Desarrollo Backend con Node.js, Express y BBDDs

Juan Quemada, DIT - UPM

Índice

1. Introducción al Curso 3
2. Internet y la plataforma Web 7
3. La plataforma Web actual: los nuevos clientes y servidores 12

Introducción al Curso

Juan Quemada, DIT - UPM

Desarrollo Backend con Node.js, Express y BBDDs

- Curso de desarrollo de aplicaciones de servidor
  - Utilizando JavaScript y node.js
- Incluye
  - Node.js, y npm
    - Repasamos últimas mejoras de JavaScript (desde JS6 a JS9)
  - Gestión de bases de datos desde node.js
  - Sockets
  - HTTP, express y MVC
  - WebSockets
  - Testing
- Utilizamos técnicas de ingeniería software
  - Gestión de versiones, de paquetes y testing

Equipos, herramientas y servicios

- Un PC o portatil de trabajo (necesario)
  - Con S.O. Linux/UNIX (incluyendo MAC) o Windows
- Móvil o tableta
  - Es conveniente para probar, pero no necesario
- Navegador: Chrome, Firefox, Safari, …
- IDE: Visual Studio Code
  - Es un entorno de desarrollo gratuito y muy potente
    - https://code.visualstudio.com
- Cuenta en Github: https://github.com/ (gratuito)
- Cuenta en Glitch: https://glitch.com/ (gratuito)

Desarrollo Web Fullstack con JavaScript y Node.js

Programa Oficial de UPM: Título Propio*

Acceso: https://miriadax.net/web/fullstack

Consta de 4 MOOCs y 4 examenes

Desarrollo Frontend con HTML, CSS y Javascript

Acceso: https://miriadax.net/web/html5mooc 

Gestión de proyectos Software con Git y GitHub

Acceso: https://miriadax.net/web/gitmooc

Desarrollo Backend con Node.js, Express y BBDDs

Acceso: https://miriadax.net/web/nodemooc 

Desarrollo de un Proyecto Fullstack con JavaScript

Acceso: https://miriadax.net/web/quiznodemooc

Los 4 MOOCs pueden cursarse en abierto sin matricularse* en Título

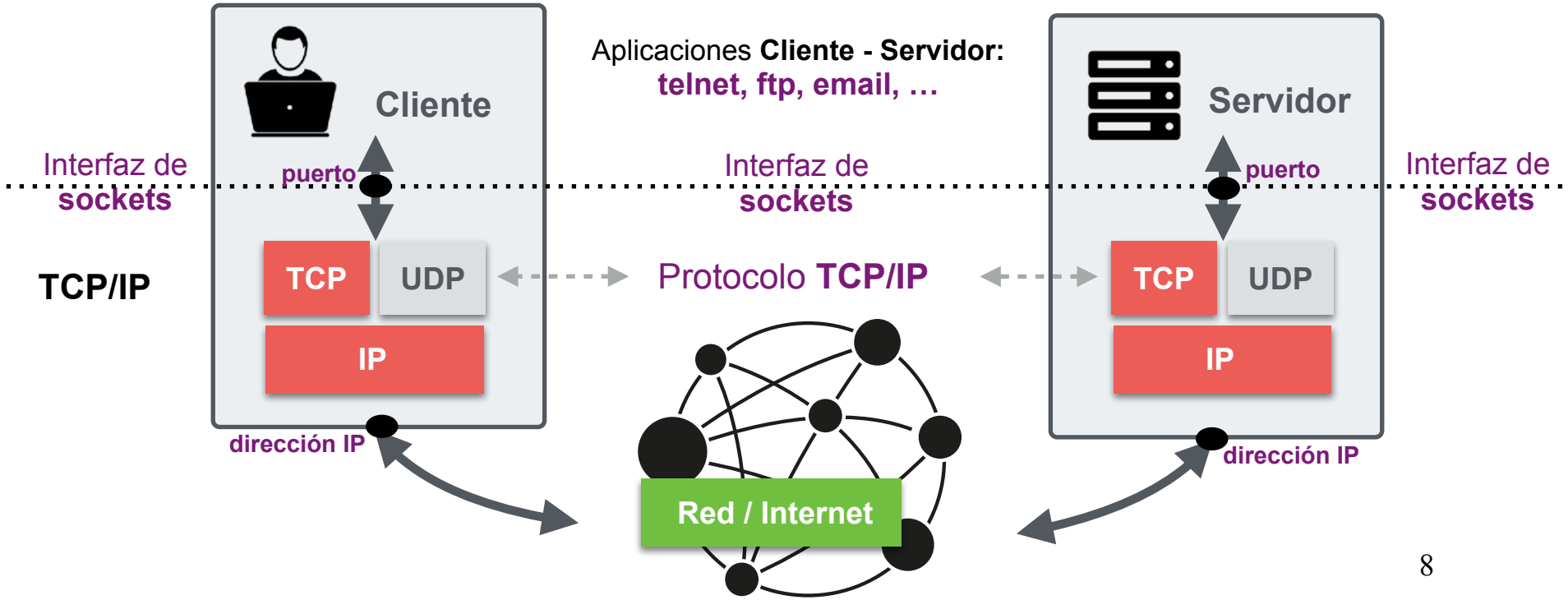
*La matricula del programa debe realizarse antes del examen del primer MOOC, 2 semanas antes del final.
También pueden obtenerse títulos UPM de los MOOCs individuales (ver: https://miriadax.net/web/fullstack).

Internet y la plataforma Web

Juan Quemada, DIT - UPM
Santiago Pavón, DIT - UPM

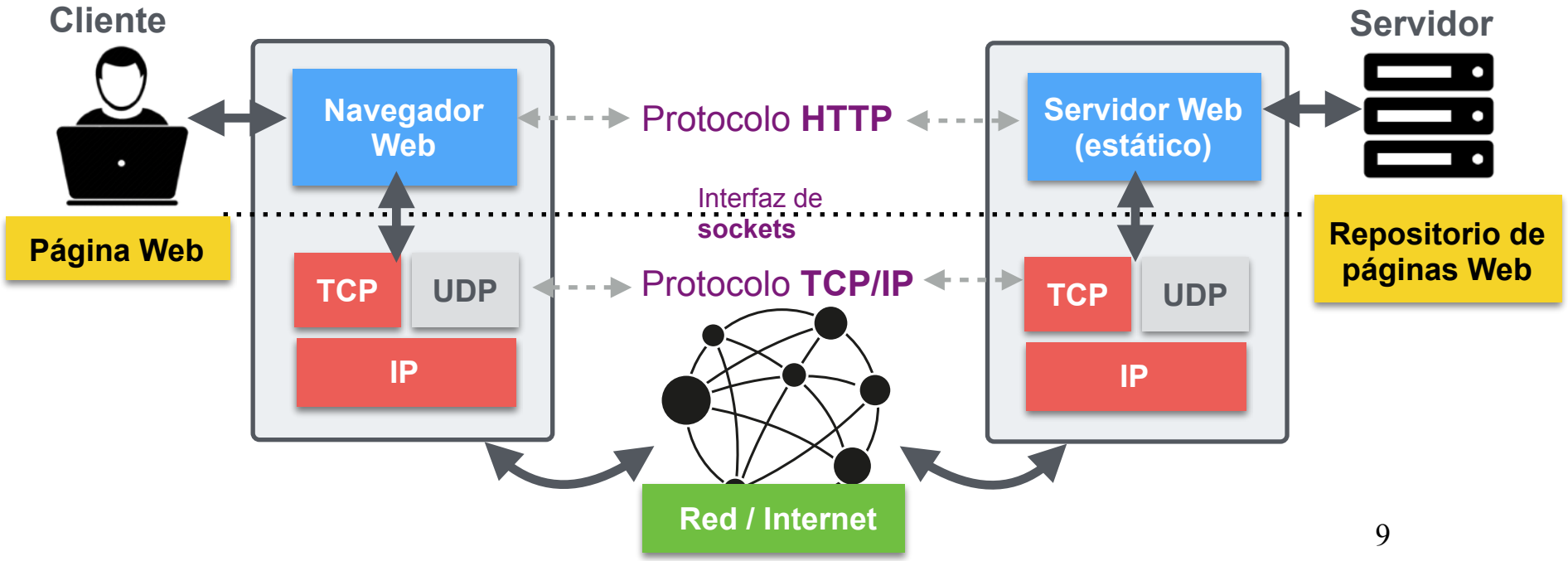
Internet y la arquitectura TCP/IP

- Internet empieza a operar en Arpanet el 1 de Enero 1983
  - Internet conecta ordenadores a Internet con la pila de protocolos TCP/IP
    - TCP/IP soporta **aplicaciones cliente-servidor** con el **interfaz de sockets**
      - La dirección IP identifica el ordenador en Internet y el puerto identifica la aplicación dentro del ordenador
- Las primeras aplicaciones cliente servidor de Internet son
  - telnet (terminal virtual), ftp (transferencia de ficheros), email (correo elec.), ..

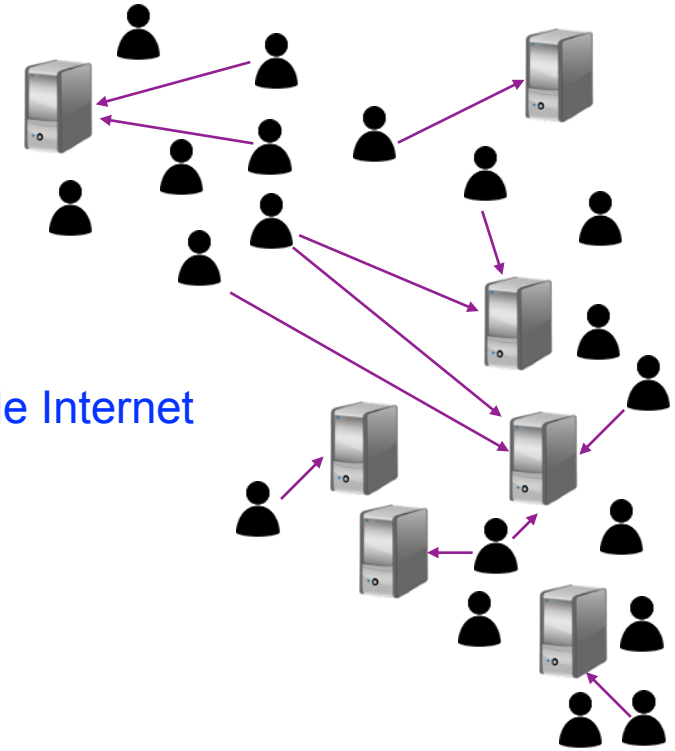


La Web

- Tim Berners Lee propone en 1989 una nueva aplicación: la Web
  - Servicio de publicación de documentos hipertexto en Internet
    - Aplicación cliente (**navegador**) <-> servidor (**servidor Web estático**)
- La Web es el almacén de contenidos que necesitaba la red
  - Transforma Internet en una **"Red de distribución de contenidos"**
    - Crece continuamente -> es **descentralizada y escalable**

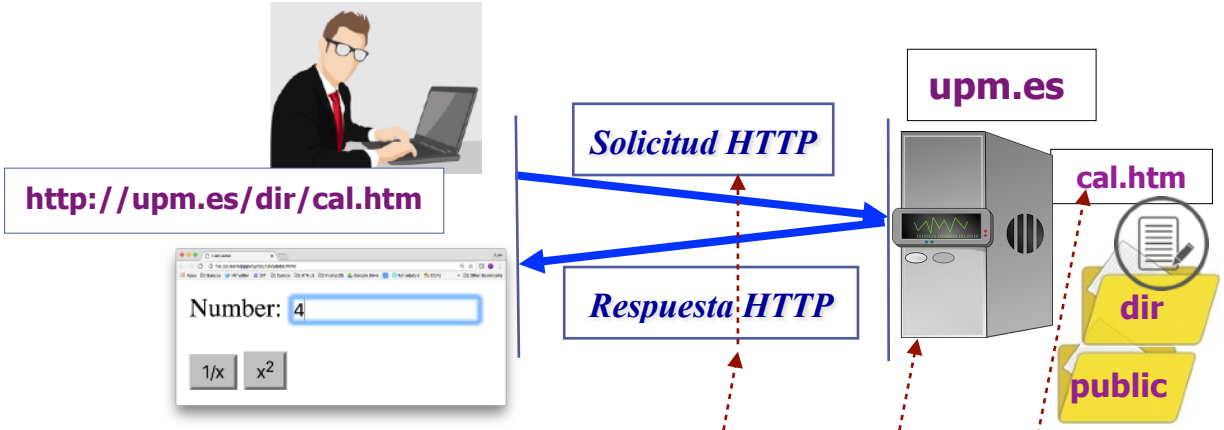


La Web inicial



- **URL**
  - **Dirección única** a un fichero (o sección) en un servidor de Internet
    - Ejemplo: https://en.wikipedia.org/wiki/URL
- **HTTP**
  - **Protocolo** para traer **ficheros** de un **servidor remoto**
    - Protocolo simple y *¡muy escalable!*
  - El fichero se identifica con un **URL**
- **HTML**
  - **Lenguaje** para definir **páginas Web** (con **hiperenlaces)** para visualizar en el navegador
- **Cliente Web (navegador)**
  - Programa para visualizar páginas Web (HTML) traídas de un servidor con HTTP
- **Servidor Web estático**
  - Programa que sirve páginas Web (ficheros HTML) a los clientes que las solicitan

La Web inicial



- Servidor Web estático
  - Programa que sirve ficheros solicitados por clientes
    - Los ficheros están en un directorio de recursos (páginas Web)
      - El directorio de recursos suele ser: www, public, ..
- Cliente Web
  - Presenta páginas Web traídas de un servidor en Internet
  - El URL identifica el recurso Web: **http://upm.es/dir/cal.htm**
    - **http**: El **protocolo** de acceso al servídor (HTTP GET)
    - **upm.es:** La dirección de dominio del **servidor** que alberga la página
    - **/dir/cal.html :** La ruta al **fichero** (página Web) en el directorio de recursos del servídor
- La transacción HTTP vista desde el cliente:
  - Establece una conexión TCP con el servidor (**upm.es**)
  - Envía por la conexión una **Solicitud** HTTP con la ruta al recurso Web (**/dir/pagina.htm)**
  - Recibe por la conexión la **Respuesta** HTTP con el fichero (**página Web**)
  - El servidor cierra la conexión TCP

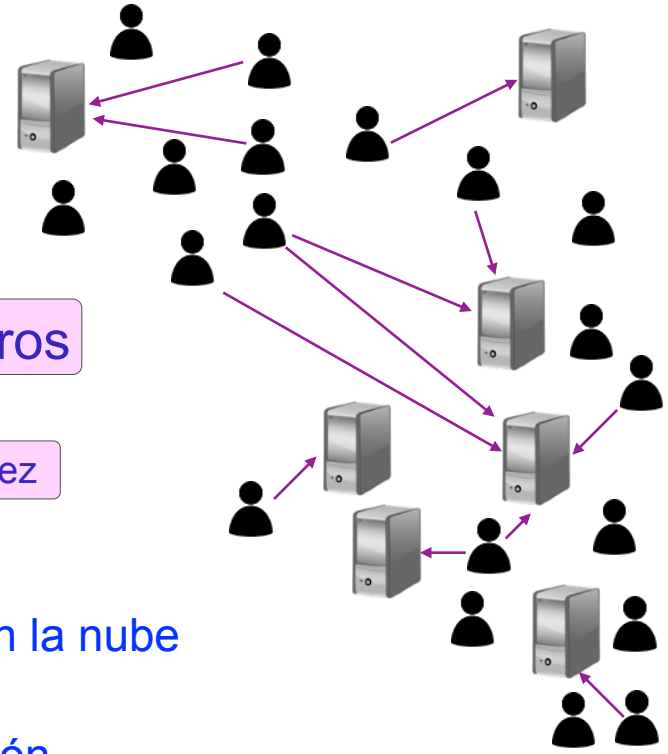
La plataforma Web actual:
los nuevos clientes y servidores

Juan Quemada, DIT - UPM
Santiago Pavón, DIT - UPM

Computación distribuida y la plataforma Web

- Paradigma de **computación distribuida**
  - Partes de un programa cooperan en un objetivo común conectados por Internet
    - Plantea múltiples retos relacionados con la concurrencia entre procesos y la comunicación entre ellos, transacciones seguras, sincronización de relojes, tolerancia a fallos de las partes, etc.
  - Se han propuesto diversas plataformas: Web, CORBA, Fractal, JavaBeans, NFS, AFS, ..
    - La **plataforma Web** es el entorno más utilizado para el desarrollo de servicios en Internet
- La **plataforma Web**
  - Arquitectura descentralizada basada en el modelo cliente <-> servidor para
    - Aplicaciones de sobremesa, teléfonos móviles u otros dispositivos
    - Servicios en la nube
    - Intranets y aplicaciones corporativas
    - Aplicaciones P2P (Pier to Pier)
    - etc.
- Este **curso** describe
  - Los componentes más importantes de la plataforma Web
  - Las técnicas de desarrollo de aplicaciones
  - El lenguaje JavaScript para programación de aplicaciones

La plataforma Web actual



- **URL** -> Se añade la **query** para envío de parámetros
  - **Transacción** con parámetros para acceder a servicios
    - Por ejemplo: https://upm.es/registro?nombre=José&apellido=Perez
- **HTTP** -> **HTTP/2, WebSockets, WebRTC, ...**
  - Se añaden nuevos **protocolos** para crear aplicaciones en la nube
    - Protocolos *muy escalables*
  - Los nuevos protocolos soportan cualquier tipo de aplicación
- **HTML** -> Aplicaciones Web en **HTML**, **CSS** y **JavaScript**
  - **Aplicaciones Web** de cliente (con hiperenlaces) que se ejecutan en el navegador
- **Cliente Web (navegador)** -> Aparecen los **móviles** con sus **apps**
  - Los **clientes** web se hacen **programables**
- **Servidor Web estático** -> **Servidor Web dinámico** (programable)
  - Los **servidores** se hacen **programables** y se conectan a BBDDs

URL y URIs

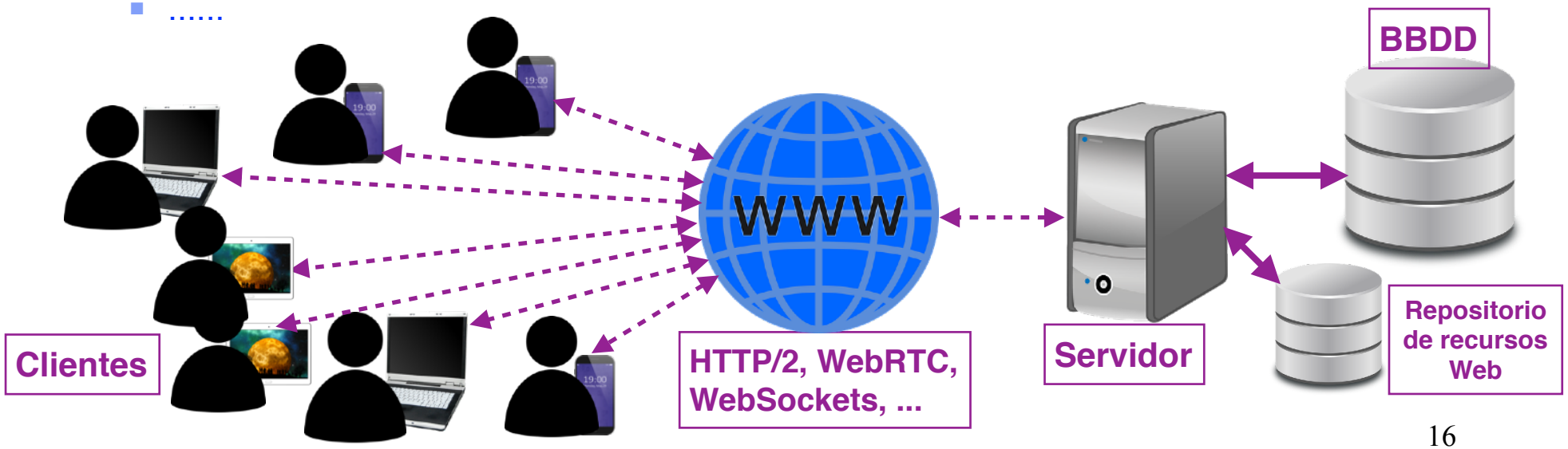
- **URL** (Uniform Resource Locator)
  - **Dirección** de acceso a cualquier **recurso** o **servicio** de Internet
    - Los **URLs** (RFC1738) son un caso particular de los **URIs** (Uniform Resource Identifiers, RFC3986)
      - https://www.ietf.org/rfc/rfc1738.txt y https://tools.ietf.org/html/rfc3986

scheme://user:password@host:port/path?query#fragment

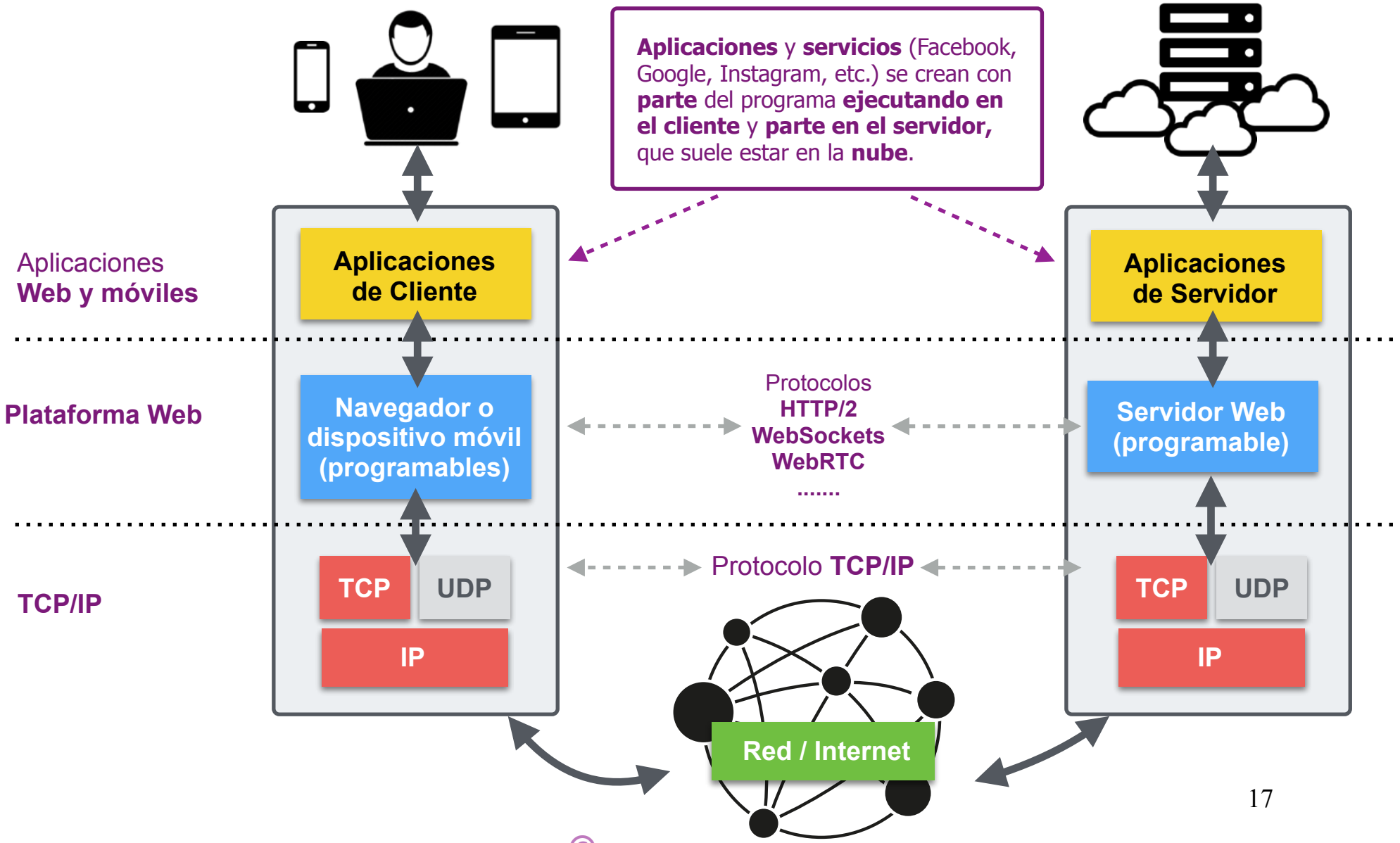
- **http://upm.es/dir/pagina.html**
  - URL Web que identifica e la página Web **/dir/pagina.html** en el servidor **upm.es**
- **http://upm.es:8080/dir/pagina.html**
  - URL Web similar a la anterior, donde el servidor escucha en el **puerto 8080** y no en el 80 asignado a Web
- **http://upm.es/dir/pagina.html#p3**
  - URL igual al anterior pero con **fragment** o **anchor** (ancla), que identifica el elemento con **id='p3'** en **pagina.html**
- **http://felix@upm.es/dir/pagina.html**
  - URL Web de un recurso asociado al usuario **felix** en su cuenta en el servidor **upm.es**
    - Se recomienda enviar passwords en URLs solo con HTTPS y no con HTTP, porque es inseguro
- **http://upm.es/registro?id=23&nombre=José**
  - URL que envía dos parámetros en la query (parámetros **id** y **nombre)**
- **mailto:felix@upm.es**
  - URL de email que identifica el buzón del usuario **felix** en el servidor **upm.es**

Arquitectura de 3 capas

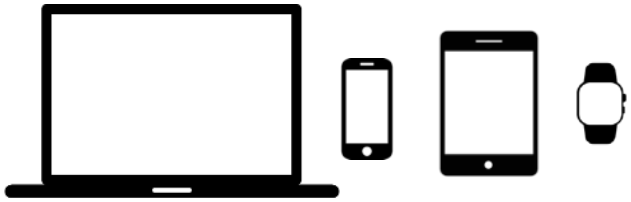
- Los servicios y aplicaciones de Internet suelen tener estás 3 capas
  - **Cliente**: Capa de visualización y presentación con el interfaz del servicio
  - **Servidor**: Capa lógica de la aplicación con las reglas de atención de peticiones
  - **BBDD**: Capa de persistencia que almacena los datos en una base de datos
- Cliente y servidor se comunican a través de múltiples protocolos:
  - HTTP/2: Versión 2 (actual) de HTTP es mas eficiente y con menos latencia
  - Web Sockets: Para aplicaciones interactivas entre clientes
  - WebRTC: Para aplicaciones de voz y video sobre IP
  - Server_send events: Para envío de eventos del servidor al cliente
  - .....



Arquitectura de la Plataforma Web



El cliente y sus aplicaciones



- **Dispositivos** cliente de acceso a Internet
  - PCs, portátiles, tabletas, teléfonos y relojes inteligentes, etc.
- **Cliente: programa** que accede a servicios en Internet
  - El **navegador (browser)** es el principal cliente de acceso desde un PC
  - Las **apps** de los dispositivos móviles son hoy los clientes mas utilizados
- **Navegadores:** Apps se programan en **HTML, CSS y JavaScript**
  - Chrome, Firefox, Internet Explorer, Opera, Safari, ...
- **Aplicaciones nativas (apps):** Android, iOS-Apple, etc.
  - Se programan en entornos de desarrollo con lenguajes específicos
    - **Android** se programa en **Java**, **IOS** en **Swift**, etc
  - Se programan en **JavaScript** en entornos para aplicaciones **nativas**, por ejemplo
    - **React Native**, Apache-Cordova/PhoneGap, ... (reutilizan el código del navegador)

El servidor y sus aplicaciones

- **Servidor***
  - **Programa** proveedor de servicios a los clientes
    - Se conecta a un **puerto** de la **máquina servidora**, el servidor **Web** usa el **puerto 80** por defecto
      - *La **máquina servidora** se denomina también servidor, pero produce ambigüedad
- El programa **servidor** se ejecuta en una **máquina servidora**
  - Una máquina servidora tiene una dirección "**conocida**" en Internet
    - La dirección esta incluida en el URL de acceso: https://**en.wikipedia.org**/wiki/URL
      - Dirección de la máquina servidora: **en.wikipedia.org**
  - La máquinas servidoras pueden ser máquinas físicas o máquinas virtuales en la nube
- Servidores Web más usados: Apache, Nginx, Microsoft-IIS, etc.
  - Los servidores Web integran aplicaciones en múltiples lenguajes de programación
    - node.js + **JavaScript**
    - Ruby on Rails
    - Django + Python
    - Spring MVC + Java
    - Zend + PHP
    - etc

Final