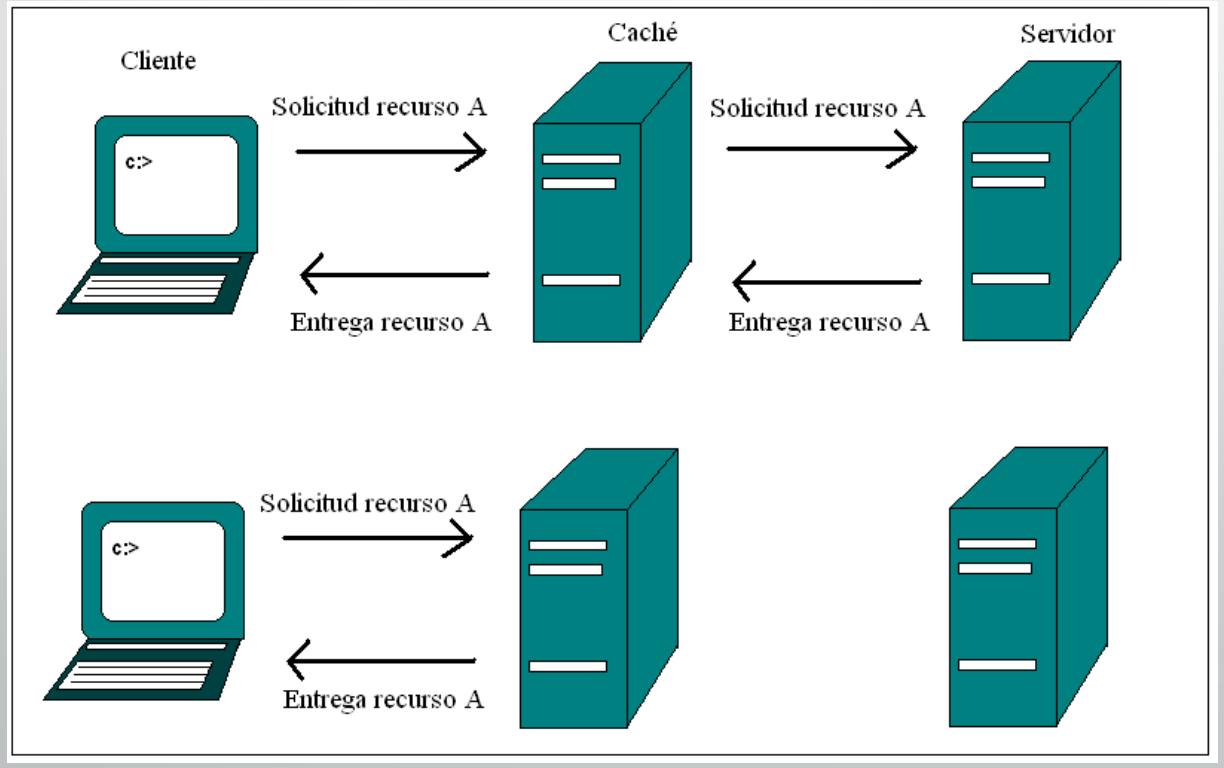
Sin Estado

- los **servicios sin estado** son mucho más simples de diseñar, escribir y distribuir a través de múltiples servidores.
- Un servicio sin estado no sólo funciona mejor, sino que además mueve la responsabilidad de mantener el estado al cliente de la aplicación
- El servidor es responsable de generar las respuestas y proveer una interfaz que le permita al cliente mantener el estado de la aplicación por su cuenta



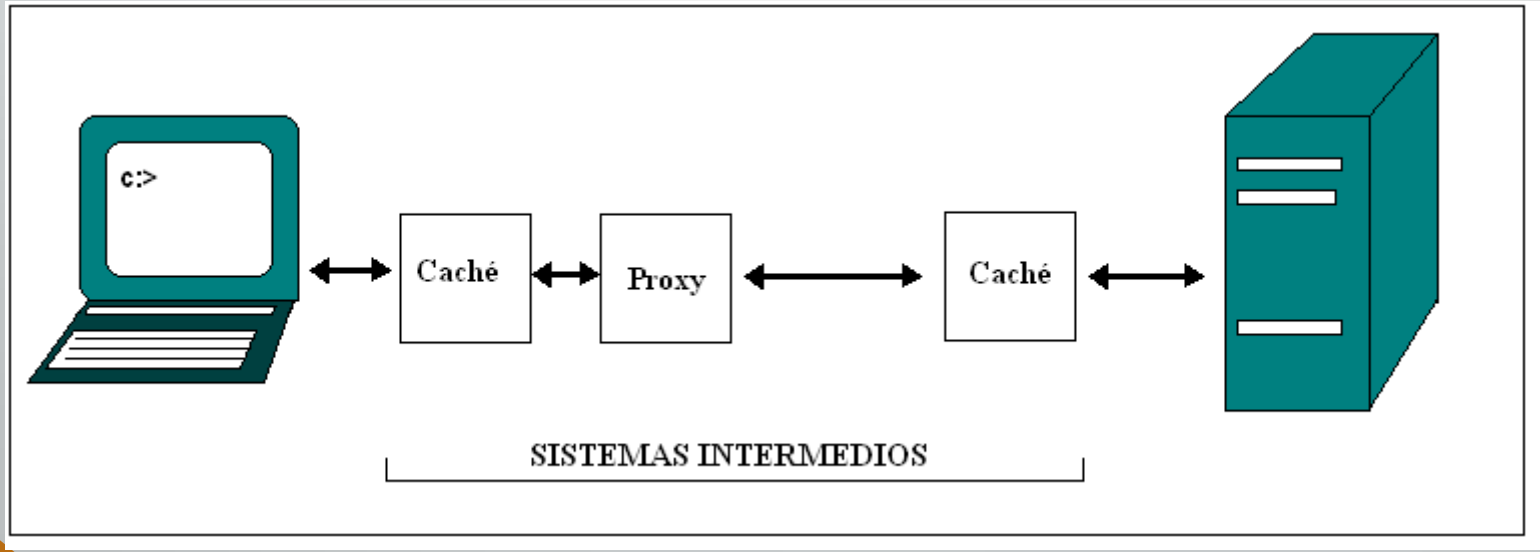
Caché

- El servidor genera respuestas que indican si son susceptibles de caché o no



Sistema de capas

- Los componentes individuales no pueden ver mas allá de la capa inmediata con la cual está interactuando.
- Esto permite que los componentes sean independientes y puedan ser fácilmente reemplazable o extensible.



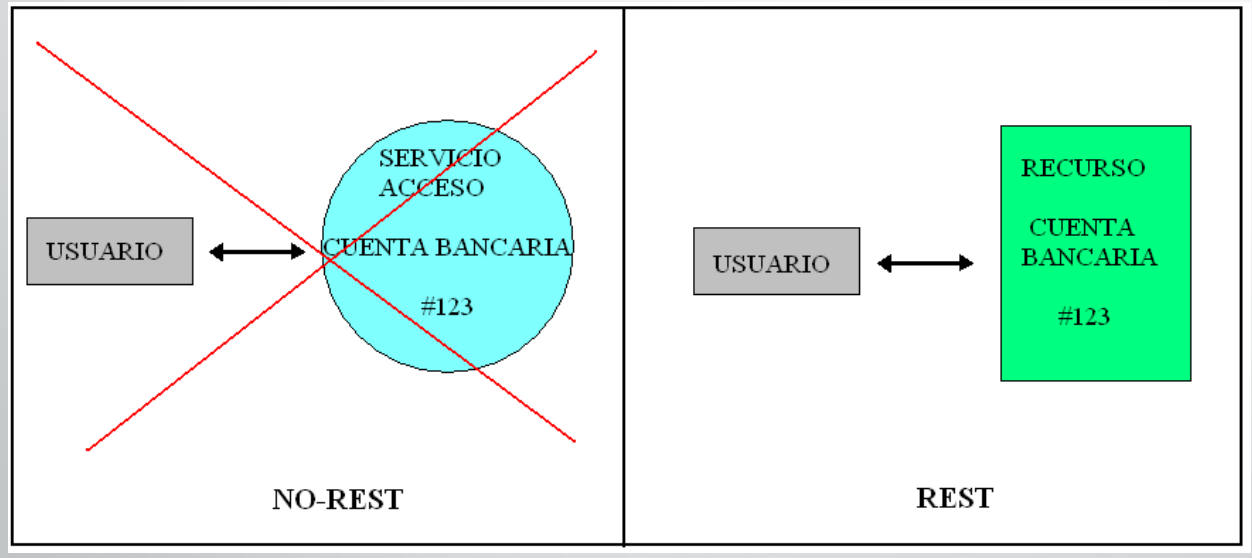
Interfaz Uniforme

- La implementación se separa del servicio que proporciona.

- Mecanismos para conseguirlo:
  - Recursos e identificación de recursos
  - Manipulación de recursos a través de sus representaciones
  - Mensajes Auto-descriptivos
  - Hipermedios como el motor de estado de la aplicación

RECURSOS

- **REST es orientado a recursos y no a métodos**



Identificación del Recurso

- Los elementos de información son identificados por una **URI** (*Identificador Único del Recurso*), que deben presentar las siguientes características:

- Las *URI* recibirán nombres que no deben implicar una acción, es decir, se debe evitar colocar verbos en ellas. Esto se debe a que se pone énfasis a los sustantivos: http://server/api/fotos/12
- Deben ser únicas, no debemos tener más de una URI para identificar un mismo recurso
- Deben mantener una jerarquía lógica

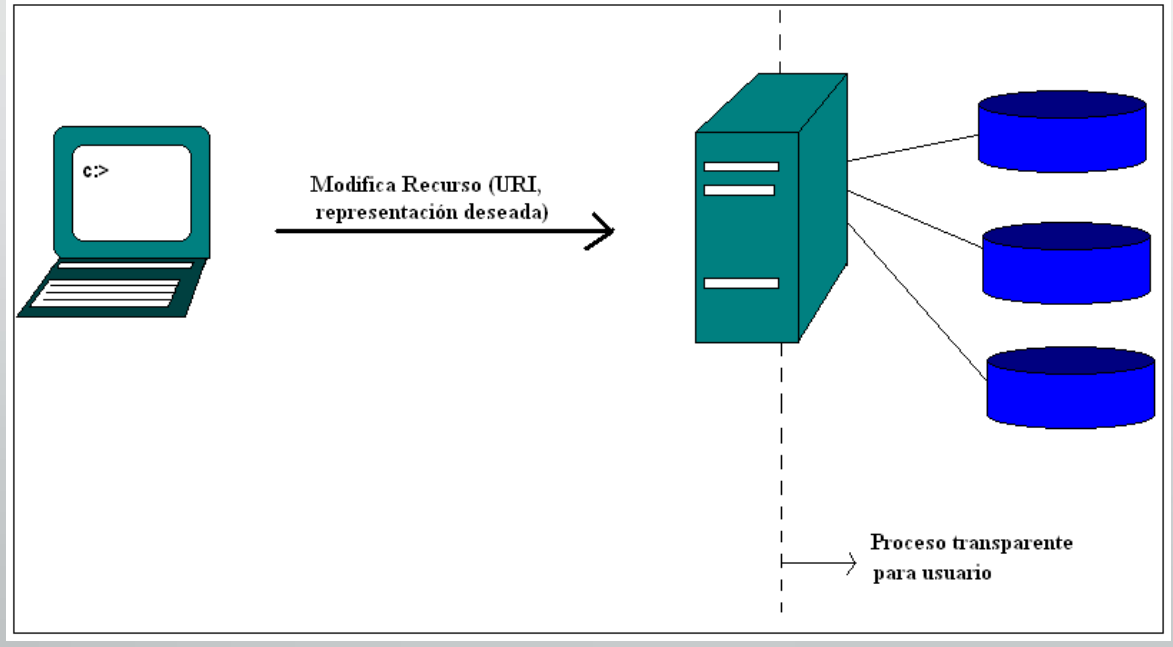
URI

Usa nombres para describir los recursos en lugar de verbos:

Nombres (preferido)	Verbos
GET /user/1234	getUser(1234)
POST /users	createUser(user)
DELETE /player/45	deletePlayer(45)

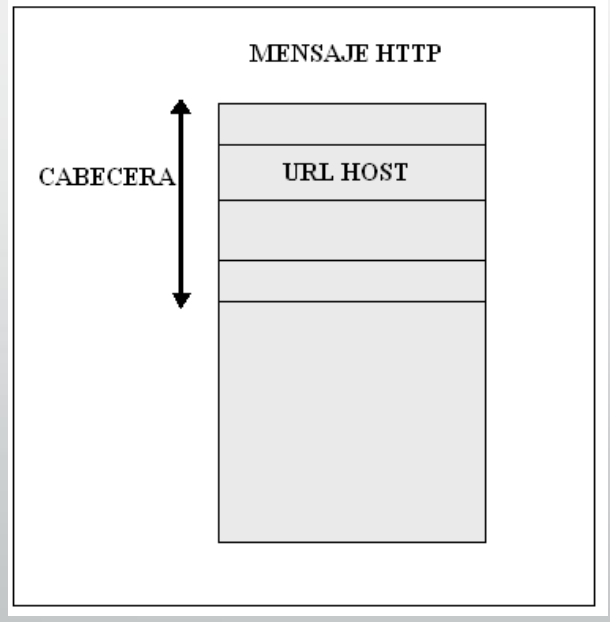
Manipulación de Recursos

- Un cliente manipula la representación de un recurso en vez de su implementación



Mensaje Autodescriptivos

- Toda la información necesaria para procesar el mensaje se encuentra en el propio mensaje.
- Usa HTTP como protocolo de aplicación.



Hipermedio para representar la información

- Emplea *hipermedios* para representar la información, que suelen ser
- **HTML**
- **XML**
- **JSON**

Métodos de REST

- Usa los métodos de HTTP
- Cumple con la restricción de interfaz uniforme

MÉTODO	FUNCIÓN
GET	Solicitar recurso
POST	Crear recurso nuevo
PUT	Actualizar o modificar recurso
DELETE	Borrar recurso

Verb	URI	Description
GET	/posts	Get the post list.
GET	/posts/42	Get a single post (the one with id 42).
DELETE	/posts/42	Delete the post 42.
POST	/posts	Create a post.
PUT	/posts/42	Update the post.
PATCH	/post/42	Partial update.
OPTIONS	/post/42	Retrieve the available operation on the resource.
HEAD	/post/42	Return only the HTTP header.

Códigos de respuesta

Código	Significado
200 – OK	todo funcionó correctamente
201 – CREATED	fue creado un nuevo recurso
204 – NO CONTENT	el recurso fue borrado con éxito
304 – NOT MODIFIED	los datos devueltos provienen del cache (los datos no han cambiado)
400 – BAD REQUEST	la petición es inválida o no puede ser procesada
401 – UNAUTHORIZED	la petición requiere autenticación
403 – FORBIDDEN	el servidor entiende la petición pero la rechaza o el acceso no está permitido
404 – NOT FOUND	no existe el recurso para la URI especificada
500 – INTERNAL SERVER ERROR	indica una falla a nivel de servidor

Diferencias SOAP y REST

SOAP	REST
Origen en el ámbito académico	Origen en el ámbito de las empresas
Orientado a RPC	Orientado a recursos
Servidor almacena parte del estado	El estado se mantiene sólo en el cliente, y no se permiten las sesiones
Usa HTTP como túnel para el paso de mensajes	Propone HTTP como nivel de aplicación