MEDAC
Instituto Oficial de Formación Profesional

DESPLIEGUE DE APLICACIONES WEB
TÉCNICO EN DESARROLLO DE APLICACIONES WEB

Arquitecturas web

01

ÍNDICE

 ISBN: 978-84-18983-26-9

OBJETIVOS

Conocer el concepto de arquitectura web.

Analizar los diferentes modelos y sus características.

Valorar puntos a favor y en contra de cada modelo según la finalidad.

Seleccionar el modelo de arquitectura que más se adapte a las necesidades de nuestro desarrollo.

/ 1. Introducción y contextualización práctica

En este tema conoceremos las diferentes estructuras y modelos que podemos utilizar para compartir información utilizando la WWW (World Wide Web), y hablaremos de la historia de esta red informática mundial. Además, analizaremos los componentes que hacen posible que a día de hoy navegar por internet sea tan fácil como pinchar en un enlace.

Planteamiento del caso práctico inicial

A continuación, vamos a plantear un caso práctico a través del cual podremos aproximarnos de forma práctica a la teoría de este tema.

Escucha el siguiente audio donde planteamos la contextualización práctica de este tema, encontrarás su resolución en el apartado Resumen y Resolución del caso práctico.

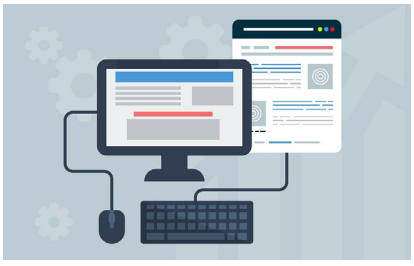


Fig. *1. Arquitecturas web*

Audio Intro. "Arquitecturas web"

https://bit.ly/3emkVZ4

/ 2. Estructura de la arquitectura web

2.1. Historia de la arquitectura web

El **concepto de arquitectura web** y su expansión tiene lugar en los años 90 con la **aparición del hipervínculo**, gracias al científico **Tim Berners-Lee**.

Fig. 2. Logo WWW

El trabajo de Berners-Lee y su equipo **permitió idear un sistema de acceso a la información de forma descentralizada**, desarrollando el protocolo HTTP (HyperText Transfer Protocol), así como un nuevo lenguaje de programación llamado HTML (HiperText Markup Language) y las URL (Uniform Resource Locator).

Gracias a la unión de estos tres avances nace la WWW (**World Wide Web**), una red de equipos interconectada a través de internet que permite compartir información.

Hasta entonces, debido a la dificultad de su uso, el acceso a la información en internet estaba presente casi exclusivamente en el **ámbito universitario y gubernamental.**

Asimismo, los primeros navegadores web no tenían una interfaz gráfica. Para acceder a la información se utilizaba un terminal para navegar en una estructura en árbol de menús.

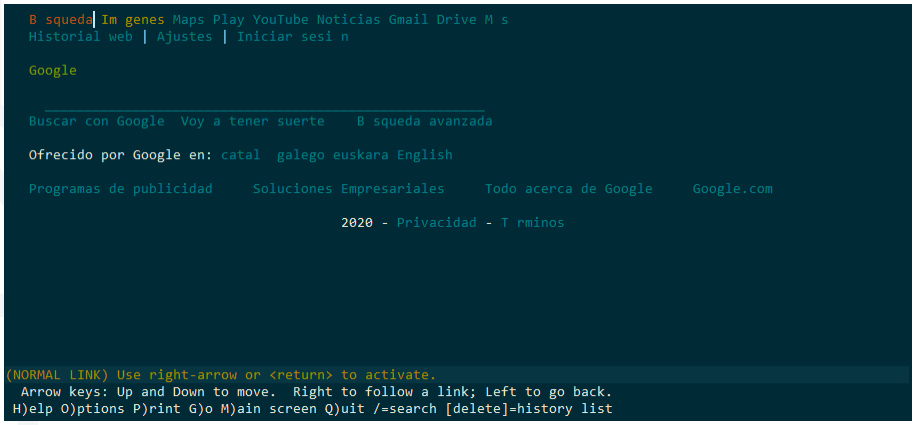


Fig. 3 Lyxn, un navegador web por consola para Linux basado en Gopher

En el año 1994, Berners-Lee fundó el W3C (**World Wide Web Consortium**), una organización cuyo objetivo fundamental es **regular las recomendaciones y los estándares utilizados, asegurando la compatibilidad y la evolución de los servicios web.**

2.2. HTTP, HTML Y URL

2.2.1. HTTP

El **protocolo HTTP** (Hypertext transfer Protocol) está diseñado para **transferir documentos HTML**, utilizando por defecto el puerto TCP 80 para establecer las conexiones. Se sitúa en la capa 7 del modelo OSI. A esta capa se la llama también capa de aplicación y engloba otros protocolos como **DNS, FTP y SSH.**

El siguiente gráfico explica qué procesos se llevan a cabo en una conexión desde que abrimos un enlace hasta que visualizamos el contenido en pantalla.

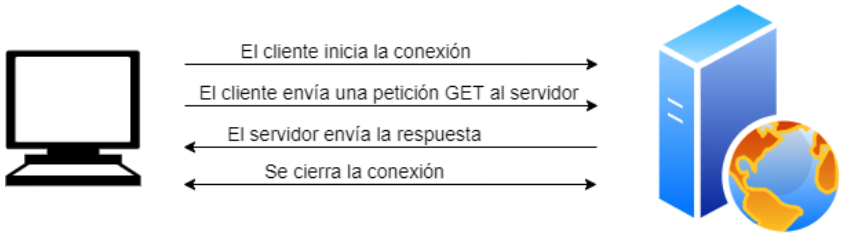


Fig. 4. Funcionamiento protocolo HTTP

Como podemos observar en la figura 4, las conexiones HTTP siempre se inician en el cliente.

Vídeo 1. "Comprobar HTTP Wireshark"
https://bit.ly/3erHMCE

2.2.2. HTML

HTML (**HyperText Markup Language**) es un **lenguaje de programación basado en etiquetas** que permite crear documentos visualmente agradables y que facilitan la lectura gracias a la posibilidad de añadir estilos de texto, imágenes, tablas o cualquier otro contenido multimedia que se nos pueda ocurrir.

En 2014 la W3C liberó la quinta versión del HTML, conocida como HTML5, que incorpora nuevas etiquetas para mejorar la compatibilidad con audio y vídeo.

2.2.3. URL

En cuanto a las URL (**Uniform Resource Locator**), se trata de la **nomenclatura utilizada para hacer referencia a objetos y poder localizar información de manera rápida y sencilla.**

La estructura de una URL es la siguiente:

esquema://servidor/directorio/archivo -> http://apache.org/

Los esquemas más conocidos son http, https, ftp, file y mailto, pero existen muchos más que podemos encontrar en la web de IANA, en el siguiente enlace:

https://www.iana.org/assignments/uri-schemes/uri-schemes.xhtml

2.3. Evolución de la arquitectura web

Hoy en día, la importancia de las aplicaciones web es tal que está empezando a utilizarse para sustituir a las clásicas aplicaciones de escritorio.

Para entender mejor las ventajas que estas aplicaciones nos ofrecen utilizaremos el siguiente ejemplo:

Si nos encontramos con un **organismo público** como, por ejemplo, un hospital, cada empleado necesita un puesto de trabajo con una serie de programas instalados para realizar sus funciones; así como cada departamento tiene unas necesidades informáticas diferentes.

Los **departamentos de informática** que se encargan de estas tareas tratan de **estandarizar el software** de los equipos y crear una plantilla o maqueta para poder instalar cada equipo nuevo con todas las aplicaciones necesarias en el menor tiempo posible. Las actualizaciones y los parches de seguridad exigen que estas plantillas se **renueven constantemente.**

La tendencia de mercado ha cambiado y ahora las aplicaciones web nos ofrecen la **facilidad de tener acceso a todos los recursos de la empresa,** incluida la información sensible de forma segura y centralizada, y estando disponible desde cualquier lugar y casi con cualquier dispositivo que cuente con un navegador web.

Otra ventaja es la reducción de costes, ya que no necesitan ordenadores potentes para ejecutar todas estas aplicaciones.

Se implementan lo que se llaman los **thin clients,** y utilizan el modelo original de terminal boba (**dumb terminal**), que se conecta a un servidor.

Audio 1. "¿Qué son los thin Clients?"
https://bit.ly/3eCTASo

En resumen, las **aplicaciones web** comparadas con las aplicaciones de escritorio nos ofrecen las siguientes **ventajas**:

- Facilidad de despliegue de nuevo software en los clientes
- Centralización de la información
- Seguridad mejorada con sistemas modernos como por ejemplo un SSO (Single Sign-On), como podría ser iniciar sesión en un sitio web autenticando contra un controlador de dominio Windows (hablaremos de esto en el tema 6).
- Compatibilidad con cualquier sistema operativo
- Reducción de costes

Fig. 5. Dumb terminal utilizada en los años 70

/ 3. Caso práctico 1: "Comprobar cómo funciona http con Wireshark"

Planteamiento: Vamos a analizar de forma práctica cómo se establece una conexión con un servidor web y de qué forma nos devuelve la respuesta.

Para ello, vamos a utilizar Wireshark, un software que nos permite capturar tráfico y analizarlo, de esta forma comprobaremos cómo se comportan los paquetes de red desde que el cliente hace la petición hasta que el servidor entrega todo el contenido.

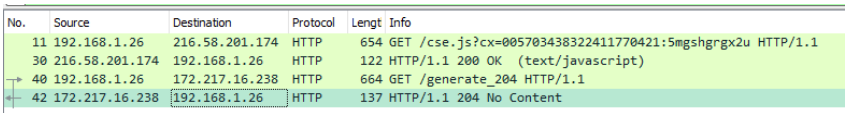
Nudo: Utilizando un navegador web, inicia una conexión con http://apache.org y captura tráfico con Wireshark para encontrar las tramas que inician, establecen y cierran la conexión.

¿Qué ocurre si repites la captura analizando el tráfico HTTPs?

Ayúdate de los filtros para facilitar la búsqueda.

Fig. 6. Logo de wireshark

Desenlace: Tras capturar el tráfico y filtrar por protocolo HTTP, obtendremos un resultado parecido al siguiente:



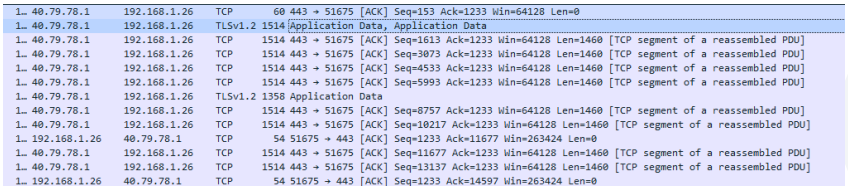
No.	Source	Destination	Protocol	Lengt	Info
11	192.168.1.26	216.58.201.174	HTTP	654	GET /cse.js?cx=005703438322411770421:5mgshgrgx2u HTTP/1.1
30	216.58.201.174	192.168.1.26	HTTP	122	HTTP/1.1 200 OK (text/javascript)
40	192.168.1.26	172.217.16.238	HTTP	664	GET /generate_204 HTTP/1.1
42	172.217.16.238	192.168.1.26	HTTP	137	HTTP/1.1 204 No Content

Fig. 7. Captura tráfico HTTP Wireshark

En la primera línea, el cliente establece la conexión enviando una petición GET al servidor, éste contesta con la información solicitada y nos devuelve tanto el código '200 OK', como la página web.

A continuación, el cliente envía una petición 204 para finalizar la conexión y el servidor nos contesta '204 No Content', que significa que la conexión ha finalizado, no existe nada más que transmitir.

Si hacemos la misma operación mediante HTTPs, al capturar tráfico comprobaremos que está cifrado y no podremos ver las peticiones en texto plano.



1…	40.79.78.1	192.168.1.26	TCP	60	443 → 51675 [ACK] Seq=153 Ack=1233 Win=64128 Len=0
1…	40.79.78.1	192.168.1.26	TLSv1.2	1514	Application Data, Application Data
1…	40.79.78.1	192.168.1.26	TCP	1514	443 → 51675 [ACK] Seq=1613 Ack=1233 Win=64128 Len=1460 [TCP segment of a reassembled PDU]
1…	40.79.78.1	192.168.1.26	TCP	1514	443 → 51675 [ACK] Seq=3073 Ack=1233 Win=64128 Len=1460 [TCP segment of a reassembled PDU]
1…	40.79.78.1	192.168.1.26	TCP	1514	443 → 51675 [ACK] Seq=4533 Ack=1233 Win=64128 Len=1460 [TCP segment of a reassembled PDU]
1…	40.79.78.1	192.168.1.26	TCP	1514	443 → 51675 [ACK] Seq=5993 Ack=1233 Win=64128 Len=1460 [TCP segment of a reassembled PDU]
1…	40.79.78.1	192.168.1.26	TLSv1.2	1358	Application Data
1…	40.79.78.1	192.168.1.26	TCP	1514	443 → 51675 [ACK] Seq=8757 Ack=1233 Win=64128 Len=1460 [TCP segment of a reassembled PDU]
1…	40.79.78.1	192.168.1.26	TCP	1514	443 → 51675 [ACK] Seq=10217 Ack=1233 Win=64128 Len=1460 [TCP segment of a reassembled PDU]
1…	192.168.1.26	40.79.78.1	TCP	54	51675 → 443 [ACK] Seq=1233 Ack=11677 Win=263424 Len=0
1…	40.79.78.1	192.168.1.26	TCP	1514	443 → 51675 [ACK] Seq=11677 Ack=1233 Win=64128 Len=1460 [TCP segment of a reassembled PDU]
1…	40.79.78.1	192.168.1.26	TCP	1514	443 → 51675 [ACK] Seq=13137 Ack=1233 Win=64128 Len=1460 [TCP segment of a reassembled PDU]
1…	192.168.1.26	40.79.78.1	TCP	54	51675 → 443 [ACK] Seq=1233 Ack=14597 Win=263424 Len=0

Fig. 8. Captura de Wireshark de tráfico HTTPS

/ 4. Tipos y modelos de arquitectura web

A medida que se desarrollaron nuevos protocolos y estándares, la forma de acceder a la información también cambió, adaptándose y mejorando.

El trabajo de **Berners-Lee** supuso un cambio de fase entre el antiguo modelo en malla o P2P y la arquitectura cliente-servidor. Esta evolución conllevó el crecimiento de los servicios web y un mayor número de servidores.

En la siguiente tabla observamos el crecimiento exponencial de los servidores en internet desde 1981 hasta 1993.

Existen tres modelos fundamentales de arquitecturas web:

- **Modelo P2P**
- **Modelo cliente-servidor**
- **Modelo de tres capas**

Date	Hosts
08/81	213
08/83	562
10/85	1,961
12/87	28,174
10/89	159,000
10/90	313,000
10/91	617,000
10/92	1,136,000
07/93	1,776,000

Fig. 9. Estudio de Harry Edward Hardy sobre el número de host en internet. Fuente.

4.1. Modelo P2P

Este modelo se caracteriza por **utilizar una topología de malla**, en la que todos los equipos deben estar interconectados entre sí y cada nodo realiza funciones de cliente, y de servidor.

Para entenderlo mediante un ejemplo real, el funcionamiento **es muy parecido a la forma de compartir archivos usando Torrent.** La información está repartida entre los diferentes servidores o peers, de modo que estos archivos no están en un único equipo, sino que se reparten en los diferentes nodos de la red.

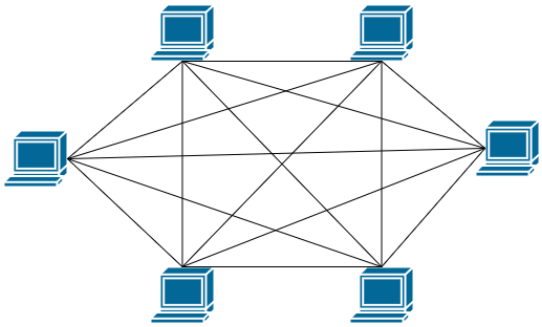


Fig. 10. Modelo P2P

4.1.1. Ventajas e inconvenientes

Analizando el modelo P2P podemos encontrar las siguientes ventajas e inconvenientes:

- **Ventajas:**
  - **Descentralización:** la información está repartida entre todos los nodos y si uno de ellos cae, el resto siguen operativos.
  - **Escalabilidad:** a medida que el número de nodos aumente, la red funcionará de manera más eficiente.
  - **Carga de trabajo repartida entre los diferentes nodos:** no hay ningún nodo que tenga mayor importancia que el resto.
- **Inconvenientes:**
  - **Posibilidad de información duplicada** o inconsistente.
  - **Mayor número de equipos que mantener:** a diferencia del modelo C/S, del que hablaremos a continuación, y en el que tenemos un único servidor centralizado, en esta modalidad las tareas de mantenimiento son mayores con relación al tamaño de la red.

4.2. Modelo cliente-servidor

A diferencia del modelo P2P, en este modelo toda la información se centraliza en un único servidor, que atenderá las peticiones de los clientes.

En arquitectura web la figura del servidor cobra importancia ya que puede responder de forma autónoma a todas las peticiones sin necesidad de consultar con otro servidor.

Desde este momento llamaremos **cliente** al navegador web que realiza las peticiones y **servidor** al equipo con funcionalidades especiales y preparado para actuar como servidor web.

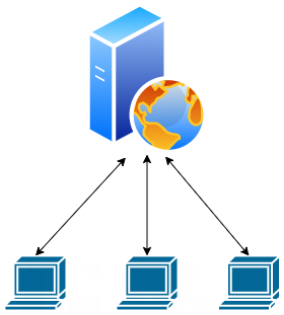


Fig. 11. Modelo cliente-servidor

4.2.1. Ventajas e inconvenientes

De la misma manera, analizaremos el modelo C/S, encontrando las siguientes ventajas e inconvenientes:

- **Ventajas:**
  - **Mantenimiento:** a diferencia del modelo P2P, solo será necesario mantener un servidor web y no una red entera.
  - **Información y recursos centralizados:** la integridad y la seguridad de la información almacenada está gestionada por el servidor.
  - **Posibilidad de balanceo de carga** entre varios servidores.
- **Inconvenientes:**
  - **Riesgo de congestión:** si la infraestructura no está preparada, ante un gran número de peticiones simultáneas el servicio puede verse afectado.
  - **Si el servidor 'está caído'**, a diferencia del modelo P2P, **no se podrá acceder a los recursos.**
  - En el modelo C/S **un único servidor afrontará una gran carga de trabajo al resolver todas las peticiones por sí mismo.** Por este motivo es necesario un hardware más potente.

4.3. Modelo de tres capas

El modelo de tres capas es una evolución del modelo C/S, que añade a la red servidores especializados en una función para optimizar el rendimiento.

Cada servidor se hace cargo de una tarea: el **servidor de base de datos** se encargará de almacenar y resolver las consultas que le lleguen, y el **servidor web** se encargará de servir el contenido solicitado por el cliente.

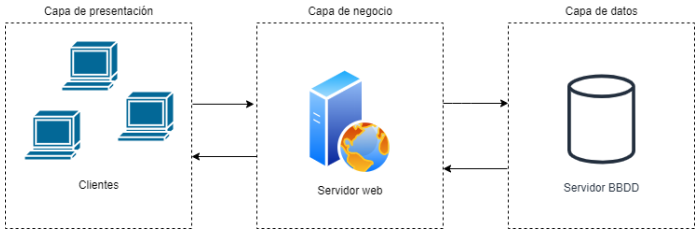


Fig. 12. Modelo de tres capas

El **funcionamiento del modelo de tres capas** es el siguiente:

- El **cliente** hace una petición al servidor web o servidor de aplicaciones.
- El **servidor web** se comunicará con la base de datos para realizar las consultas necesarias.
- El **servidor de BBDD** procesará las consultas realizadas por el servidor web y devolverá las respuestas únicamente a este último.
- El **servidor web** devolverá la información solicitada al cliente

/ 5. Caso práctico 2: “Analizando modelos en infraestructuras existentes”

Planteamiento: Una empresa gestiona la facturación y la relación de clientes usando aplicaciones web desarrolladas en JAVA. La información se almacena en una base de datos MySQL en otro servidor.

Además, el servidor (Apache) tiene configurado en su index.html una página estática con información útil para los trabajadores.

Los servidores utilizados por la empresa son:

- Servidor de base de datos MySQL
- Servidor Web Apache
- Servidor de aplicaciones Apache Tomcat

Nudo: Utilizando la información que tenemos, analiza los diferentes modelos de arquitecturas web utilizados por la empresa y argumenta tu respuesta.

Realiza un pequeño esquema de los modelos que encuentres y describe los elementos que lo componen.

Desenlace: Web estática: utiliza el modelo cliente-servidor ya que las peticiones realizadas las responde directamente el servidor Apache y éste devuelve una web estática.

Los elementos que intervienen son el cliente y servidor web.

Aplicaciones web facturación y contabilidad:

En este caso estaríamos hablando de una arquitectura con modelo de tres capas. El cliente realiza una petición al servidor web que redirige al servidor de aplicaciones, y éste en segundo plano realiza las consultas necesarias con el servidor de BBDD y devuelve la respuesta al cliente.

En este modelo encontramos el cliente, los servidores apache y tomcat en la capa de negocio y el servidor de base de datos en la capa de datos.

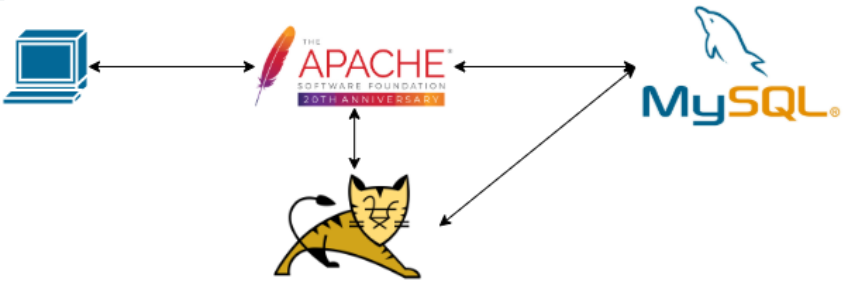


Fig. 13. Esquema caso práctico 2

**Ver Fig. 11 Ejemplo funcionamiento del modelo de tres capas.*

/ 6. Ventajas e inconvenientes modelo 3 capas

Después de haber estudiado este modelo podemos llegar a las siguientes conclusiones:

- **Ventajas:**
  - **Mayor seguridad** que el modelo C/S al poder definir permisos por niveles.
  - **Mayor flexibilidad** al existir servidores especializados.
  - **Mejor rendimiento** al compartir tareas entre los diferentes servidores.
- **Inconvenientes:**
  - Es necesario **aumentar el número de servidores.**
  - Se genera un **mayor tráfico de red** entre los diferentes servidores.
  - Si no se configura correctamente podría suponer un **riesgo de seguridad**, ya que el servidor web y el de BBDD intercambian paquetes de red con las consultas.
  - Es posible que se produzca un **mayor tiempo de espera** hasta que el cliente descarga toda la información debido a los procesos internos que tienen lugar entre los diferentes servidores.

6.1. Comparativa de los tres modelos

Para poder comparar mejor y entender las ventajas de cada modelo revisaremos la siguiente tabla:

	P2P	Cliente Servidor	Modelo de tres capas
Riesgo de congestión	NO	SI	SI
Información centralizada	NO	SI	SI
Disponibilidad de los datos si el servidor principal deja de estar operativo	SI	NO	NO
Mantenimiento complicado1	SI	NO	NO
Mejor rendimiento que su predecesor	-	SI	SI

Tabla 1. Comparativa 3 modelos arquitecturas

Debemos recordar que el modelo P2P tendrá un mejor rendimiento cuanto más grande sea la red. Mientras que en el modelo C/S solo mantenemos un servidor, en el modelo de tres capas este número se incrementa según las necesidades del desarrollo.

Vídeo 2. "Ataque DOS a servidor Apache"
https://bit.ly/3esmNj9

/ 7. Resumen y resolución del caso práctico de la unidad

En este tema nos hemos podido **acercar al mundo de las aplicaciones web** y conocer qué elementos son necesarios para que toda esto sea posible.; desde un paseo por la historia de la World Wide Web y su nacimiento, hasta una primera aproximación al protocolo HTTP.

También hemos **analizado los diferentes modelos de arquitecturas existentes:** P2P, Cliente-Servidor y el modelo de tres capas; y hemos evaluado sus ventajas e inconvenientes teniendo en cuenta su funcionalidad y aspectos esenciales como el rendimiento y la seguridad.

Resolución del caso práctico de la unidad

En el caso práctico inicial se nos plantea el dilema sobre si una empresa debería dar el salto a las aplicaciones web, o invertir recursos en un servidor físico, en donde instalarían un software de gestión de escritorio. Además, para esta segunda opción deberían instalar en todos los equipos el software de cliente y afrontar el coste del servidor y las licencias por cada equipo en el que se instale.

Debemos analizar, entre otros, factores como:

- Recursos económicos invertidos
- Escalabilidad
- Dificultad de implantación y mantenimiento
- Copias de seguridad y plan de contingencia

Valorando todos los datos optamos por una aplicación web en la nube, la cual sea accesible desde cualquier ordenador y sin necesidad de instalación. Esto nos facilita la implantación en el parque de equipos ya que la solución se reduciría a iniciar sesión en una web con nuestro usuario y contraseña. Otro punto importante es que, al contratar un servidor en la nube, evitamos el problema de averías hardware y mantenimiento de un servidor físico. En la figura 10 podemos ver como el proveedor OVH nos ofrece la opción de backup automático en un VPS (Servidor Privado Personal) teniendo un coste de poco más de 30€ al mes.

/ 8. Bibliografía

Rodriguez, A., Gatrell, J., Karas, J., & Peschke, R. (s. f.). TCP/IP Tutorial and Technical Overview (7th Edition). En TCP/IP Tutorial and Technical Overview (7th Edition) (7.a ed., pp. 605-613). 1, 1: Prentice Hall.

Escobedo, J. G. (2013, 19 septiembre). Modelos de arquitecturas web. Recuperado 14 de junio de 2020, de https://javiergarciaescobedo.es/despliegue-de-aplicaciones-web/76-arquitecturas-web/253-modelos-de-arquitecturas-web#:%7E:text=Los%20modelos%20de%20arquitecturas%20web,las%20p%C-3%A1ginas%20y%20aplicaciones%20web.

Pisuwala, U. (2019, 23 diciembre). Fundamentals of web application architecture. Recuperado de https://www.peerbits.com/blog/web-application-architecture.html

Martínez, F. J. (s. f.). Implantación de aplicaciones web (GRADO SUPERIOR) (Spanish Edition) (1.a ed.). 1, 1: Ra-Ma.

Banga, S. (s. f.). Web Application Architecture: Definition, Models, Types, and More. Recuperado de https://hackr.io/blog/web-application-architecture-definition-models-types-and-more

Hardy, H. E. (s. f.). The History of the Net. Recuperado de http://www.cs.kent.edu/%7Ejaved/internetbook/nethistory/nethist.html