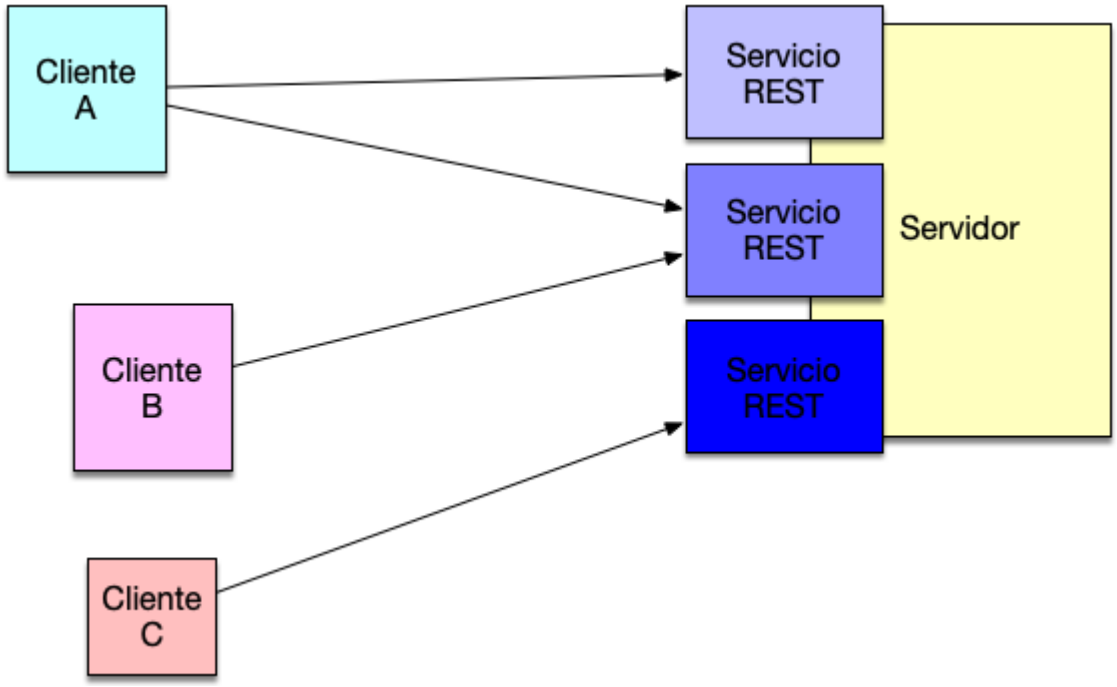
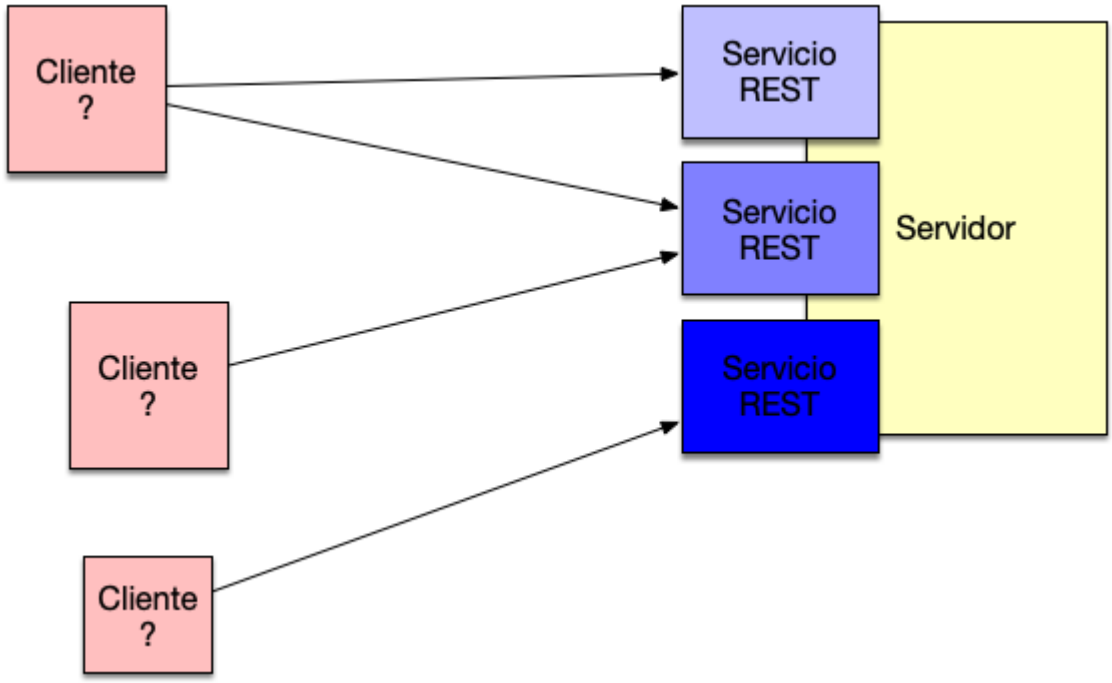
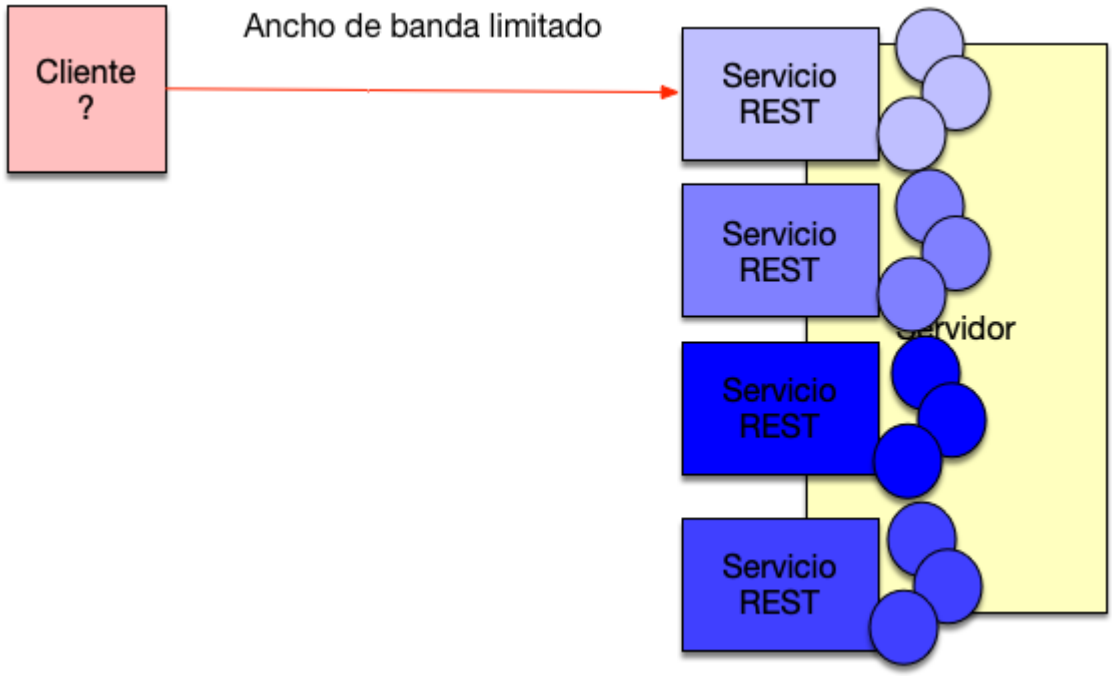
¿Qué es un API GateWay Pattern y cómo se usa este concepto? . Hoy en día estamos poco a poco migrando nuestras arquitecturas clásicas web a un tipo de Arquitecturas más modernas que se apoyan en servicios REST. La ventaja de los servicios REST es que podemos conectar a ellos cualquier tipo de cliente.



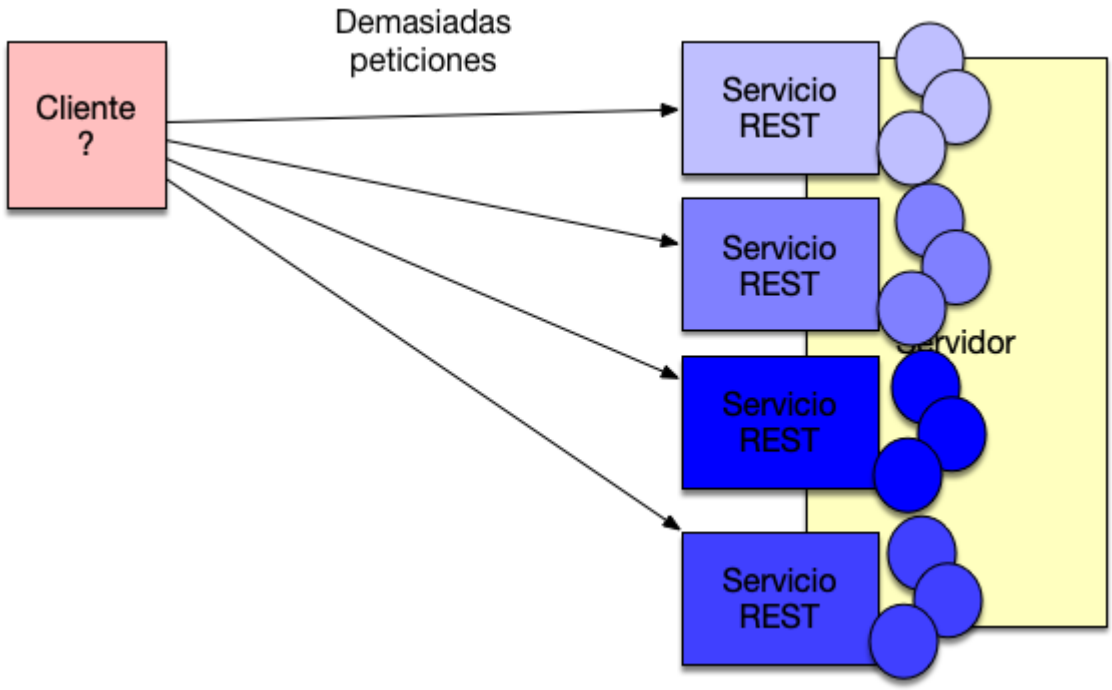
Esto tiene sus ventajas y también sus inconvenientes . La ventaja es clara , cualquier interface de usuario puede enlazarse a un servicio REST desde una tablet , pasando por un Móvil, una web clásica o incluso elementos ligados al mundo de la Domótica etc. Todo es conectable … esa es la gran ventaja . ¿Cuál es uno de sus problemas? uno de los problemas que los servicios REST tienen es que en muchos casos no sabemos quién se va a conectar a ellos .



Es decir somos completamente agnósticos del cliente , esto en un primer momento nos puede parecer que carece de importancia . pero la tiene.¿Qué sucede si hemos diseñado unos servicios REST muy genéricos? . Es decir todo el mundo se conecta a ellos, pero no hemos tenido en cuenta casuísticas especiales como que las personas se conecten a él desde una aplicación móvil que tiene un ancho de banda muy restringido.

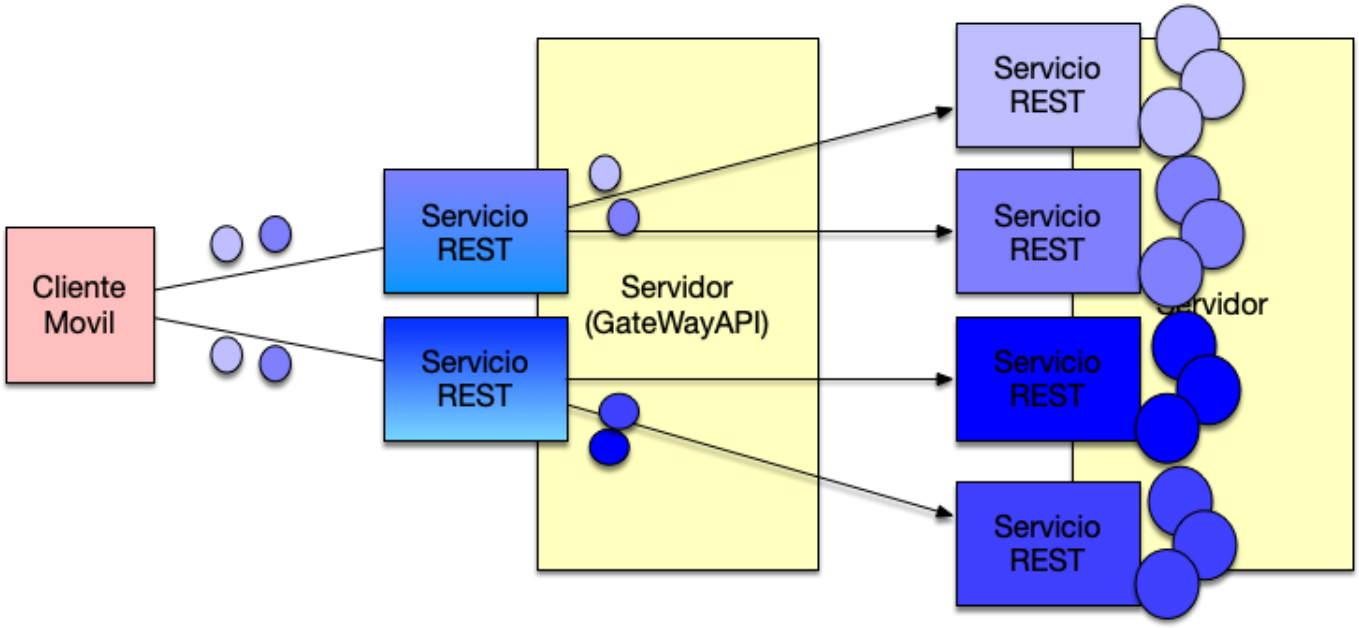


O que los datos que tiene que obtener este aplicación móvil no pertenecen a uno o dos servicios REST sino que es la combinación de muchos de ellos.



API GateWay Pattern

Estas casuísticas son muy comunes y nos encontramos ante situaciones en las los Servicios REST no dan el rendimiento que nos esperamos . Para ello dentro de las arquitecturas de MicroServicios existe el concepto de API GateWay Pattern o patrón de pasarela de API. Este patrón se encarga de definir un nuevo API a partir de las APIs existentes que enfoque de una manera mas directa una casuística determinada. Por ejemplo podemos diseñar un GateWay API para la plataforma móvil que publique los datos de forma mucho más reducida y en menos servicios.



De esta manera podemos mejorar el rendimiento a nivel de nuestros servicios REST enfocando las casuísticas adecuadas .

Otros artículos relacionados:

1. JSON API y Arquitecturas REST

2. Spring REST CORS y su configuración
3. Swagger documentando nuestro API REST
4. https://microservices.io/