CoAP

Protocolo de Capa de Aplicación para Entornos Restringidos

Entornos Restringidos

Dispositivo:

- MicroProcesadores de 8 bits
- Capacidad de Memoria Limitada
- Consumo de Energía Limitado

Comunicación:

- Transmisión monetariamente costosa por unidades de bit
- Ancho de banda Limitado

Capas TCP/IP - IETF

Capas de Comunicación:

Acceso a la red:

- Define la transmisión de una estructura de bits sobre una red
- Intercambia los datos entre una computadora y la red física
- Entrega datos entre dos dispositivos en la misma red

Interconexión de Redes:

- Responsable de enrutar mensajes a través de redes intercontectadas usando dispositivos como gateways y routers

Transporte Dispositivo a Dispositivo:

- Responsable de proveer integridad de datos a través de un servicio de comunicación de extremo a extremo altamente confiable

Capa de Aplicación:

- Lidia con Software que usa la red
- Ofrece servicios a usuarios para comunicar sobre una red

Arquitectura de Software

- Definida por Elementos: **Componentes**, **Conectores** y **Datos**. Relacionados (lógica) mediante condiciones (forma) para lograr Propiedades.

  - **Componente**: Unidad de instrucciones de software que proveen una transformación de datos mediante su interfaz
  - **Conector**: Mecanismo que media la comunicación, coordinación o cooperación entre Componentes
  - **Dato**: Elemento de información enviada/recibida por Componentes mediante un Conector

Propiedades de Arquitectura

- Desempeño
  - De Red rendimiento, sobrecarga, ancho de banda, Percepción del Usuario latencia, Eficiencia de Red
- Escalabilidad soportar muchos componentes
- Simplicidad cohesión
- Adaptabilidad
  - Evolución implementación, Extensión funcionalidad, Personalización especialización de comportamiento temporalmente, Configuración, Reutilización
- Visibilidad interfaz, intermediarios entre componentes
- Portabilidad estándares
- Confiabilidad soporte a fallos

Sistemas monolíticos

Arquitectura Orientada a Servicios

Arquitectura basada en MicroServicios

Representational State Transfer (REST)

Arquitectura basada en los estilos que condicionan la WWW:

- Cliente-Servidor
  - Escalabilidad, Simplicidad (separación de responsabilidades)
- *Stateless*
  - Visibilidad, Confiabilidad, Escalabilidad
- *Cache*
  - *Performance*
- Interfaz Uniforme
  - Visibilidad
- Sistema en Capas
  - Evolución, Reutilización
- Código a Demanda
  - Extensión, Configuración, Performance de Usuario

Datos de REST

Datos	Ejemplos Web
Recurso	El objetivo conceptual esperado de una referencia de hipertexto
Identificador del recurso	URL, URN
Representación	Documento HTML, imagen JPEG
Metadatos de la representación	*media type*, *last-modified time*
Metadatos del recurso	*source link*, *alternates*, *vary*
Datos de control	*if-modified-since*, *cache-control*

Conectores REST

Conector	Ejemplos Web
Cliente	libwww, libwww-perl
Servidor	libwww, Apache API, NSAPI
Cache	caché del navegador, red de cache Akamai
Resolución	bind (DNS)
Túnel	SOCKS, SSL después de HTTP CONNECT

Componentes REST

Componente	Ejemplo Web
Servidor de Origen	Apache httpd, Microsoft IIS
Gateway	Squid, CGI, Proxy Inverso
Proxy	CERN Proxy, Netscape Proxy, Gauntlet
Agente de usuario	Netscape Navigator, Lynx, MOMspider

Vista de Proceso de una Arquitectura REST

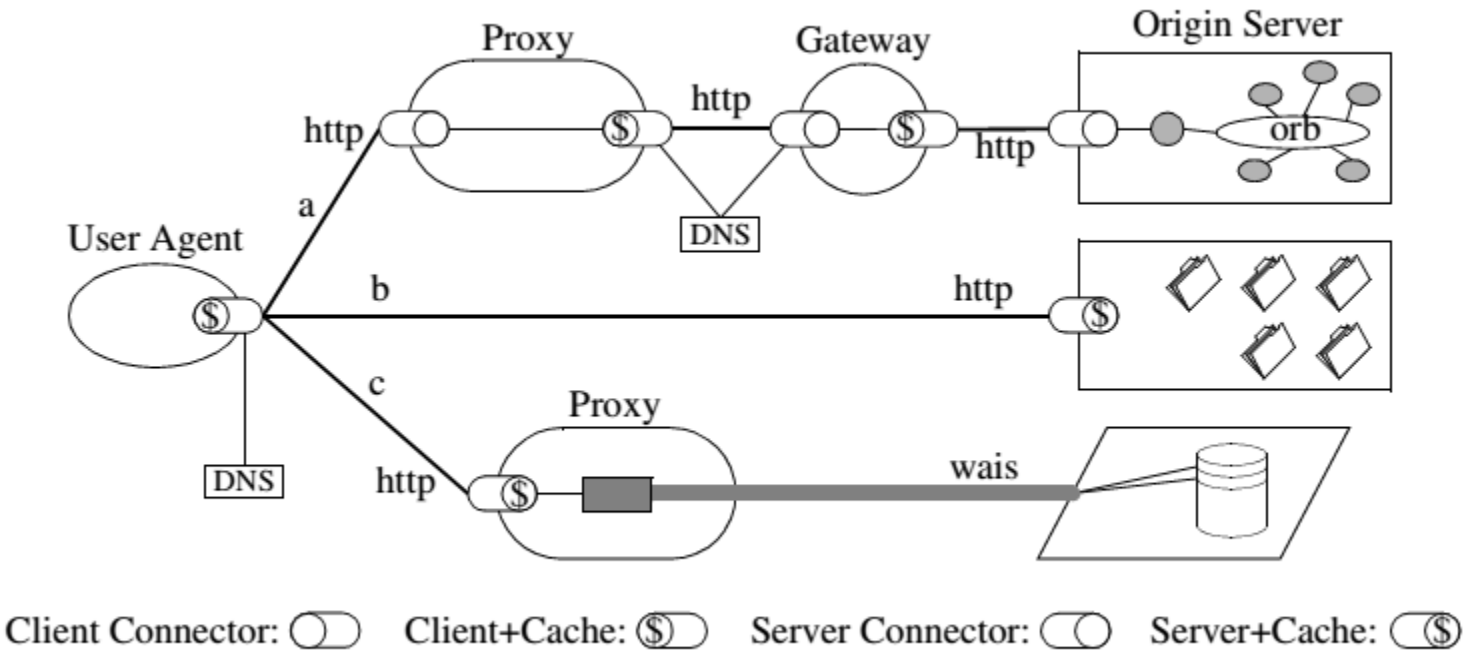


Figure 5-10. Process View of a REST-based Architecture

REST aplicado a URI Identificadores de Recurso Uniforme

- Recurso: la semántica que lo identifica no el valor en ese momento
- Concepto: puede ser representado de múltiples formas
- Implementación remota: el servidor se encarga del mapeo del recurso

REST aplicado a HTTP

- Versión HTTP/1.1
- Metodos
  - GET, HEAD, POST, PUT, DELETE, CONNECT, OPTIONS, TRACE
- Respuestas de Códigos de Estado
  - 100-199 Información
  - 200-299 Éxito
  - 300-399 Redirección a otro recurso
  - 400-499 Error del cliente que no debe reintentar
  - 500-599 Error del servidor, el cliente puede reintentar luego
- Cabeceras
  - Host
  - Transfer Encoding
  - Content Length / Chunked
  - Cache control / Age / Etag / Vary – no-cache

Ejemplo de Solicitud HTTP

GET /html/rfc7230 HTTP/1.1

Host: tools.ietf.org

Cache-Control: no-cache

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/50.0.2661.102 Safari/537.36

Referer: https://tools.ietf.org/html/rfc2616

Accept-Encoding: gzip, deflate, sdch, br

Accept-Language: es,en-US;q=0.8,en;q=0.6

Ejemplo de Respuesta HTTP

HTTP/1.1 200 OK

Date: Wed, 08 Jun 2016 15:38:19 GMT

Server: Apache/2.2.22 (Debian)

Content-Location: rfc7230.html

Vary: negotiate,Accept-Encoding

TCN: choice

Last-Modified: Sun, 05 Jun 2016 07:23:56 GMT

ETag: "3caa95-418c0-53482d6b66324;534c614df22bc"

Accept-Ranges: bytes

Cache-Control: max-age=604800

Expires: Wed, 15 Jun 2016 15:38:19 GMT

Content-Encoding: gzip

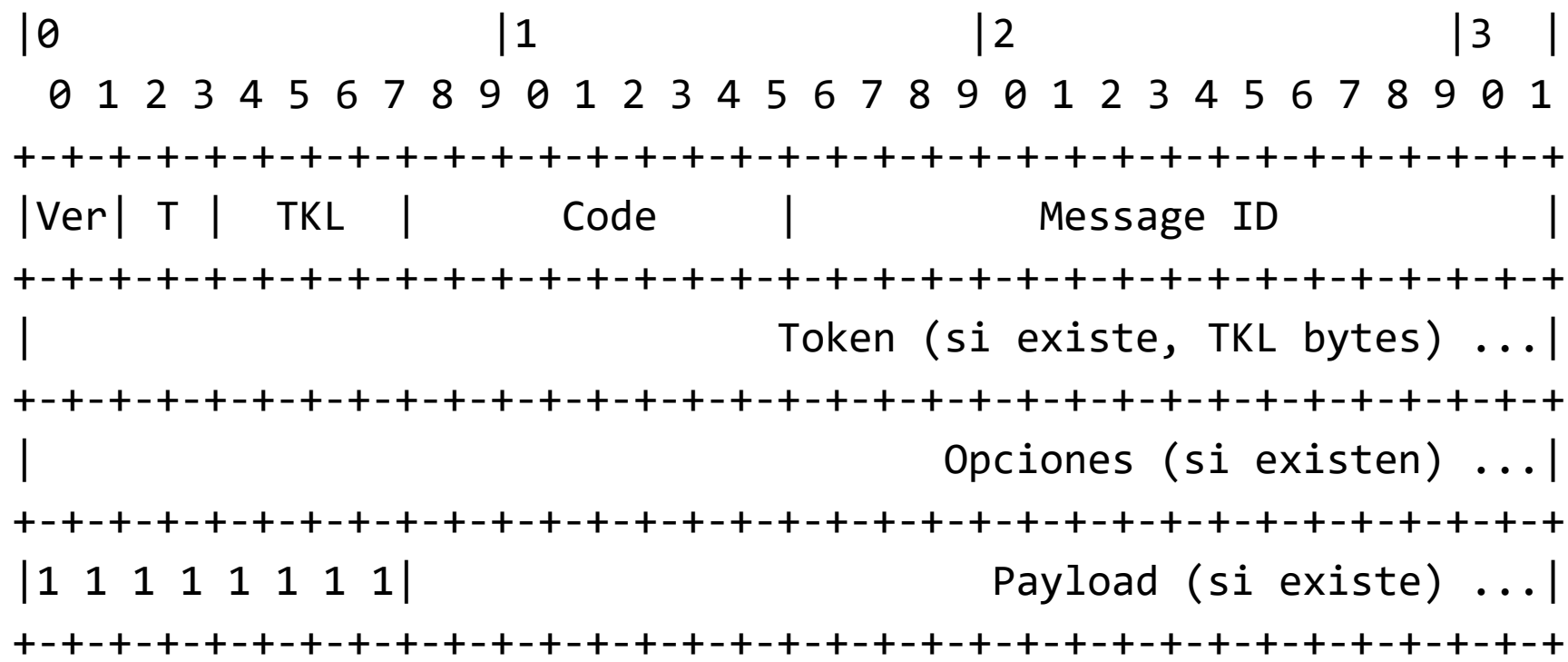
Keep-Alive: timeout=5, max=100

Connection: Keep-Alive

Transfer-Encoding: chunked

Content-Type: text/html; charset=UTF-8

Formato del Mensaje CoAP



Protocolo CoAP

- **Encabezado:**
  - Versión: (2 bits)
  - Tipo: (2 bits) [0: CONfirmable, 1: NON confirmable, 2: ACKnowledgement, 3: ReSeT]
  - Longitud Token: (4 bits) (*LTK)
  - Código: (3 Bits) [0: Req, 1: Informativo, 2: Éxito, 3: Redirección, 4: Error Req, 5: Error Servidor)
  - Detalle: (5 Bits) [0.01: GET, 0.02: POST, 0.03: PUT, 0.04: DELETE]
    [2.01: Created, 2.02: Deleted, 2.03: Valid, 2.04: Changed, 2.05: Content]
    [4.00: Bad Request, 4.01: Unauthorized, 4.02: Bad Option, 4.03: Forbidden, 4.04: Not Found,
    4.05: Method Not Allowed, 4.06: Not Acceptable, 4.12: Precondition Failed,
    4.13: Request Entity Too Large, 4.15: Unsupported Content-Format]
    [5.00: Internal Server Error, 5.01: Not Implemented, 5.02: Bad Gateway, 5.03: Service Unavailable,
    5.04: Gateway Timeout, 5.05: Proxying Not Supported]
  - MessageID: (2 Bytes)
  - Token: (*LTK Bytes)
- **Opciones**:
  - Tipo: 4 bits (delta respecto al anterior)
    [1: If-Match, 3: Uri-Host 4: ETag, 5: If-None-Match,
    6: Observe [0: register, 1: desregister],
    7: Uri-Port, 8: Location-Path, 11: Uri-Path,
    12: Content-Format: [0: text/plain; charset=utf-8, 40: application/link-format, 41: application/xml,
    42: application/octet-stream, 47: application/exi, 50: application/json],
    14: Max-Age, 15: Uri-Query, 17: Accept, 20: Location-Query, 35: Proxy-Uri, 39: Proxy-Scheme, 60: Size1]
  - Longitud: 4 bits (*L)
  - Valor: (*L Bytes)
- **Cuerpo**:
  La longitud está determinado por el tamaño del datagrama.