**Despliegue de Aplicaciones Web**

Resumen 1ª Evaluación

Guía:

· Conceptos Importantes.

· Conceptos complementarios.

· Software utilizado.

· Comandos / Acciones.

## Unidad 1. Arquitecturas Web.

## Introducción

Una **aplicación web** (o web app) son herramientas de software que los usuarios pueden utilizar accediendo a un servidor web a través de internet o de una intranet mediante un navegador.

Tiene al menos dos partes: una en el lado del servidor, donde está la información o los datos que necesita el usuario, y otra es ejecutada en el navegador web de la máquina cliente.

Esta aplicación necesita una estructura para ser accesible desde diferentes dispositivos. Esta estructura es conocida como **arquitectura web**.

A **web application** (or web app) is application software that runs on a web server, unlike computer-based software programs that are run locally on the operating system (OS) of the device. Web applications are accessed by the user through a web browser with an active internet connection.

It has at least two parts: one in the server side, where the information or data that the user needs are, and the other is executed in a web browser in the client machine.

This web application needs a structure to be accessible from different devices. This structure is known as **web architecture**.

**Arquitectura web** es el diseño de una estructura de la aplicación web, incluyendo estudio, análisis, organización, disposición y estructura de la información de las páginas.

La mayoría de las arquitecturas web actuales están basadas en el modelo cliente/servidor.

A **web architecture** is the design of a web application’s structure, including study, analysis, organization, disposition and structure of the information of web pages.

Most of the current web architectures are based on the client/server model.

## Features of web architectures.

The main web architecture features are:

**Scalability**: Una aplicación es escalable cuando podemos incrementar el número de datos o su tamaño. Es importante saber que la escalabilidad y la eficiencia no es lo mismo. Una web puede ser rentable pero no escalable.

Hay tres formatos:

Vertical: Es la más sencilla. Consiste en incrementar el hardware para incrementar el rendimiento. La principal ventaja es que no necesita ningún cambio, solo tienes que conseguir una máquina mejor. La desventaja es que el presupuesto es limitado para incrementar el hardware o no existe mejor máquina.

Horizontal: Consiste en añadir más máquinas, incrementando la capacidad. En este caso, aunque la máquina no necesite hardware caro, la aplicación tiene que hacer algunos cambios.

Para hacerlo, necesitamos una arquitectura web. La más usada es **shared nothing**, donde cada nodo (o máquina) es independiente, autosuficiente y “autocontenido”, eso significa que la aplicación es compartida por todo el sistema, no es almacenada en nodos, si un nodo falla, no perderemos la información y el sistema seguirá trabajando. Google usa esta estrategia.

Cluster: Un cluster consiste en dos o más ordenadores trabajando juntos para proporcionar un mayor nivel de disponibilidad, fiabilidad, y escalabilidad que puede ser obtenida por un solo ordenador.

Por lo que, un server cluster es un grupo independiente de servidores trabajando juntos como un solo sistema proporcionando alta disponibilidad de servicios para clientes. Cuando falla un ordenador de un cluster, los recursos son direccionados y la carga de trabajo es distribuida a otro ordenador en el cluster. Se puede usar para asegurar un constante acceso a recursos importantes.

• **Portability**: La portabilidad se refiere a la capacidad de la aplicación web de ejecutarse en diferentes plataformas o arquitecturas con las mínimas modificaciones.

Una aplicación web tiene que ser capaz de adaptarse a diferentes posibles arquitecturas.

• **Design Patterns**: Son soluciones generales reusables a un problema frecuente que ocurre en el desarrollo de aplicaciones. Permiten la comunicación entre analista y desarrollador, ayudando a su desarrollo y mantenimiento. Un ejemplo de pattern es el “creacional”. Es el esqueleto de inicialización y configuración para un Model-View-Controller (MVC).

El MVC es un patrón de arquitectura de software para implementar interfaces de usuario en ordenadores. Divide la aplicación en tres capas:

* El modelo administra los datos, lógica y reglas de la aplicación.
* La vista es la representación de la información: gráficos, diagramas, etc.
* El controlador acepta entradas y las convierte en comandos para las otras dos capas.

• **Responsibility separation**: Significa que las diferentes responsabilidades deben ir a diferentes capas. Hoy en día se tiende a separar lo máximo posible las responsabilidades en capas. Por lo que cada capa trabaja en una parte de la aplicación. La más usada es “3-independent layers”:

* Presentación (Cliente): el navegador web es ejecutado en la máquina del usuario.
* Business (Web server): La capa de presentación puede acceder a la lógica y procedimientos de esta capa, pueden ser archivos HTML, Java, etc. Algunas tecnologías usadas en esta capa son PHP. Perl, ASP, etc.
* Persistence (Data): Es un sistema de administración para acceder a diferentes datos. Siempre es un Database Management System (DBMS) pero también archivos XML.

• **User’s session management**: Una sesión de usuario es el tiempo de actividad que el usuario, con una dirección IP, pasa en el sitio web. El número de sesiones son usadas para calcular los datos de tráfico en una aplicación.

## Web architecture Models.

Hay varios modelos de arquitectura, como el W3C (World Wide Web Consortium) que describe cuatro tipos diferentes: Message Oriented Model, Service Oriented Model, Resource Oriented Model and Policy Model.

**Client/Server Model.**

Este modelo trabajo como una aplicación distribuida que distingue proveedores (Servidores) de peticionantes (Clientes). Entonces, un cliente pide información al servidor, y este le contesta.

Normalmente el cliente y el servidor no son la misma máquina, pero es posible.

Un ejemplo básico es cuando un cliente solicita una página web a un servidor. Pasos:

1. From the client web browser, the user request a web page writing the URL (Uniform Resource Locator). For example, http://www.ceedcv.org
2. The server receives the request.
3. The server looks for the web page in its management system.
4. The server sends the source code of the web page to the client.
5. The client receives the source code and the web browser shows the web page.

It is interesting to know that the Business layer can be divided into two more layers:

* **Presentation layer**: it is in charge of integrating the dynamic and the static part of the web page. Some solutions to do this is to use languages like ASP, JSP or PHP.

A static web page is delivered to the user exactly as stored. They are HTML documents stored as files in the file system and made available by the web server over HTTP.

A dynamic web page is a web page whose construction is controlled by an application server processing server-side scripts (PHP, ASP, etc.). It processes the web pages that are send by the client, for example the data of a form.

The HTTP protocol (Hyper Text Transfer Protocol) is the most used in the Internet. Its objective is to allow the files transfer between a browser and a web server through an URL address.

* **Business logic layer**: it is the most complex. It controls the results of the functional analysis (it realizes all the operations and validations, manages the workflow and manages the sessions) and manages all the data access operations to the Presentation Layer, showing the set of business rules. It used to be located in an “Applications Server”.

Some advantages of this model are:

- Centralization of the information: the access, resources and data are controlled by the server avoiding that an unauthorized client can enter to the system and damage it.

- Scalability: you can increase the number of clients or servers separately.

- Easy maintenance: all the functions and responsibilities are distributed between some independent machines so is easy to replace, repair or update any server.

-There are many technologies based on this model.

And some disadvantages are:

- Management of the traffic congestion: if many clients send petitions to the same server, it can be down.

- Resource access: if a server falls down, its resources are not available.

- Dependence of hardware and software: normally we need specific software and hardware for each case.

An example of this model is a LAMP server.

## Web servers.

A web server is a program or a set of them, which offer services over a network. Its aim is to offer necessary resources in order that two or more programs could communicate independently.

The web servers belong to the system called distributed, which means that the system is made up of more than one machine or a set of hardware. Thus, the resources must distribute among them. The technology that takes carry of this is known as load balancing.

The load balancing is implemented by an algorithm that distributes the tasks in the most efficient and equitable possible way, avoiding the bottlenecks.

## How to have a web server.

First of all, we need to have a powerful machine, dedicated and able to attend all the requests.

If it is not possible to have this kind of machine, the web hosting is a good solution to a reasonable price, in exchange for a specialized company that looks after our resources.

So we will need:

- A stable operating system, with security and permissions control.

- A static IP address (an IP address which never changes), to allow the remote access.

- A Domain Name Server (DNS) to accede easier to our server.

- 24 Internet Connection to be accessible from another networks.

- Specific software of web server.

## Web hosting control panels.

Nowadays, the hosting companies allow to costumers to manage their hosted services using a webbased interface.

Some available modules are:

• Access to server logs.

• Create and configure virtual web servers.

• DNS Server settings and routing configuration.

• Details of available and used webspace and bandwidth.

• Email account configuration.

• Maintaining FTP users’ accounts.

• Managing database.

• Visitor statistics using web log analysis software.

• Web-based file manager.

[Comparison of web hosting control panels - Wikipedia](https://en.wikipedia.org/wiki/Comparison_of_web_hosting_control_panels)

**Webmin.**

Webmin is a program that simplifies the process of managing a Linux or Unix system. Normally you need to manually edit configuration files and run commands to create accounts, set up a web server or manage email forwarding. Webmin lets you perform these tasks through an easy to use web interface, and automatically updates all of the required configuration files for you. This makes the job of administering your system much easier.

Some of the things that you can do with Webmin are:

• Create, edit and delete Unix accounts on your system.

• Export files and directories to other systems with the NFS protocol.

• Set up Disk Quotas to control how much space users can use up with their files.

• Install, view and remove Software Packages in RPM and other formats.

• Change your system's IP address, DNS Server settings and routing configuration.

• Set up a Linux Firewall to protect your computer, or to give hosts on an internal LAN access to the Internet.

• Create and configure virtual web servers for the Apache Webserver.

• Manage databases, tables and fields in a MySQL or PostgreSQL Database Server

• Share files with Windows systems by configuring Samba Windows File Sharing.

# Unidad 2. Involved Services. DNS, FTP y SSH.

1. Introduction.

# Unidad 1. Actividad.