**2. BASES TEORICAS.**

**2.1. MARCO HISTORICO Y CONTEXTUAL**

EL hospital privado Santa Fe se encuentra localizado en la carretera de Villahermosa a Macuspana del estado de Tabasco, Mexico. Esta privilegiado por estar muy bien ubicado cerca del Hotel Hilton y el aeropuerto de Villahermosa, principales lugares de extranjeros y reuniones de negocios. Esta ubicada en la colonia dos montes, una colonia segura libre de inundaciones, en proceso de modernización. Al ser un hospital privado de especialidades multiple y realizar cirugias tiene todos los servicios de máxima seguridad lo que le da mas ventaja en su infraestructura. Electricidad, internet, agua, gas, aire acondicionado, teléfonos, arquitectura, todo en el hospital esta normalizado teniendo una gran inversión y por lo mismo siendo un lugar con todos los servicios para partir de un proyecto. Ademas esta muy controlado en la parte del hospital, por las enfermedades, curación de pacientes, asi que se trabajaría en la area administrativa que por suerte esta desacoplada para ejercer a una distancia segura de la parte operativa del hospital.

Al estar localizado fuera de la ciudad para llegar al hospital desde la ciudad de villaherosa de tres formas:

Por auto propio. (15 minutos de distancia aproximadamente, cuenta con estacionamiento).

Por colectivo. (Combis en el mercado pasan muy pocas cada media hora. Autobuses de escala a Macuspana pasan cada 45 minutos. No hay transporte después de las 21:00 horas.)

Servicio vehicular particular (Taxi: En cualquier momento pero cobran caro, mas de noche).

No tienen cerca consultorias de Tecnologias de la información, no es una zona de negocios sino de reuniones y de llegadas.

El personal administrativo esta de 9:00 a 2:00 pm, esta saturado en sus ocupación siendo que atiende clientes, proveedores, empleados, maneja los servicios hospital, contrata servicios externos, negocios, etc. Por lo tanto es difícil conseguir su apoyo para estar en el proyecto.

El director del hospital se le tiene que reservar, y usualmente esta de tarde.

Necesidad de normalización con los procesos de tecnologías de la información.

Falta de un equipo de tecnologías de la información en su infraestructura.

La comunidad en la que influye esta empresa mas de cerca es a sus empleados, en sus inversores, clientes antiguos y actuales, el aeropuerto y el hospital, y la colonia dos montes.

Las personas que podrían colaborar en la ejecución de un proyecto de intervención serian el Director del hospital, la administradora, otros administradores, y el encargado de marketing.

Diferentes puntos de vista de cada uno del proyecto.

Director: Quiero una pagina que me muestre lo que he invertido en este hospital, profesional que comunique y refleje estatus.

Administrativo: Expandir las operaciones del hospital, sus activos fijos, la ganancia, el potencial de ganancias, el aumento de otros activos mediante este desarrollo del proyecto.

Encargado de marketing: Que muestre todo, se ve bonito, normalizado, estándar, permita al hospital relacionarse y comunicarse, ampliar sus expectativas y oportunidades de canal.

Personal es consultivo o decisivo. Director es decisivo. Marketing es decisivo. Administracion y encargado de cuentas consultivo.

Utilidad a que personas complementaria en sus funciones de la empresa (interesados) un proyecto de intervención: este proyecto apoyaría al director porque es el complemento del todo al administritativo para que tenga mas activos. Encargado de marketing y publicidad.

Otras partes que se (accederían por el beneficio) beneficiarían con el recurso que le falta al hospital (o con la mejora de los problemas del hospital) (y no son tan importantes). Clientes ocasionales, otras colonias de Villahermosa, municipios y estados de los alrededores, comunidad de doctores, sección amarilla.

Antecedentes: El hospital ha estado bajo remodelación los últimos 5 años, siendo este un gasto muy fuerte y que requería toda su atención. Esta remodelación trajo al hospital el poder ofrecer una serie de servicios de ultima punta, que lo hace destacar como los primeros 5 mejores hospitales privados de tabasco estando a la altura de hospitales privados de renombre como el air, del sureste, angeles.

La empresa tiene doctores que tienen experiencia tanto fuera como dentro de Villahermosa, siendo la visión de sus especialistas muy amplia. El dueño de la empresa es un reconocido doctor de Tabasco, que además tiene una buena visión por los hospitales en los que ha estado, una visión de negocios e integral en las caracteristicas del servicio.

Se promocionan por llamadas, carteles, folletos, radio, periódicos, y sobre todo por el renombre de los médicos que trabajan en ese hospital, siendo doctores muy solicitados y que usualmente tenían una relación de trabajo o reconocimiento de sus habilidades previa por el dueño del hospital. Siendo su personal ejecutivo, después el coste de sus instalaciones y el servicio responsable de sus empleados del que no se ha tenido queja lo que lo mantiene publicitado.

Surge pues la necesidad de un proyecto que encauze esas areas que por su mismo contexto no han tenido relación y que reflejan los problemas que en estos momentos tiene esta empresa.

La empresa mueve a sus miembros mediante las responsabilidades delegadas para buscar el arreglo de las faltas y necesidades en base a su prioritizacion y urgencia definida. Usualmente dejando aquellas que no pueden ser resueltas por su propia estructura organizacional, una vez que se han definido presupuestos y generación de juntas para aprobarlas, contratando expertas externas por políticas de la empresa muy raramente considerando añadir eso a su estructura organizacional, por políticas de ahorro de la empresa.

Los miembros inconformes con la situación o que ven problemas, lo expresan verbalmente cumpliendo su misión de atender específicamente el area que les corresponde.

Indirectamente se han visto estos problemas en:

Falta de Proyectos o dificultad para iniciar cambios en sus procesos de internet. Años sin tocar su pagina web. Desactualizacion de la información.

Disconformidad con su web actual

Pocas visitas web. Pocos beneficios del rendimiento de la web. Desantender el desarrollo de la web, o postergación por la falta de ver un camino, o saber en que punto están, por la misma falta de identificación del problema.

¿Qué hacen? Contratan a otros en base a sus necesidades especificas sin hacer un análisis de las relaciones en las que forman los problemas. Por lo mismo ellos especifican estas necesidades y los servidores las resuelven, pero no atacan a los factores que las originaron y el problema sigue apariciendo en diferentes aspectos pero al final en la misma dimensión.

¿En que plazo se debería dar una respuestas? En cuanto se haga un análisis detallado de la ubicación actual ya que ellos mismos están buscando que sus necesidades particulares se resuelvan, pero lo que se necesita es una intervención especifica que interrumpa este ciclo de incidencias en el que han caído por su propio contexto. Un proyecto de esta magnitud dura alrededor de 4 meses, siendo necesario una determinación del problema. Del que se propondrá un modelo o patrón para salir de este.

El nudo actual: su pagina web ha quedado estatica, obsoleta, por un espacio grande de años y no hay nadie que la atienda o sepa como funciona y no se puede hacer uso o beneficie de ella al no haber documentación o departamento experto en su estructura organizacion que intuya cómo funciona y le aplique una forma de sacar beneficios. Se esta pagando un mantenimiento que se traduce como gasto en la empresa en vez de inversión ya que no se tiene expectativa de que mejore.

**2.2 MARCO REFERENCIAL**

MARCO REFERENCIAL

I made students in my software engineering class read some of the

patterns as part of a section on reengineering. The section went well even

though none of the students got excited by the patterns. They didn’t have

the experience to evaluate them. However, one of the students came back

to me after his summer job and said that of everything in the course, the

material that was the most useful was the patterns on reverse engineering.

Before that experience, the patterns seemed believable. Afterwards, they

were believed!

( We see reengineering patterns as stable units of expertise which can be

consulted in any reengineering effort: they describe a process without

proposing a complete methodology, and they suggest appropriate tools

without “selling” a specific one.)

“People are charged with the task of figuring out whether it would be better

to rearchitect or rewrite the system. In other organizations, the system limps

along for years. Yes, it takes longer than it should to add new features, but that

is just considered the price of doing business. Nobody knows how much better

it could be or how much money is being lost because of poor structure.”

La arquitectura de un proyecto permite alcanzar los requerimientos y abrir las expectativas que el cliente merece. Entre otras cosas la evaluacion de una arquitectura nos permitirá saber si esta a la altura de los cambios y los nuevos requerimientos que se piden.

“From November 1996 to December 1999, we participated in a European

industrial research project called FAMOOS (ESPRIT Project 21975—

Framework-based Approach for Mastering Object-Oriented Software Evolution).

The partners were Nokia (Finland), Daimler-Benz (Germany), Sema

Group (Spain), Forschungszentrum Informatik Karlsruhe (FZI, Germany),

and the University of Bern (Switzerland). Nokia and Daimler-Benz were

both early adopters of object-oriented technology, and had expected to

reap significant benefits from this tactic. Now, however, they were experiencing

many of the typical problems of legacy systems: they had very

large, very valuable, object-oriented software systems that were very difficult

to adapt to changing requirements. The goal of the FAMOOS project

was to develop tools and techniques to rejuvenate these object-oriented

legacy systems so they would continue to be useful and would be more

amenable to future changes in requirements.

Most surprising to us, however, was the fact that, although each of

the case studies we looked at needed to be reengineered for very different

reasons—such as unbundling, scaling up requirements, porting to new

environments, and so on—the actual technical problems with these systems

were oddly similar. This suggested to us that perhaps a few simple

techniques could go a long way to fixing some of the more common problems. “

The Intranet Team then began to review open source solutions. Open source

technology had not been used extensively at Florida Hospital (outside of the server

OS arena), so the team considered their choices carefully. After looking at the leading

open source platforms, the team recommended implementing Drupal. Intranet Team

Manager, Rick Mann, cites the rational to go with Drupal, “Drupal offered a wide range

of tools, an active development community, and we determined that we could get the

core platform running relatively quickly and easily.”

Acquia Drupal, or Acquia’s distribution of Drupal that includes Drupal core and a set

of essential modules, helped get the Florida Hospital Intranet Team on the right track.

By initially paring down available Drupal tools, the team could continue with their

decentralized content management style that allows departments to manage their

own sites. This would be key to the team’s ability to support such a large number of

sites. In theory, the team would create and host departmental team Drupal sites using

a standard template and consistent set of tools (modules), and then pass over the

day-to-day management of sites to their respective site owners and content creators.

This would, in turn, free up the Intranet Team to subsequently research, develop and

deploy additional tools and functionality, to further enrich the Drupal experience.

The team’s goal of supporting a large number of intranet departmental sites was met,

with 4 employees supporting nearly 100 separate, but standardized Drupal sites used

for collaboration and content management. Because of Drupal’s ease of deployment,

and the diligence put into creating a pre-packaged site template, these 4 team

members are able to support their intranet, while also balancing a full workload of

other, unrelated IT duties. A large portion of success is also due to excellent end-user

training by the Florida Hospital MIS training team, and diligent research and testing for

the core set of modules included in their standard department template.

Staying connected to the user base is also important to the Florida Hospital Intranet

Team. They provide on-going collaboration with all the site owners and content

creators by implementing an internal “Drupal Community” site, as well as quarterly

“Lunch and Learn” presentations and email blasts. In this way, the Intranet Team can

provide additional training tips, provide a repository of FAQ’s, and help “close the

support loop”.

As the sites become more mature, the team is looking to add more functionality in a

controlled fashion. Says Rick Mann, “We are soliciting feedback from the departments

about what features they’d like to see. We are currently in the planning stages of

adding additional functionality, such as web forms, CCK and Views. We will also

expand our training to include “advanced” classes, empowering our users to leverage

these additional, powerful tools.”

Carlos Escobar draws a fine point on the role that Acquia has played, “Acquia has

demonstrated great thought leadership and guidance. The value isn’t just the cost

savings; it’s the quality of the relationship. It’s true that we’re saving at least 25% over

the legacy vendor support; but that was also very low-level, reactive support. We

only heard from the vendor when there was a problem, or when it was time to renew

contracts. With Acquia, we are working together to build smarter applications, and

improve our ability to get the most out of Drupal. Acquia isn’t selling software, so

their success is more closely tied to ours. Because of this, we’ve been able to build a

fantastic and unique partner relationship with Acquia.”

**2.3 MARCO TEORICO**

UNA HIPOTESIS. Sale de lo anterior?

El marco teorico respalda la solución del proyecto. Mediante argumentos teoríca y referenciales, evaluación, presentación y pertinencia de enfoques y resultados de teorías e investigaciones en diversas areas de conocimiento.

**I. COMO FUNCIONAN LAS APLICACIONES DE INTERNET**

**1. Aplicaciones web**

Una [aplicaci](http://www.alegsa.com.ar/Dic/aplicacion%20web.php)ón web es cualquier aplicación que es accedida vía [we](http://www.alegsa.com.ar/Dic/web.php)b por una [red](http://www.alegsa.com.ar/Dic/red.php) como [internet](http://www.alegsa.com.ar/Dic/internet.php) o una [intranet](http://www.alegsa.com.ar/Dic/intranet.php). En general, el término también se utiliza para designar aquellos programas informáticos que son ejecutados en el entorno [del navegador](http://www.alegsa.com.ar/Dic/navegador.php) o codificado con algún lenguaje soportado por el navegador; confiándose en el navegador web para que reproduzca ([renderice](http://www.alegsa.com.ar/Dic/renderizacion.php)) la aplicación.

En la ingeniería de software se denomina aplicación web a aquellas herramientas que los usuarios pueden utilizar accediendo a un servidor web a través de Internet o de una intranet mediante un navegador. En otras palabras, es una aplicación software que se codifica en un lenguaje soportado por los navegadores web en la que se confía la ejecución al navegador.

Las aplicaciones web son populares debido a lo práctico del navegador web como cliente ligero, a la independencia del sistema operativo, así como a la facilidad para actualizar y mantener aplicaciones web sin distribuir e instalar software a miles de

usuarios potenciales.

Es importante mencionar que una página Web puede contener elementos que permiten una comunicación activa entre el usuario y la información. Esto permite que el usuario acceda a los datos de modo interactivo, gracias a que la página responderá a cada una de sus acciones, como por ejemplo rellenar y enviar formularios, participar en juegos diversos y acceder a gestores de base de datos de todo tipo. Desde la perspectiva de un usuario, puede ser difícil percibir la diferencia entre un sitio web y una aplicación web. Según el **Diccionario Oxford** en línea, una aplicación es "un programa o conjunto de programas para ayudar al usuario de un ordenador para procesar una tarea específica". Una aplicación web es básicamente una manera de facilitar el logro de una tarea específica en la Web, a diferencia de un sitio web estático que es más bien una herramienta de difusión menos personalizada.

El término más decisivo de esta definición es "tarea específica". La aplicación web por lo tanto permite al usuario interactuar directamente contigo y tus datos, todo en forma personalizada, para llevar a cabo esa tarea específica. Si ya has puesto en marcha una búsqueda en Google, has usado una aplicación web. Esto se aplica incluso si: lees el correo desde el correo electrónico de Gmail, pagas tus cuentas en línea, compras en la Web, te estás comunicando con tus amigos y, por último, mantienes tu jardín virtual. El denominador común sigue siendo el mismo: el uso de una aplicación web para llevar a cabo todas estas tareas específicas.

**2. Historia de las Aplicaciones web**

El concepto de la aplicación web no es nuevo. De hecho, uno de los primeros lenguajes de programación para el desarrollo de aplicaciones web es el "Perl". Fue inventado por Larry Wall en 1987 antes de que internet se convirtiera en accesible para el público en general. Pero fue en 1995 cuando el programador Rasmus Lerdorf puso a disposición el lenguaje PHP con lo que todo el desarrollo de aplicaciones web realmente despegó. Hoy en día, incluso muchas de estas aplicaciones se han desarrollado en PHP, como Google, Facebook y Wikipedia.

Unos meses más tarde, Netscape, el navegador web más antiguo y popular, anunció una nueva tecnología, JavaScript, lo que permite a los programadores cambiar de forma dinámica el contenido de una página Web que había sido hasta el momento texto estático. Esta tecnología permite un nuevo enfoque para el desarrollo de aplicaciones Web, que eran, y aún hoy, mucho más interactivas para los usuarios. Por ejemplo, la instantánea de Google, que muestra los resultados de búsqueda en un momento en que la palabra se escribe, hace un uso intensivo de JavaScript. Las actualizaciones del sitio web de productos de Microsoft también utiliza esta tecnología. Al año siguiente, en 1996, dos desarrolladores, Sabeer Bhatia y Jack Smith lanzaron Hotmail (no fue un desarrollo original de microsoft), un servicio de correo en línea que permite (por primera vez) para el público en general para acceder y consultar el correo electrónico siempre que sea los usuarios pudieran estar en cualquier sitio lejos desde su ordenador.

Luego vino la famosa plataforma Flash utiliza para añadir contenido interactivo para sitios Web. Flash hizo su aparición en 1997, conocido como Shockwave Flash. Más tarde, después de ser adquirido por Macromedia y Adobe, Flash se convirtió en una plataforma para desarrollar aplicaciones web interactivas.

El año siguiente marcó un punto de inflexión para los medios de comunicación en línea. De hecho, el 17 de enero de 1998, el sitio web The Drudge Report anunció por primera vez un informe de noticias antes de que se difundiera en los medios de televisión y la prensa tradicional. Se informó el escándalo Clinton/Lewinsky. Este evento fue el detonante del periodismo en línea tal como lo conocemos hoy en día. Antes de esa fecha, internet nunca había sido considerado un medio de comunicación importante.

El mismo año, la compañía Google desarrolló su primer motor de búsqueda en línea que, por su nueva forma de indexar páginas web, facilita enormemente la búsqueda de información en internet. Google sigue innovando y se convirtió en uno de los más prolíficos en cuanto a las aplicaciones Web, con indicación del muy popular Google Maps, Google Docs, Gmail y en aumento. A principios de 2001, poco después de la explosión de la burbuja de internet,

Wikipedia se lanzó como un sub-proyecto de Nupedia, una enciclopedia en línea tradicional.

Para desarrollar su plataforma, se utiliza un tipo de Wikipedia de la aplicación web denominada "wiki", que permite a cualquier usuario agregar contenido. Las contribuciones no se hicieron esperar, y al final del primer año de funcionamiento, Wikipedia ya contaba con 20000 páginas en 18 idiomas. Hoy en día, casi 21 millones de artículos en 285 idiomas conforman el sexto sitio más visitado en el mundo, siendo el primero Google. En 2003, MySpace fue fundado y más tarde, de

2005 a 2008, el sitio se convirtió en el medio de comunicación social más visitado. MySpace fue una plataforma de lanzamiento para otras aplicaciones web conocidos como YouTube, y Slide.com! RockYou, todos los cuales comenzaron como módulos adicionales para los usuarios de MySpace antes de convertirse en sus propios sitios web en su propio derecho. Entonces, tres acontecimientos muy importantes ocurrieron en 2004. En primer lugar, en una conferencia de la Web 2.0 a cargo de John Battelle y Tim O'Reilly, el concepto de "**web como plataforma**" fue mencionado por primera vez.

Esta innovación allanó el camino para futuras aplicaciones web, es decir, un software que aprovecha las ventajas de la conexión a internet y que se desvían del uso tradicional del escritorio. En segundo lugar, el sitio interactivo de Digg se puso en marcha. Propuso una forma innovadora de crear y encontrar contenido en internet mediante la promoción de noticias y enlaces democráticamente votado por los usuarios. Y, por último, el tercer gran evento, pero no menos importante, fue el lanzamiento de Facebook, que estaba entonces en su infancia, abierto sólo a los estudiantes. Con un millón de suscriptores a finales de 2004, Facebook se ha convertido en el medio de comunicación social más utilizado con más de 900 millones de usuarios. Este es el segundo sitio más visitado en el planeta y tiene la mayor cantidad de fotos compartidas por los usuarios con un total de casi 500000 millones de fotos subidas a la plataforma.

En 2005, YouTube fue lanzado oficialmente, permite a los usuarios compartir vídeos en línea. De simple sitio para compartir vídeos en internet a una plataforma madura que se conoce hoy en día, YouTube ahora ofrece cerca de 4000 millones de videos al día, además de un servicio de alquiler de películas en línea, y, finalmente, episodios de emisión para las empresas o las películas de MGM, Lions Gate Entertainment y CBS. Twitter, por su parte, se puso en marcha en 2006. Con los años, la popularidad de Twitter ha aumentado de 1,6 millones de 'tweets' en

2007 con la impresionante cifra de 340 millones de dólares por día en marzo de

2012 (equivalente a casi 4000 'tweets' por segundo). El año 2007 estuvo marcado por la aparición del iPhone, que fue sin duda responsable de la llegada de la nueva moda para las plataformas móviles y aplicaciones web. Ahora son accesibles por teléfono inteligente. A principios de 2011, la empresa Kickstarter, que facilita la financiación de

proyectos en línea de forma participativa, ha llegado a los 4000 proyectos con más de 30 millones de dólares en donaciones. Por otra parte, casi el 44% de los proyectos se han iniciado con éxito desde esta plataforma. En resumen, los desarrolladores de aplicaciones web de hoy están dando forma al futuro digital.

**3. Consideraciones técnicas**

Una ventaja significativa es que las aplicaciones web deberían funcionar igual independientemente de la versión del sistema operativo instalado en el cliente. En vez de crear clientes para Windows, Mac OS X, GNU/Linux y otros sistemas operativos, la aplicación web se escribe una vez y se ejecuta igual en todas partes. Sin embargo, hay aplicaciones inconsistentes escritas con HTML, CSS, DOM y otras especificaciones estándar para navegadores web que pueden causar problemas en el desarrollo y soporte de estas aplicaciones, principalmente debido a la falta de adhesión de los navegadores a dichos estándares web (especialmente versiones de Internet Explorer anteriores a la 7.0). Adicionalmente, la posibilidad de los usuarios de personalizar muchas de las características de la interfaz (tamaño y color de fuentes, tipos de fuentes, inhabilitar Javascript) puede interferir con la consistencia de la aplicación web. Otra aproximación es utilizar Adobe Flash Player o Java applets para desarrollar parte o toda la interfaz de usuario. Como casi todos los navegadores incluyen soporte para estas tecnologías (usualmente por medio de plug-ins), las aplicaciones basadas en Flash o Java pueden ser implementadas con aproximadamente la misma facilidad. Dado que ignoran las configuraciones de los navegadores, estas tecnologías permiten más control sobre la interfaz, aunque las incompatibilidades entre implementaciones Flash o Java puedan crear nuevas complicaciones, debido a que no son estándares. Por las similitudes con una arquitectura cliente-servidor, con un cliente "no ligero", existen discrepancias sobre el hecho de llamar a estos sistemas “aplicaciones web”; un término alternativo es “Aplicación Enriquecida de Internet”.

**4. Estructura de las Aplicaciones web**

Aunque existen muchas variaciones posibles, una aplicación web está normalmente estructurada como una aplicación de tres-capas. En su forma más común, el navegador web ofrece la primera capa, y un motor capaz de usar alguna tecnología web dinámica (ejemplo: PHP, Java Servlets o ASP, ASP.NET, CGI, ColdFusion, embPerl, Python (programming language) o Ruby on Rails) que constituye la capa intermedia. Por último, una base de datos constituye la tercera y última capa.

El navegador web manda peticiones a la capa intermedia que ofrece servicios valiéndose de consultas y actualizaciones a la base de datos y a su vez proporciona una interfaz de usuario.

**5. Ventajas**

Una de las [ventajas](http://www.alegsa.com.ar/Dic/aplicacion%20web.php) de las aplicaciones web cargadas desde internet (u otra red) es la facilidad de mantener y actualizar dichas aplicaciones sin la necesidad de distribuir e instalar un software en, potencialmente, miles de clientes. También la posibilidad de ser ejecutadas en múltiples [plataformas](http://www.alegsa.com.ar/Dic/plataforma.php).

- Ahorra tiempo: Se pueden realizar tareas sencillas sin necesidad de descargar ni instalar ningún programa.

- No hay problemas de compatibilidad: Basta tener un navegador actualizado para poder utilizarlas.

- No ocupan espacio en nuestro disco duro.

- Actualizaciones inmediatas: Como el software lo gestiona el propio desarrollador, cuando nos conectamos estamos usando siempre la última versión que haya lanzado.

- Consumo de recursos bajo: Dado que toda (o gran parte) de la aplicación no se encuentra en nuestro ordenador, muchas de las tareas que realiza el software no consumen recursos nuestros porque se realizan desde otro ordenador.

- Multiplataforma: Se pueden usar desde cualquier sistema operativo porque sólo es necesario tener un navegador.

- Portables: Es independiente del ordenador donde se utilice (un PC de sobremesa, un portátil...) porque se accede a través de una página web (sólo es necesario disponer de acceso a Internet). La reciente tendencia al acceso a las aplicaciones web a través de teléfonos móviles requiere sin embargo un diseño específico de los ficheros CSS para no dificultar el acceso de estos usuarios.

- La disponibilidad suele ser alta porque el servicio se ofrece desde múltiples localizaciones para asegurar la continuidad del mismo.

- Los virus no dañan los datos porque éstos están guardados en el servidor de la aplicación.

- Colaboración: Gracias a que el acceso al servicio se realiza desde una única ubicación es sencillo el acceso y compartición de datos por parte de varios

usuarios. Tiene mucho sentido, por ejemplo, en aplicaciones online de calendarios

u oficina.

- Los navegadores ofrecen cada vez más y mejores funcionalidades para crear aplicaciones web ricas (RIAs).

**6. Servidor web**

Imagínense a los **servidores web como computadoras especialmente preparadas para estar prendidas las 24 horas**, los 365 días del año. La principal función de un servidor Web es almacenar los archivos de un sitio y emitirlos por Internet para poder ser visitado por los usuarios. Básicamente, un servidor Web es una gran computadora que guarda y transmite datos vía Internet. Cuando un usuario entra en una página de Internet su navegador se comunica con el servidor enviando y recibiendo datos que determinan qué es lo que ve en la pantalla. Por eso decimos que los servidores Web están para almacenar y transmitir datos de un sitio según lo que pida el navegador de un visitante. Cada servidor Web y cada computadora conectada a Internet tiene asignado una dirección de IP irrepetible que lo identifica en la red incluyendo la PC que tenés en tu casa. La dirección de IP vendría a ser como los datos del remitente en una carta postal. Cuando llegas a un sitio Web, se envía un pedido desde tu dirección de IP hacia la dirección IP del servidor. El servidor Web responde mandando datos a la dirección IP que los pide. Esto es lo que pasa cada vez que estamos navegando un sitio en Internet.

**Más del 90% de los sitios en Internet utilizan un servidor alquilado a través de una empresa de Web hosting.**

La capacidad de un servidor depende del tipo de servidor que sea y de los componentes que lo conforman. Sin los servidores Web la Internet tal como la conocemos, no existiría. Los servidores son como la columna vertebral de la estructura de Internet. La industria del Web hosting es simplemente la forma de alquilar esos espacios de memoria y administración de datos.

**Funcionamiento**

El Servidor web se ejecuta en un ordenador manteniéndose a la espera de peticiones por parte de un cliente (un navegador web) y que responde a estas peticiones adecuadamente, mediante una página web que se exhibirá en el navegador o mostrando el respectivo mensaje si se detectó algún error. A modo de ejemplo, al teclear [www.google.o](http://www.google.org)rg en nuestro navegador, éste realiza una petición HTTP al servidor de dicha dirección. El servidor responde al cliente enviando el código HTML de la página; el cliente, una vez recibido el código, lo interpreta y lo exhibe en pantalla. Como vemos con este ejemplo, el cliente es el encargado de interpretar el código HTML, es decir, de mostrar las fuentes, los colores y la disposición de los textos y objetos de la página; el servidor tan sólo se limita a transferir el código de la página sin llevar a cabo ninguna interpretación de la misma.

Además de la transferencia de código HTML, los Servidores web pueden entregar aplicaciones web. Éstas son porciones de código que se ejecutan cuando se realizan ciertas peticiones o respuestas HTTP.

Hay que distinguir entre:

 Aplicaciones en el lado del cliente: el cliente web es el encargado de ejecutarlas en la máquina del usuario. Son las aplicaciones tipo Java "applets" o Javascript: el servidor proporciona el código de las aplicaciones al cliente y éste, mediante el navegador, las ejecuta. Es necesario, por tanto, que el cliente disponga de un navegador con capacidad para ejecutar aplicaciones (también llamadas scripts). Comúnmente, los navegadores permiten ejecutar aplicaciones escritas en lenguaje javascript y java, aunque pueden añadirse más lenguajes mediante el uso de plugins.

 Aplicaciones en el lado del servidor: el servidor web ejecuta la aplicación; ésta, una vez ejecutada, genera cierto código HTML; el servidor toma este código recién creado y lo envía al cliente por medio del protocolo HTTP.

Las aplicaciones de servidor muchas veces suelen ser la mejor opción para realizar aplicaciones web. La razón es que, al ejecutarse ésta en el servidor y no en la máquina del cliente, éste no necesita ninguna capacidad añadida, como sí ocurre

en el caso de querer ejecutar aplicaciones javascript o java. Así pues, cualquier

cliente dotado de un navegador web básico puede utilizar este tipo de aplicaciones.

El hecho de que HTTP y HTML estén íntimamente ligados no debe dar lugar a confundir ambos términos. HTML es un lenguaje de marcas y HTTP es un "protocolo".

Aplicación del lado del Servidor

Una aplicación del lado del servidor es cualquier programa o conjunto de instrucciones diseñadas con la finalidad de que un Servidor Web las procese para realizar alguna acción. Las aplicaciones del lado del servidor están escritas

mediante algún lenguaje de programación, entre los que destacan:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Lenguaje** | **Año** | **Sistema Operativo** |  | **Versión** |
| PHP | 1995 | Multiplataforma | 5.3.5 |  |
| ASP.Net | 1998 | Windows (Algunas versiones) 4.0 | | |
| Perl | 1987 | Multiplataforma 5.12.3 | | |
| Python |  | 1991 Multiplataforma 3.2.0 | | |
| Ruby | 1995 | Multiplataforma 1.9.3-p125 | | |

El 75% de las aplicaciones del lado del servidor están escritas en PHP, seguido de ASP y las demás opciones usadas de forma alternativa y muy casual.

En 2009 Node.js fue creado por Ryan Dahl, abriendo públicamente la oportunidad de usar javascript del lado del servidor. Evento que revoluciona la web, y da inicio a una nueva era del desarrollo en internet. Node.js, dado a sus usuarios,

programadores web de todo el mundo, está creciendo como la mejor opción para

desarrollo de aplicaciones del lado del servidor, desplazando lenguajes

tradicionales en esta tarea. Node.js trabaja sobre el rápido motor V8 que usa

Google Chrome para la interpretación de javascript. Procesamiento del lado del servidor

Un servidor web tiene la función de procesar los scripts del lado del servidor para dar una salida en HTML y otros lenguajes del lado del cliente al Navegador Web del cliente. La información a procesar podrá ser cedida por el cliente al script mediante cualquier aplicación en el entorno del Navegador. Para ello pueden utilizarse formularios web, enlaces con los valores implícitos en la cadena o cualquier otro método.

**7. Lenguaje de marcado de Hipertexto**

Es un estándar a cargo de la W3C, organización dedicada a la estandarización de casi todas las tecnologías ligadas a la web, sobre todo en lo referente a su escritura e interpretación.

El lenguaje HTML basa su filosofía de desarrollo en la referenciación. Para añadir un elemento externo a la página (imagen, vídeo, script, etc.), este no se incrusta directamente en el código de la página, sino que se hace una referencia a la ubicación de dicho elemento mediante texto. De este modo, la página web contiene sólo texto mientras que recae en el navegador web (interpretador del código) la tarea de unir todos los elementos y visualizar la página final. Al ser un estándar, HTML busca ser un lenguaje que permita que cualquier página web escrita en una determinada versión, pueda ser interpretada de la misma forma (estándar) por cualquier navegador web actualizado.

Sin embargo, a lo largo de sus diferentes versiones, se han incorporado y suprimido características, con el fin de hacerlo más eficiente y facilitar el desarrollo de páginas web compatibles con distintos navegadores y plataformas (PC de escritorio, portátiles, teléfonos inteligentes, tablets, etc.). Sin embargo, para interpretar correctamente una nueva versión de HTML, los desarrolladores de navegadores web deben incorporar estos cambios y el usuario debe ser capaz de usar la nueva versión del navegador con los cambios incorporados. Usualmente los cambios son aplicados mediante parches de actualización automática (Firefox, Chrome) u ofreciendo una nueva versión del navegador con todos los cambios incorporados, en un sitio web de descarga oficial (Internet Explorer). Un navegador no actualizado no será capaz de interpretar correctamente una página web escrita en una versión de HTML superior a la que pueda interpretar, lo que obliga muchas veces a los desarrolladores a aplicar técnicas y cambios que

permitan corregir problemas de visualización e incluso de interpretación de código HTML. Así mismo, las páginas escritas en una versión anterior de HTML deberían ser actualizadas o reescritas, lo que no siempre se cumple. Es por ello que ciertos navegadores aún mantienen la capacidad de interpretar páginas web de versiones HTML anteriores. Por estas razones, aún existen diferencias entre distintos navegadores y versiones al interpretar una misma página web.

**II. COMO SE DESARROLLA UNA APLICACIÓN WEB**

**1. ¿Qué es el diseño y desarrollo de las aplicaciones web?**

El diseño y desarrollo de aplicaciones web consiste en implementar sus necesidades, objetivos o ideas en Internet utilizando las tecnologías más idóneas según su proyecto.

**2. Framework**

La palabra inglesa "**framework**" (marco de trabajo) define, en términos generales, un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar.

En el desarrollo de software, un **framework** o **infraestructura digital**, es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de software concretos, que puede servir de base para la organización y desarrollo de software. Típicamente, puede incluir soporte de programas, bibliotecas, y un lenguaje interpretado, entre otras herramientas, para así ayudar a desarrollar y unir los diferentes componentes de un proyecto. Representa una arquitectura de software que modela las relaciones generales de las entidades del dominio, y provee una estructura y una especial metodología de trabajo, la cual extiende o utiliza las aplicaciones del dominio.

Son diseñados con la intención de facilitar el desarrollo de software, permitiendo a los diseñadores y programadores pasar más tiempo identificando requerimientos de software que tratando con los tediosos detalles de bajo nivel de proveer un sistema funcional. Por ejemplo, un equipo que usa Apache Struts para desarrollar un sitio web de un banco, puede enfocarse en cómo los retiros de ahorros van a

funcionar en lugar de preocuparse de cómo se controla la navegación entre las páginas en una forma libre de errores. Sin embargo, hay quejas comunes acerca de que el uso de frameworks añade código innecesario y que la preponderancia de frameworks competitivos y complementarios significa que el tiempo que se pasaba programando y diseñando ahora se gasta en aprender a usar los frameworks.

Fuera de las aplicaciones en la informática, puede ser considerado como el conjunto de procesos y tecnologías usados para resolver un problema complejo. Es el esqueleto sobre el cual varios objetos son integrados para facilitar una solución dada.

**8. Content management system**

Es un sistema que consiste en una interfaz que controla una o varias bases de datos donde se aloja el contenido del sitio web. Su función es manejar de manera independiente el contenido y el diseño. Así, es posible manejar el contenido y darle en cualquier momento un diseño distinto al sitio web sin tener que darle formato al contenido de nuevo, además de permitir la fácil y controlada publicación en el sitio a varios editores. Un ejemplo clásico es el de editores que cargan el contenido al sistema y otro de nivel superior (moderador o administrador) que permite que estos contenidos sean visibles a todo el público (los aprueba).

Esto permite gestionar, bajo un formato estandarizado, la información del servidor, reduciendo el tamaño de las páginas para descarga y reduciendo el coste de gestión del portal con respecto a un sitio web estático, en el que cada cambio de diseño debe ser realizado en todas las páginas web, de la misma forma que cada vez que se agrega contenido tiene que maquetarse una nueva página HTML y subirla al servidor web.

**Funcionamiento**

Un sistema de administración de contenidos siempre funciona en el servidor web en el que esté alojado el portal. El acceso al gestor se realiza generalmente a través del navegador web, y se puede requerir el uso de FTP para subir contenido. Cuando un usuario accede a una URL, se ejecuta en el servidor esa llamada, se selecciona el esquema gráfico y se introducen los datos que correspondan de la base de datos. La página se genera dinámicamente para ese usuario, el código HTML final se genera en esa llamada. Normalmente se predefinen en el gestor varios formatos de presentación de contenido para darle la flexibilidad a la hora de crear nuevos apartados e informaciones.

**Gestión de usuarios**

Dependiendo de la plataforma elegida se podrán escoger diferentes niveles de acceso para los usuarios; yendo desde el administrador del portal hasta el usuario sin permiso de edición, o creador de contenido. Dependiendo de la aplicación podrá haber varios permisos intermedios que permitan la edición del contenido, la supervisión y reedición del contenido de otros usuarios,etc.

El sistema de gestión de contenidos controla y ayuda a manejar cada paso de este proceso, incluyendo las labores técnicas de publicar los documentos a uno o más sitios. En muchos sitios con estos sistemas una sola persona hace el papel de creador y editor, como por ejemplo en los blogs personales.

**Ventajas y Oportunidades**

El gestor de contenidos facilita el acceso a la publicación de contenidos a un rango mayor de usuarios. Permite que sin conocimientos de programación ni maquetación cualquier usuario pueda añadir contenido en el portal web.

Además permite la gestión dinámica de usuarios y permisos, la colaboración de varios usuarios en el mismo trabajo, la interacción mediante herramientas de comunicación.

Los costes de gestión de la información son mucho menores ya que se elimina un eslabón de la cadena de publicación, el maquetador. La maquetación es hecha al inicio del proceso de implantación del gestor de contenidos.

La actualización, backup y reestructuración del portal son mucho más sencillas al tener todos los datos vitales del portal, los contenidos, en una base de datos estructurada en el servidor.

**9. Drupal**

**10. Desarrollo ágil de software**

El desarrollo ágil de software son métodos de ingeniería del software, basados en el desarrollo iterativo e incremental, donde los requerimientos y soluciones evolucionan mediante la colaboración de grupos autos organizados y multidisciplinarios. Existen muchos métodos de desarrollo ágil; la mayoría minimiza riesgos desarrollando software en lapsos cortos. El software desarrollado en una unidad de tiempo es llamado una iteración, la cual debe durar de una a cuatro semanas. Cada iteración del ciclo de vida incluye: planificación, análisis de requerimientos, diseño, codificación, revisión y documentación. Una iteración no debe agregar demasiada funcionalidad para justificar el lanzamiento del producto al mercado, sino que la meta es tener una «demo» (sin errores) al final de cada iteración. Al final de cada iteración el equipo vuelve a evaluar las prioridades del proyecto.

Los métodos ágiles enfatizan las comunicaciones cara a cara en vez de la documentación. La mayoría de los equipos ágiles están localizados en una simple oficina abierta, a veces llamadas "plataformas de lanzamiento" (bullpen en inglés). La oficina debe incluir revisores, escritores de documentación y ayuda, diseñadores de iteración y directores de proyecto. Los métodos ágiles también enfatizan que el software funcional es la primera medida del progreso. Combinado con la preferencia por las comunicaciones cara a cara, generalmente los métodos ágiles son criticados y tratados como "indisciplinados" por la falta de documentación técnica.

**Historia**

La definición moderna de desarrollo ágil de software evolucionó a mediados de la década de 1990 como partes de una reacción contra los métodos de “pico pizado”, muy estructurados y estrictos, extraídos del modelo de desarrollo en cascada. El proceso originado del uso del modelo en cascada era visto como burocrático, lento, degradante e inconsistente con las formas de desarrollo de software que realmente realizaban un trabajo eficiente.

Los métodos de desarrollo, ágiles e iterativos pueden ser vistos como un retroceso a las prácticas observadas en los primeros años del desarrollo de software (aunque en ese tiempo no había metodologías formales). Inicialmente, los métodos ágiles fueron llamados métodos de "peso liviano".

En el año 2001, miembros prominentes de la comunidad se reunieron en Snowbird, Utah, y adoptaron el nombre de "métodos ágiles". Poco después, algunas de estas personas formaron la "alianza ágil", una organización sin fines de lucro que promueve el desarrollo ágil de aplicaciones. Muchos métodos similares al ágil fueron creados antes del 2000.

**Características fundamentales**

Las características fundamentales del método son:

 Desarrollo iterativo e incremental: pequeñas mejoras, unas tras otras.

 Pruebas unitarias continuas, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión. Se aconseja escribir el código de la prueba

antes de la codificación. Véase, por ejemplo, las herramientas de prueba JUnit orientada a Java, DUnit orientada a Delphi, NUnit para la

plataforma.NET o PHPUnit para PHP. Estas tres últimas inspiradas en JUnit, la cual, a su vez, se insipiró en SUnit, el primer framework orientado a

realizar tests, realizado para el lenguaje de programación Smalltalk.

 Programación en parejas: se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto. La mayor calidad del código escrito de esta manera -el código es revisado y discutido mientras se escribe- es más importante que la posible pérdida de productividad inmediata.

 Frecuente integración del equipo de programación con el cliente o usuario.

Se recomienda que un representante del cliente trabaje junto al equipo de desarrollo.

 Corrección de todos los errores antes de añadir nueva funcionalidad. Hacer entregas frecuentes.

 Refactorización del código, es decir, reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad pero sin modificar su

comportamiento. Las pruebas han de garantizar que en la refactorización no se ha introducido ningún fallo.

 Propiedad del código compartida: en vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método

promueve el que todo el personal pueda corregir y extender cualquier parte del proyecto. Las frecuentes pruebas de regresión garantizan que los

posibles errores serán detectados.

 Simplicidad en el código: es la mejor manera de que las cosas funcionen.

Cuando todo funcione se podrá añadir funcionalidad si es necesario. La programación extrema apuesta que es más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar

algo complicado y quizás nunca utilizarlo.

La simplicidad y la comunicación son extraordinariamente complementarias. Con más comunicación resulta más fácil identificar qué se debe y qué no se debe hacer. Cuanto más simple es el sistema, menos tendrá que comunicar sobre éste, lo que lleva a una comunicación más completa, especialmente si se puede reducir el equipo de programadores.

**III. ENTORNO DE TRABAJO**

**1. Sistema operativo: Lubuntu**

Lubuntu es una distribución oficial del proyecto Ubuntu que tiene por lema "menos recursos y más eficiencia energética", usando el gestor de escritorio LXDE. El nombre Lubuntu es una combinación entre LXDE y Ubuntu.

El gestor LXDE usa el administrador de ventanas Openbox e intenta ser un sistema operativo que demande pocos recursos de RAM, CPU y otros componentes, especialmente ideados para equipos portátiles de recursos limitados como netbooks, dispositivos móviles y computadores antiguos. Con este ideal y esquema de desarrollo que posee, se presenta como una competencia a Xubuntu.

**Características:**

Los requerimientos de hardware de Lubuntu/LXDE son similares a los de

Windows XP. Un viejo CPU Pentium II o III entre 400-500 MHz es suficiente.

Una vez iniciado X.org y LXDE, el uso total de memoria aproximado es de 60 MB en máquinas i386. Este valor puede ser más alto o más bajo dependiendo de las diferentes configuraciones de sistema.

Aunque Lubuntu/LXDE no requiere hardware sofisticado, otras aplicaciones bajo

X.org sí pueden necesitarlo. Por ejemplo, Firefox y OpenOffice consumen niveles relativamente más altos de memoria. Así que es recomendable tener mínimo entre

192-256 MB de memoria RAM o más.

**2. Editor de texto: SublimeText2**

Multiplataforma. Sublime Text es un editor de texto y editor de código fuente creado en Python desarrollado originalmente como una extensión de Vim, con el tiempo fue creando una identidad propia, por esto aún conserva un modo de edición tipo vi llamado Vintage mode. Se distribuye de forma gratuita, sin embargo no es software libre o de código abierto,se puede obtener una licencia para su uso ilimitado, pero el no disponer de esta no genera ninguna limitación más allá de una alerta cada cierto tiempo.

**3. Sistema controlador de versiones: GIT**

El diseño de Git resulta de la experiencia del diseñador de Linux, Linus Torvalds, manteniendo una enorme cantidad de código distribuida y gestionada por mucha gente, que incide en numerosos detalles de rendimiento, y de la necesidad de rapidez en una primera implementación.

Entre las características más relevantes se encuentran:

Fuerte apoyo al desarrollo no-lineal, por ende rapidez en la gestión de ramas y mezclado de diferentes versiones. Git incluye herramientas específicas para navegar y visualizar un historial de desarrollo no-lineal. Una presunción fundamental en Git es que un cambio será fusionado mucho más frecuentemente de lo que se escribe originalmente, conforme se pasa entre varios programadores que lo revisan.

Gestión distribuida. Git le da a cada programador una copia local del historial del desarrollo entero, y los cambios se propagan entre los repositorios locales. Los cambios se importan como ramas adicionales y pueden ser fusionados en la misma manera que se hace con la rama local.

Los almacenes de información pueden publicarse por HTTP, FTP, rsync o mediante un protocolo nativo, ya sea a través de una conexión TCP/IP simple o a través de

cifrado SSH. Git también puede emular servidores CVS, lo que habilita el uso de clientes CVS pre-existentes y módulos IDE para CVS pre-existentes en el acceso de

repositorios Git.

**4. Computación en la nube: Heroku**

Heroku es una plataforma como servicio de computación en la Nube que soporta distintos lenguajes de programación.

Heroku es propiedad de Salesforce.com.Heroku, es una de las primeras plataformas de computación en la nube, que fue desarrollada desde junio de 2007, con el objetivo de soportar solamente el lenguaje de programación Ruby, pero posteriormente se ha extendido el soporte a Java, Node.js, Scala, Clojure y Python y (no documentado)PHP. La base del sistema operativo es Debian o, en la nueva plataforma, el sistema basado en Debian Ubuntu.

James Lindenbaum, Adam Wiggins, y Orion Henry fundaron Heroku para dar soporte a proyectos compatibles con Rack.El 8 de diciembre de 2010

Salesforce.com adquirió Heroku como una subsidiaria. El 12 de julio de

2011 Yukihiro "Matz" Matsumoto, el creador de Ruby se unió a la empresa como Arquitecto Jefe. Ese mismo mes, Heroku incorporó el soporte para Node.js y Clojure. El 15 de septiembre de 2011 Heroku y Facebook presentaron Heroku para Facebook

Actualmente Heroku soporta Cloudant, Couchbase Server, MongoDB y Redis, además de la conocida PostgreSQL, tanto como parte de la plataforma o como servicio independente.

Las aplicaciones se corren desde un servidor Heroku usando Heroku DNS Server para apuntar al dominio de la aplicación (típicamentenombreaplicacion.herokuapp.com). Cada aplicación corre sobre un motor a traves de una "red de bancos de prueba" que consta de varios servidores. El servidor Git de Heroku maneja los repositorios de las aplicaciones que son subidas por los usuarios

**2.6 TERMINO-DEFINICION**

**Framework:** La palabra inglesa "**framework**" (marco de trabajo) define, en términos generales, un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar

**HTML:** siglas del inglés HyperText Markup Language, "lenguaje de etiquetado de hipertexto": es el lenguaje que se utiliza para crear las páginas Web, un lenguaje muy sencillo que permite combinar gráficos, textos y enlaces

**Http:** siglas del inglés HyperText Transfer Protocol, "protocolo de transferencia de hipertexto": es el protocolo de comunicación que utiliza la WWW.

**Interfaz:** sistema de comunicación de un programa con su usuario; la interfaz comprende las pantallas y los elementos que informan al usuario sobre lo que puede hacer, o sobre lo que está ocurriendo.

**Internet:** conjunto de ordenadores, o servidores, conectados en una red de redes mundial, que comparten un mismo protocolo de comunicación, y que prestan servicio a los ordenadores que se conectan a esa red; debe decirse siempre "la Internet".

**Link:** palabra inglesa para enlace; hay quien usa incluso linkar, porque no se ha enterado de que existe "enlazar"

**Sitio Web:** traducción del inglés Web site, conjunto de páginas de una institución o persona; también se dice sólo sitio: ("he visitado el sitio de Sí, Spain", "¿has estado en mi sitio?")

**W3:** forma abreviada de referirse a la World Wide Web

**WWW:** forma abreviada de referirse a la World Wide Web

**Web** (siempre en femenino: "la Web") 1 forma abreviada de referirse a la World Wide Web. 2...hay quien usa esta palabra --bastante horrendamente-- como abreviatura de "página Web", por ejemplo: "¿te gusta mi web?"

**World Wide Web:** (en español "Telaraña Mundial") interfaz de comunicación en la Internet, que hace uso de enlaces de hipertexto en el interior de una misma página, o entre distintas páginas.

**CMS:** Un sistema de gestión de contenidos (o CMS, del inglés Content Management System) es un programa que permite crear una estructura de soporte (framework) para la creación y administración de contenidos, principalmente en páginas web, por parte de los administradores, editores, participantes y demás roles.

**Git:** Git es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando estas tienen un gran número de archivos de código fuente. Al principio, Git se pensó como un motor de bajo nivel sobre el cual otros pudieran escribir la interfaz de usuario o front end como Cogito o StGIT. Sin embargo, Git se ha convertido desde entonces en un sistema de control de versiones con funcionalidad plena. Hay algunos proyectos de mucha relevancia que ya usan Git, en particular, el grupo de programación del núcleo Linux.

**Petición GET:** Un servidor web opera mediante el protocolo HTTP, de la capa de aplicación del Modelo OSI. Al protocolo HTTP se le asigna habitualmente el puerto TCP 80. Las peticiones al servidor suelen realizarse mediante HTTP utilizando el método de petición GET en el que el recurso se solicita a través de la url al servidor web.

Por ejemplo para: GET /index.html HTTP/1.1 HOST: [www.host.co](http://www.host.com)m. En la barra de URL de un navegador cualquiera la petición anterior sería análoga a la siguiente dirección Web: [www.host.com/index.](http://www.host.com/index.html)html

**Petición POST:** Es el segundo tipo de petición HTTP más utilizado. Los datos a enviar al servidor se incluyen en el cuerpo de la misma petición con las cabeceras HTTP asignadas correspondientemente respecto al tipo de petición. Generalmente se asocia con los formularios web en el que los datos suelen ser cifrados para enviarlos de manera segura al servidor.

El navegador recopila la información del formulario para crear la petición y enviarla. Las cabeceras las envía junto a la petición POST, y se recopilan en base a metatags definidos en el código, los automáticos del navegador y los Cookies. Es el navegador, también, el que codifica los datos si es necesario.

**Petición WEB:** El navegador por medio de la interfaz de usuario permite al usuario realizar una o varias peticiones web. La interfaz de usuario o entorno de usuario es el conjunto de elementos del navegador que permiten realizar la petición de forma activa. Una petición Web no sólo puede ser realizada mediante un navegador sino con cualquier herramienta habilitada para tal fin, como una consola de comandos Telnet