

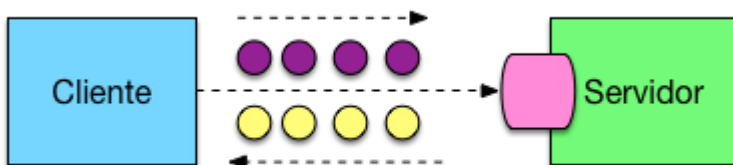
Tabla de contenidos

- ¿ Que es REST ? (Nivel 1 Recursos)
- ¿ Que es REST ? (Nivel 2 HTTP Verbs)
- HATEOAS (Nivel 3)

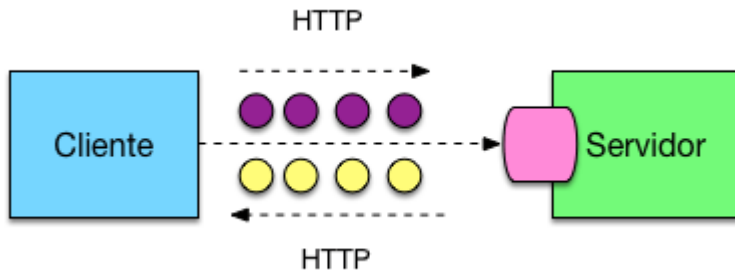
¿ Que es REST ? Esta pregunta es una de las más habituales en nuestros días. Para algunas personas REST es una arquitectura , para otras es un patrón de diseño , para otras un API. ¿Que es REST exactamente? . REST o Representational State Transfer es un ESTILO de Arquitectura a la hora de realizar una comunicación entre cliente y servidor.



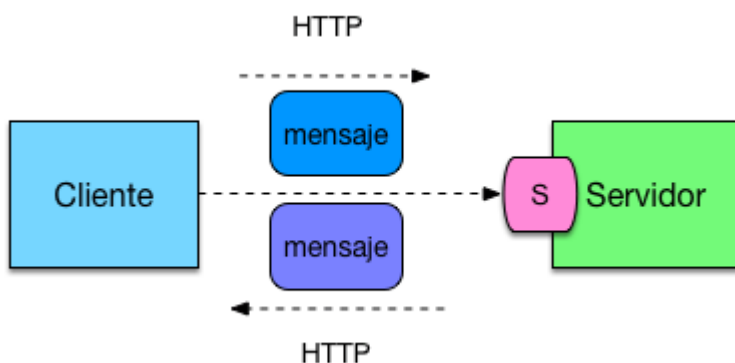
Vamos a intentar explicarlo esto paso a paso . Habitualmente cuando nosotros realizamos una comunicación cliente servidor accedemos al servidor en un punto de acceso , le enviamos una información y recibimos un resultado.



Ahora bien hay muchas formas de realizar esta operación .¿Cual es la más correcta? . Esa es una buena pregunta. Hoy por hoy una de las necesidades más claras es que esa comunicación sea abierta y podemos acceder desde cualquier sitio. Asi pues estamos hablando de una comunicación HTTP (Hyper Text Transfer Protocol) . Usamos el puerto 80 de nuestro servidor para permitir el acceso a la información desde cualquier lugar.



Una vez tenemos claro el protocolo de comunicación el siguiente paso es decidir que tipología de mensajes enviamos. Como punto de partida podemos mandar a un servicio un mensaje en formato XML o JSON. El servicio lo recepcionará y nos devolverá una respuesta. Hoy por hoy es mucho más habitual usar JSON que XML. ¿Porque? pues porque muchos de los clientes que se utilizan son capaces de procesar JavaScript de una forma muy nativa. JSON recordemos que es el acrónimo de JavaScript Object Notation.



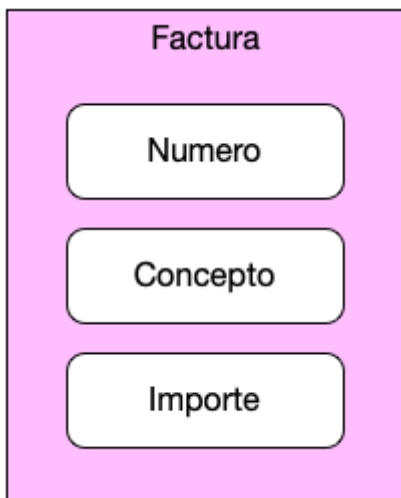
Esta situación es lo que habitualmente en Arquitecturas REST se denomina el nivel 0. No tenemos ningún tipo de organización es caótico pero accedemos a la información en formato de dato puro.

¿ Que es REST ? (Nivel 1 Recursos)

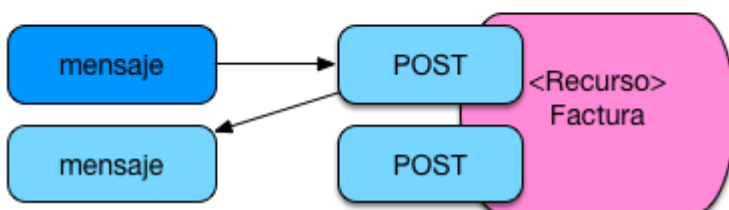
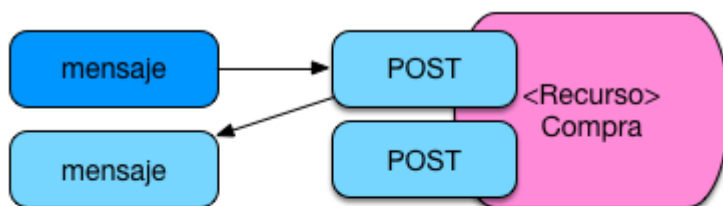
El siguiente paso es lo que se denomina nivel 1 , en vez de tener servicios con métodos diversos declaramos Recursos.

El concepto de Recurso REST

¿Qué es un Recurso? . Un recurso hace referencia a un concepto importante de nuestro negocio (Facturas ,Cursos , Compras etc). Es lo que habitualmente se denomina un objeto de negocio.



Este estilo permite un primer nivel de organización permitiendo acceder a cada uno de los recursos de forma independiente, favoreciendo la reutilización , aumentando la flexibilidad y abordando operaciones de inserción ,borrado , búsqueda etc. Estamos definiendo los conceptos que vamos a compartir entre Cliente y Servidor.



¿ Que es REST ? (Nivel 2 HTTP Verbs)

Hasta este momento para realizar las peticiones y enviar datos se usan GET o POST indistintamente . En el nivel 2 las operaciones pasan a ser categorizadas de forma más estricta. Dependiendo de cada tipo de operación se utilizará un método diferente de envío.

1. GET: Se usara para solicitar consultar a los recursos
2. POST: Se usará para insertar nuevos recursos
3. PUT : Se usará para actualizar recursos
4. DELETE : Se usará para borrar recursos

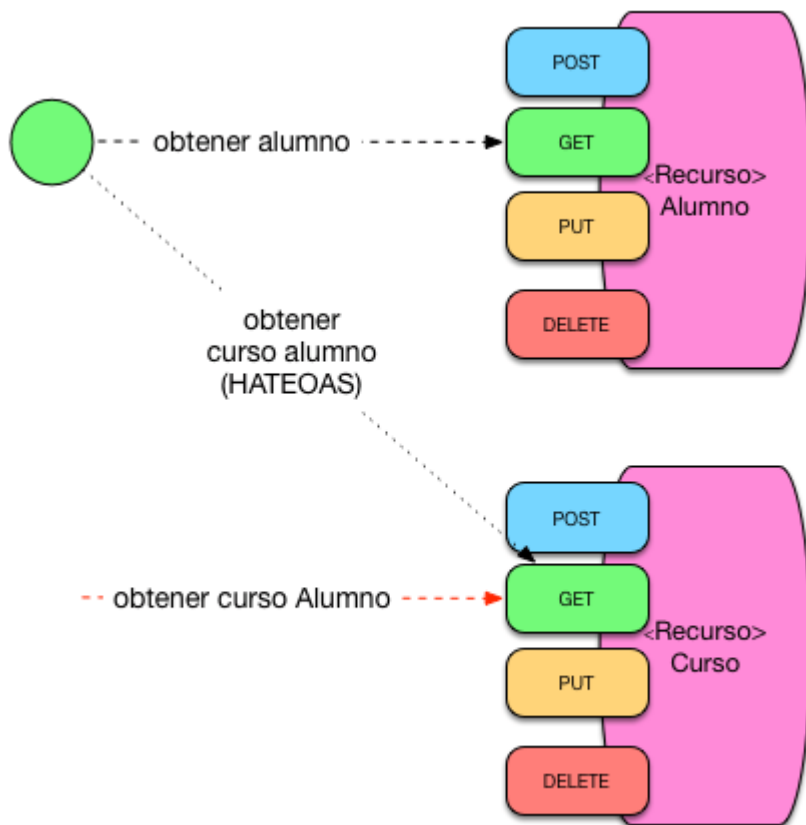


qu

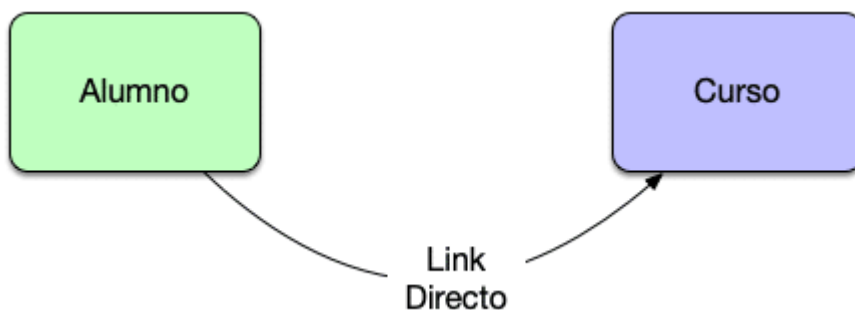
Este es el nivel en el que hoy en día se encuentran muchas de las Arquitecturas REST y que nos encontraremos en muchas aplicaciones.

HATEOAS (Nivel 3)

Todavía nos queda un nivel más que es el que se denomina HATEOAS (Hypertext As The Engine Of Application State). ¿Para qué sirve este nivel?. Supongamos que queremos acceder a un recurso Alumno via REST . Si tenemos una Arquitectura a nivel 2 primero accederemos a ese recurso utilizando GET. En segundo lugar deberemos acceder al recurso de Cursos para añadir al alumno al curso (línea roja) .



Esto implica que el cliente que accede a los servicios REST asume un acoplamiento muy alto, debe conocer la url del Alumno y la del Curso. Sin embargo si el recurso del Alumno contiene un link al recurso de Curso esto no hará falta y desde la url de Alumno directamente podremos acceder a la url del curso que pertenece a ese alumno.



Podríamos tener una estructura JSON como la siguiente:

```
{nombre:"pedro", apellidos:"gomez", cursos:"http://miurl/cursos"}
```

Podremos acceder directamente al recurso de Curso utilizando las propiedades del Alumno esto es HATEOAS. De esta forma se aumenta la flexibilidad y se reduce el acoplamiento. Construir arquitecturas sobre estilo REST no es sencillo y hay que ir paso a paso. Mi recomendación es que empecemos a abordarlas en un nivel 2 para más adelante si las necesidades surgen abordar un nivel 3 .

Otros artículos relacionados:

- [¿Que es el principio de HATEOAS?](#)
- [Arquitecturas REST y sus niveles](#)
- [Node.js presente y futuro](#)
- [REST JSON y Java](#)

**CURSO JPA
GRATIS
APUNTATE!!**