Conocimientos previos:

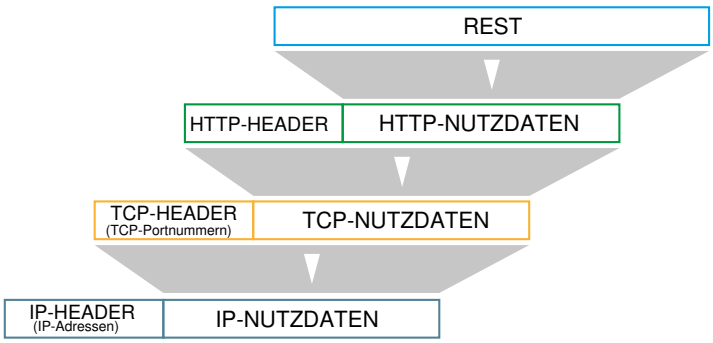
# REST - REpresentational State Transfer

**Comunicación industrial en base a peticiones HTTP estandarizadas**

REST no describe ninguna estructura de protocolo concreta. REST especifica más bien las propiedades que tiene que cumplir un intercambio de datos en un entorno industrial.

## REST - Transmisión a nivel de redes

Para empezar hay que señalar que REST utiliza HTTP o HTTPS como protocolo de orden superior.




La ventaja de HTTP(S) como base para la comunicación radica en que la mayoría de las redes pueden ser utilizadas sin excepción con HTTP(s), aunque estén protegidas por cortafuegos, etc. Además, HTTP(S) cumple las propiedades exigidas para REST.

Se utilizan los puertos TCP 80 o 443, como es habitual para HTTP o HTTPS.

## REST - Propiedades y elementos básicos

**Modelo cliente/servidor**
REST trabaja según el procedimiento de cliente/servidor y la acción siempre es iniciada por el cliente. El servidor pone los recursos a disposición (datos, contenidos, funciones). El cliente envía una petición (request) para acceder a los recursos seleccionados. Con su respuesta (reply) el servidor suministra los datos solicitados o confirma la acción deseada.

**Sin asignación de estado**
En muchas aplicaciones de cliente/servidor se asigna un estado determinado (autorización, tareas especiales, finalidad específica, ...) al establecerse la conexión. Y ese estado se mantiene durante toda la comunicación. En REST se trata de la misma manera a todos los envíos al servidor y solo el contenido del envío de datos determina el posterior tratamiento por parte del servidor o la clasificación de la petición. Por lo tanto, el estado de una aplicación está en manos del cliente.

**Sistema estratificado**
REST predispone una clara separación de competencias. Esa separación se aplica en particular en el desarrollo de la comunicación y en el tratamiento posterior de los contenidos transportados.

Eso permite utilizar proxies, gateways o estaciones intermedias similares en una vía de transmisión.

Además, se puede elegir HTTP o la transmisión codificada vía HTTPS para la comunicación, según el grado de seguridad exigido. Para el desarrollo funcional de los contenidos de REST no hay ninguna diferencia.

**Direccionabilidad de los recursos**
Todos los recursos disponibles en un servidor están accesibles a través de una dirección unívoca (URI = Uniform Resource Identifier). La estructura de URI es similar al URL utilizado para las direcciones en el navegador:

`Protokoll://<Host>:<Port>/<Pfad>/<Ressource>?<Parameter>&<Parameter>`

**Métodos de petición**
Para las peticiones se dispone de los accesos HTTP estandarizados:

- **GET**
  solicita los recursos del servidor y se utiliza exclusivamente como lectura
- **POST**
  crea nuevos recursos o modifica existentes
- **PUT**
  modifica recursos existentes
- **DELETE**
  borra recursos existentes
- **HEAD**
  solicita únicamente la cabecera HTTP para, por ejemplo, verificar la disponibilidad de los recursos a pedir

- **OPTIONS**
  abre la información relativa a las opciones de comunicación

Los datos abiertos reciben el nombre de representaciones de uno o varios recursos. De ahí el nombre de REST para REpresentational State Transfer.

En definitiva, las representaciones no son otra cosa que una reproducción de partes de los datos del proceso. La transmisión puede tener lugar casi en cualquier formato, pero este tiene que estar acordado. Los más habituales son JSON, XML o texto en bruto. Aunque según la aplicación también se emplean SVG, MP3 u otros formatos. La representación puede contener también hipervínculos a otros recursos.

El cliente envía:

http://192.168.1.19/rest/json/iostate/

Y el Web-Termo-higrobarómetro envía como respuesta:


```
{
  "iostate": {
    "sensor": [
      {
        "name": "Temperatur",
        "number": 0,
        "unit": "°C",
        "value": 25.9
      },
      {
        "name": "rel. Feuchte",
        "number": 1,
        "unit": "%",
        "value": 43.2
      },
      {
        "name": "Luftdruck",
        "number": 2,
        "unit": "hPa",
        "value": 994.7
      }
    ]
  }
}
```


**Code on Demand**
Hoy en día en las páginas web es habitual cargar simultáneamente o más tarde, por ejemplo, JavaScript. REST también permite cargar más tarde códigos fuente, partes de programas o elementos de funciones. De ese modo se puede ampliar o modificar la función del cliente durante la ejecución.

**Cache**
REST predispone la respuesta a peticiones repetidas del cliente desde un cache (memoria intermedia) con el fin de reducir la carga de datos en las vías de transmisión. A través de los datos correspondientes, el servidor establece en la cabecera HTTP si se permite utilizar el cache para la representación solicitada o no.

**Ventajas de REST**
Fácil implementación, pues utiliza en gran medida los mecanismos de HTTP.

**Desventajas de REST**
La técnica de petición y respuesta solo permite el pollen, es decir la solicitud dirigida de datos, pero no una comunicación basada en eventos. Es necesario solicitar (pollen) los datos relevantes de forma continuada para detectar cambios.

## MQTT - Message Queue Telemetry Protocol

Intercambio de datos sin conexión directa

## OPC UA - OPC Unified Architecture

OPC Unterstützung out of the box

## OPC DA - OPC Data Access

El intérprete de los datos de procesos

**Productos para aplicaciones industriales con protocolos estándar**

Web-IO 4.0 Digital 2xIn, 2xOut

Si es necesario, alimentación también por PoE

Web-IO 4.0 Digital 12xIn, 12xOut

12 salidas (6-30V), 12 entradas (8-30V)

Otros Web-IO

Todos los Web-IO Digital 24V de W&T

W&T
www.WuT.de

Le atendemos personalmente:

Wiesemann & Theis
GmbH
Porschestr. 12
42279 Wuppertal
Tel: +49 202/2680-110 (lu-vi de 8-17 horas)
Fax: +49-202/2680-265
info@wut.de



Protección de datos