***Programación MODULO I***

¿Qué es la programación?

Es el proceso utilizado para idear y ordenar las acciones necesarias para realizar un proyecto, preparar ciertas máquinas o aparatos para que empiecen a funcionar en el momento y en la forma deseados o elaborar programas para su empleo en computadoras.

¿Qué es un algoritmo?

Un algoritmo informático es un conjunto de instrucciones definidas, ordenadas y acotadas para resolver un problema o realizar una tarea. En programación, supone el paso previo a ponerse a escribir el código. Primero debemos encontrar la solución al problema (definir el algoritmo informático), para luego, a través del código, poder indicarle a la máquina qué acciones queremos que lleve a cabo. De este modo, un programa informático no sería más que un conjunto de algoritmos ordenados y codificados en un lenguaje de programación para poder ser ejecutados en un ordenador.

En conclusión, primero se resuelve el problema mediante el algoritmo y luego se realiza la codificación de dicha resolución en un determinado lenguaje.

Habilidades digitales:

Keywords:

Consiste en el proceso de buscar e investigar de la manera mas eficiente dentro de internet, de este modo ahorramos tiempo e invertimos en eficiencia. A menor cantidad de resultados, mayor será la calidad y especificidad del contenido.

Analizar el problema; Desgranar sus términos; Sistemas de búsqueda avanzada; Ordenar en pasos lógicos la búsqueda.

Otros métodos:

-Las comillas: Buscan una frase textual. También sirve para encontrar errores de ortografía o el nombre de una persona.

-Conjunción: Se cumple los ambos conjuntos. La conjunción más común es la “y”. En el buscador se realiza de la siguiente forma: “información” +”anexo de esta información” +”otro anexo”.

-Negación y disyunción: Negación: Nos permite quitar un término de búsqueda que no deseemos. En el buscador se realiza de la siguiente forma: “información” +”anexo de esta información” +”otro anexo” –“otro anexo que no deseemos u estorbe en la búsqueda”. Disyunción (“o”): En el buscador se realiza de la siguiente forma: “información” |”anexo de esta información” OR”otro anexo”. La disyunción nos permite buscar, por ejemplo, mediante sinónimos.

Otros operadores de búsqueda:

El (), permite combinar operadores.

DEFINE, define el significado de la palabra que quisiéramos.

STOCKS, muestra el estado actual de una empresa que busquemos en la bolsa.

Herramientas de desarrolladores:

*Devtools:*

Todos los navegadores web modernos incluyen un potente conjunto de herramientas para desarrolladores. Estas herramientas hacen una variedad de cosas, desde inspeccionar HTML, CSS y JavaScript actualmente cargados, hasta mostrar qué archivos ha solicitado la página y cuánto tiempo tardaron en cargarse. Vamos a explicar cómo utilizar las funciones básicas de las herramientas de desarrollo del navegador. A esto lo denominamos devtools. Se pueden encontrar si buscamos en herramientas o CTRL+MAYUS+I.

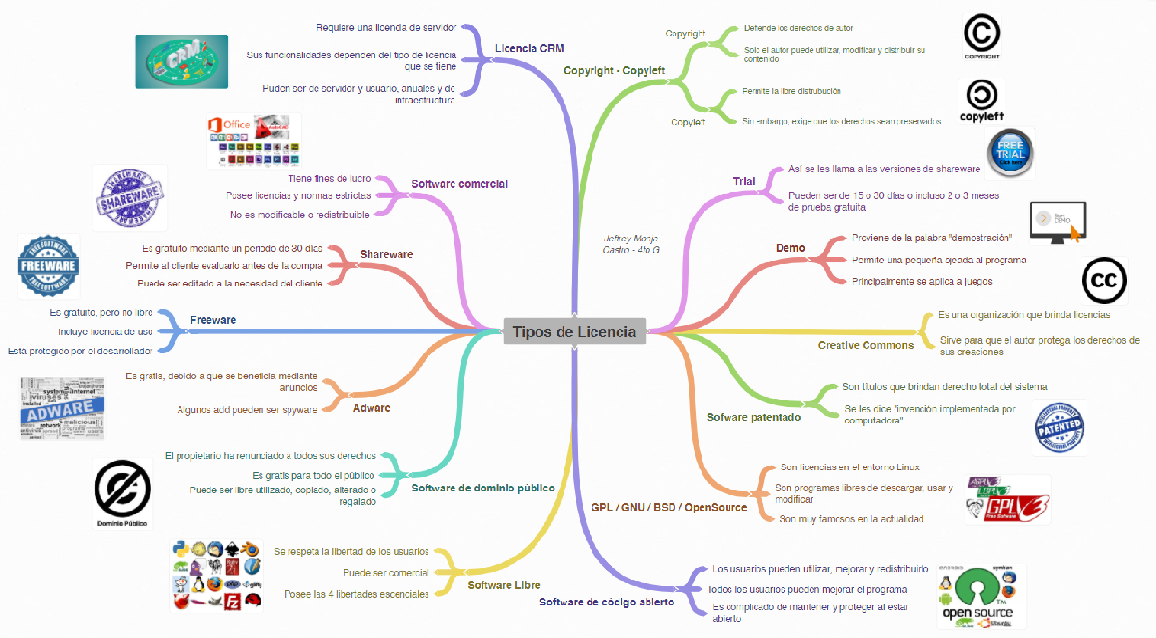
Más información de devtools:<https://developer.chrome.com/docs/devtools/overview/>

*Traductores:*

[*https://translate.google.com.ar/?hl=es*](https://translate.google.com.ar/?hl=es)

[*https://www.deepl.com/translator*](https://www.deepl.com/translator)

*Licencias de software:*



*¿Dónde realizar preguntas?*

<https://es.stackoverflow.com/>

Es recomendable preguntar en inglés y logearse.

Ideas de portafolios: <https://youtu.be/_YwKSdtWr0Q>

***ARQUITECTURA WEB***

Se trata de una arquitectura cliente-servidor en el que cada dispositivo conectado a la red actúa como cliente o servidor lo que implica la comunicación entre procesos que hacen peticiones, y procesos que responden a esas peticiones. Este proceso se logra gracias al protocolo HTTP.

Historia: A partir del desarrollo de ARPANET en 1969 empieza el descomunal crecimiento de internet. En 1990 se crea la WWW, World Wide Web, que permitió la primera conexión de un sitio web a un servidor con las 3 tecnologías fundamentales de la web: HTML, HTTP (protocolo servidor-cliente) y URL (localización del sitio web).

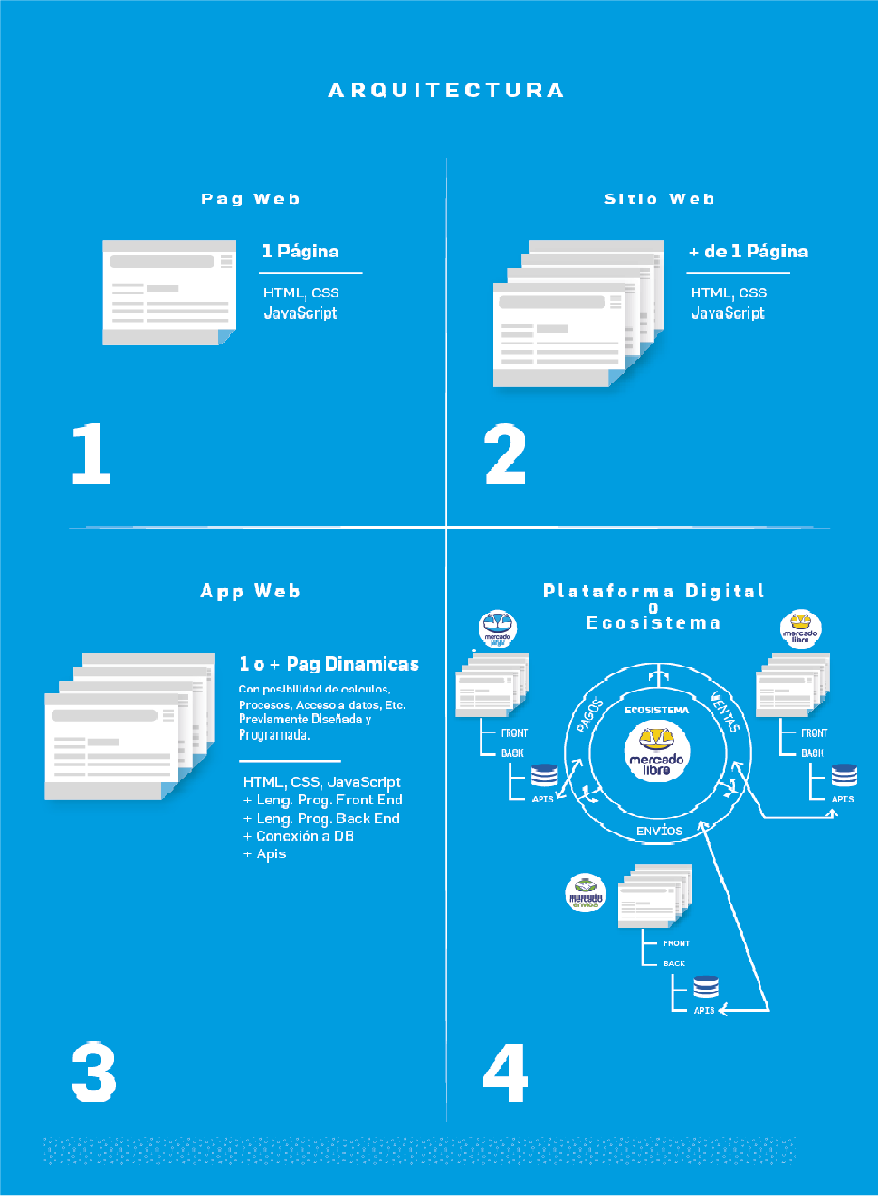
Arquitectura de las aplicaciones WEB:

*Variantes:*

* Servidor web + base de datos en un mismo servidor (De dos capas). En este caso el servidor gestiona la lógica del negocio, la lógica de los datos y los datos.
* Servidor web y base de datos separados (De tres capas). En este caso se separa la lógica de negocios de la lógica de datos en diferentes servidores.
* Servidor web + servidor de aplicaciones + servidor de datos. (Dos capas)
* Servidor de aplicaciones + servidor de datos en un mismo servidor. (Tres capas)
* Servidor de aplicaciones, bases de datos, y servidores de aplicaciones separados en diferentes servidores. (Cuatro capas)

El objetivo de separar las funciones en capas de diferentes servidores es buscar la mayor funcionabilidad y efectividad de los recursos, por ejemplo, un servidor web debe tener muy buena conexión a internet, mientras que un servidor de datos solo requiere gran nivel de almacenamiento.

Un grupo de páginas web dinámicas se conceptualiza como Front End (pensadas para el acceso y vista del usuario) y el otro grupo de paginas dinámicas web como Back End (pensadas para el procesamiento y acceso de datos), además la base de datos puede encontrarse en otro servidor. Esto se lo conoce como desacoplamiento, en el que el Front, el Back y las bases de datos pueden desacoplarse.



Análisis:

* **En el escenario 1:**Vemos una página web con contenido estático, es decir, no tiene conexión con ningún servidor, significa que no actualiza su información y solo se puede navegar dentro de la misma página. Para crearla se utilizó HTML, CSS y JavaScript
* **En el escenario 2:**Vemos una arquitectura centralizada (todo está ubicado en el mismo sitio), contiene varias páginas web con contenido estática, a esto le llamamos sitio web estático, como en el caso anterior tampoco tiene conexión con otro servidor lo que significa que solo se puede navegar entre las mismas páginas. Para crearla se utilizó HTML, CSS y JavaScript.
* **En el escenario 3:**Vemos una Aplicación Web o Web Dinámica, en este caso además de utilizar HTML, CSS y JavaScript se utilizaron lenguajes de programación para poder hacer intercambio de información con las distintas capas de la aplicación, hacer cálculos, crear nueva información en la base de datos, actualizarla o conectar con otros sistemas mediante API. Esta arquitectura tiene la siguiente separación conceptual o de arquitectura:

**Front End:**  La parte de la aplicación web que ve un usuario cuando entra desde el navegador a nuestra aplicación.

**Back End:** La parte de la aplicación web *que no se ve a simple vista, pero ejecuta acciones que pide el usuario desde el Front End*.

**Conexión a BD:**Es el nombre que se le da a la conexión con la base de datos, de esta manera facilita el almacenamiento y búsquedas de los datos.

**Apis:**Las Apis nos permiten conectemos con otros sistemas o bien que otros sistemas se conecten con el nuestro.

* **En el escenario 4:**Vemos una Plataforma Digital o un Ecosistema, en este caso podemos ver que consiste en muchos sistemas que trabajan en conjunto, colaborando para resolver una necesidad o problema. En el ejemplo de Mercado Libre se dedica a vender online, pero también tiene que cobrar y hacer envíos. Por eso han desarrollado aplicaciones independientes pero que saben cómo comunicarse a otros sistemas para pedir o enviar datos para realizar alguna tarea. De esta manera las aplicaciones pueden dar solución integral, comprar, pagar y enviar el producto sin tener que salir de la página.

Elementos de la arquitectura WEB:

• Infraestructura de red: Internet, intranet o extranet. Son los componentes físicos que permiten el funcionamiento de la red.

•ISP: Es el proveedor del servicio de internet.

•Cliente web: Es el navegador Web ejecutado sobre un hardware.

•Nombre dominio: Es el nombre de un sitio web, aparece luego de un @ o un www. Los dominios, son en palabras la IP que entiende la computadora. Es decir, cada sitio tiene una IP (Dirección de Protocolo de Internet) que es su dominio. Por ejemplo, nuestro nombre es el dominio y nuestro DNI la IP. Para registrar un dominio en el caso de Argentina, es mediante Nic.ar.

•URL (Localizador Uniforme de Recursos): Es una dirección completa, para encontrar una página web específica.

•DNS (Sistema de Nombre de Dominios): Realiza las conversiones de nombres de dominio, interpretados por las personas, a la IP interpretadas por las computadoras. Los DNS pertenecen a un servidor.

•Hosting: Los hostings son servidores constantemente encendidos y conectados a la red de internet, los cuales se encargan de alojar nuestro sitio u aplicación una vez finalizado.

•Servidor Web: O HTTP es un programa informático que se encarga del protocolo servidor-cliente.

•Servidores de aplicaciones WEB: Módulo que permite la ejecución de páginas WEB.

•Servidor de Bases de Datos: Son contenedores de datos que permiten organizar y administrar datos que deben ser almacenados de forma permanente u constante. Pueden almacenar o enviar datos de un o para un sitio web.

MODELO OSI: Modelo de Inter conexión de sistemas abiertos, Open Systems Interconection.

Arquitectura centralizada:

Es cuando tenemos todos los elementos de nuestra aplicación web de arquitectura de Cliente / Servidor en un solo lugar equipo o servidor, es decir, tener el Back End, Front End, Bases de datos y APIs en el mismo equipo. Esto hace que en el caso de una falla del equipo toda nuestra aplicación también fallará.

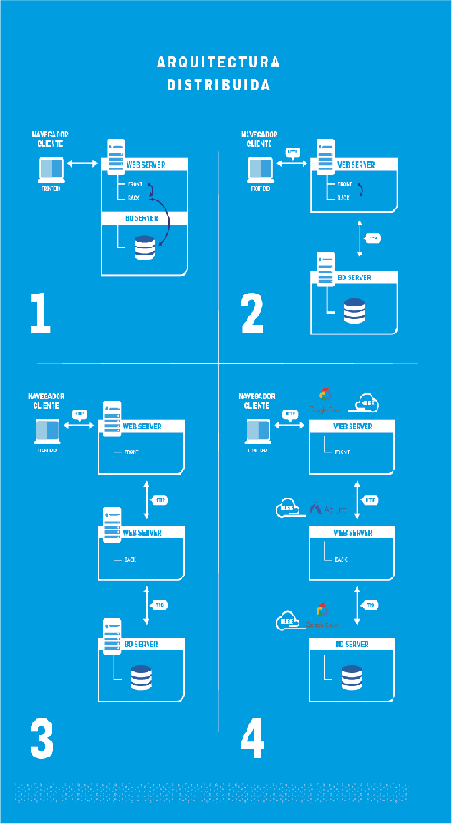
Arquitectura descentralizada o distribuida:

Es tener la posibilidad y capacidad de separar nuestro sistema en distintos servidores de la red (sea red local o internet).

Para saber si una arquitectura es distribuida desde el enfoque de programador, debemos tener en cuenta los siguientes aspectos:

1. ***Por la Líder del Proyecto***: Cuando se comienza a trabajar en un proyecto generalmente la líder del mismo hace una explicación del tipo de aplicación con la que se está trabajando, además de indicarnos en qué parte del proyecto estaremos trabajando.
2. ***Por el perfil asignado***: Cuando nos asignan el trabajo en una empresa nos especifican si trabajaremos en el Front End, en el Back End o en ambos Full Stack, de esa manera podemos inferir que la arquitectura es distribuida, igualmente siempre es mejor preguntar para estar seguros.
3. ***Por un diagrama*:** Generalmente se utilizan diagramas de aplicación para documentar un sistema, en el se puede ver la separación del sistema y si está distribuido en 1 o varios servidores.

Análisis de escenarios posibles de arquitectura distribuida:



* **Escenario 1:**En este caso podemos ver que la aplicación está separada en Front End, Back End y bases de datos están en el mismo servidor. Notemos que la aplicación fue diseñada de forma modular o separada (para poder distribuirla) todas las partes del sistema están en un mismo servidor, es decir, en caso de falla del servidor afecta a todo el sistema.
* **Escenario 2:**En este caso podemos ver que se ha separado la base de datos y el sistema sigue funcionando porque el desarrollador Back End escribió el código pensando en una arquitectura distribuida.  Pero la parte del Front End y Back End aún están en un mismo servidor.
* **Escenario 3:**  En este caso podemos ver que cada parte del sistema Front End, Back End y Base de datos está en un servidor diferente. Con esto comenzamos a ver los beneficios del diseño con arquitectura distribuida en los sistemas.
* **Escenario 4:** En este caso podemos ver que cada parte del sistema está en la nube de distintas empresas y nuestro sistema sigue funcionando por su diseño modular o distribuido.

Cuando la distribución del sistema no es suficiente, comenzamos a distribuir funcionalidades del sistema. Por ejemplo:

* **Por alta demanda**: Cuando el sistema tiene una funcionalidad que es compleja, consume mucho recurso del servidor o es muy demandada por distintas partes del sistema.
* **Por interconexión**: Cuando un sistema tiene funcionalidades que se necesita dar acceso a otros sistemas para consumir ese proceso, función o datos.
* **Por segregación de roles**: Cuando es necesario separar roles o funciones por motivos de seguridad o aspectos técnicos, también puede ser porque negocio lo requiere, por ejemplo, si se decide por seguridad separar el proceso de autenticación del sistema para reforzar la seguridad.
* **Por escalamiento:**  Cuando las proyecciones indican que en un periodo de tiempo la demanda aumentará considerablemente, será necesario agregar más servidores en la red con la misma funcionalidad para que satisfaga la demanda.

Para crear los esquemas de tu portfolio web te podemos recomendar algunas herramientas online gratuitas como [https://www.draw.io](https://www.draw.io/)  también [https://www.figma.com](https://www.figma.com/) te invitamos a que conozcas estas y crees tus esquemas con ellas.

Desarrollo web Full Stack:

Los desarrolladores web FS se encargan de desarrollar páginas webs dinámicas, es decir que modifican su contenido según el usuario que ha ingresado, teniendo en cuenta gustos, preferencias, actualizaciones, etc. Esto se logra cumpliendo con estándares web (ej. W3C) y el stack de desarrollo.

El stack: Es el conjunto de herramientas tecnológicas utilizadas para construir y ejecutar una aplicación, en nuestro caso web. Un ejemplo del stack tecnológico, puede ser los lenguajes implementados para el desarrollo de la aplicación. El stack se implementará dentro de frameworks.

**Front End:** Para el desarrollo del Front End se necesita.

* Estructura y estilos: HTML, CSS o framework como Bootstrap
* Lenguaje programación: JavaScript, java, PHP o bien pueden ser framework como Angular

**Back End:**Para el desarrollo de Back End se necesita un web server, una base de datos y un lenguaje de programación.

* Base de datos: puede ser MySQL, PostgreSQL, etc.
* Lenguaje: Java, PHP, etc.
* Web server: Apache, NGINX, etc.

Uno de los stacks o pila de tecnologías más utilizado por los desarrolladores es el que se conoce por **LAMP**: **L**inux, **A**pache, **M**ySQL y **P**HP. Cualquier web hecha con Wordpress, Drupal o Prestashop, por ejemplo, están hechas sobre estos cuatro pilares. Pero se pueden hacer las variaciones que se crean convenientes, puesto que muchas de estas tecnologías son intercambiables por otras similares. Por ejemplo, NginX en lugar de Apache, PostgreSQL en lugar de MySQL o Ruby on Rails en lugar de PHP.

Otro stack muy utilizado es el llamado **MEAN**, que se compone de **M**ongoDB, **E**xpress, **A**ngular y **N**odeJS. te dejamos una imagen ilustrativa del Stack MEAN.

Video explicativo:  <https://www.youtube.com/channel/UCX9NJ471o7Wie1DQe94RVIg>

En el curso:

**Para Front End:**

* Estructura y estilos: HTML, CSS y framework como Bootstrap
* Lenguaje programación: TypeScrypt y el framework  como Angular ambos basados en Javascript

**Para Back End:**

* Base de datos: Usaremos MySQL
* Lenguaje: Java con Framework Spring Boot
* Web server: Apache Tomcat

Modelo OSI: <http://www1.frm.utn.edu.ar/comunicaciones/modelo_osi.html>

Proceso de petición WEB:

1. Cliente Web: Solicita la resolución de nombres al servidor DNS. Por ejemplo: google.com
2. Servidor DNS: Recibe y trata la solicitud. Una vez recibida la petición realiza las consultas necesarias para resolver y obtener la dirección IP.
3. Servidor DNS: Devuelve al navegador Web la dirección IP que corresponde al Servidor Web.
4. Cliente Web: Conecta con el servidor web mediante la dirección IP y el puerto. Realiza la petición mediante una URL (Método GET) o un formulario (Método POST). Dicha solicitud incluye: la dirección IP del servidor web, el puerto del servidor web, URL y parámetros.
5. Servidor Web: Control de Acceso, Análisis de la petición y localización del recurso. Como detecta que es el acceso a un fichero o ruta de aplicación tiene que traspasar el control al Contenedor de aplicaciones Web
6. Paso de la petición del servidor web al contenedor de aplicaciones web
7. El contenedor analiza la petición y en base a la ruta traspasa el control a la aplicación web.
8. Paso del control de la petición desde el CAW a la aplicación.
9. La aplicación recibe la petición y decide qué hacer en base a ella, es decir, elegir la función que se encargará de gestionar esa petición, normalmente en base a la ruta, el método HTTP y los parámetros de entrada por URL. Una vez elegida ejecutará esa función.
10. La aplicación realiza una petición SQL a la base de datos.
11. La Base de Datos recibe la petición SQL y la procesa realizando los cambios que tenga que hacer, si corresponde.
12. Una vez procesada la petición devuelve los datos a la aplicación web, normalmente un conjunto de datos. Ej. los 10 últimos clientes.
13. La aplicación web recibe estos datos y tiene que generar una salida, normalmente HTML, donde estructura el contenido de los datos devueltos por la BBDD en etiquetas HTML.
14. La aplicación web devuelve una respuesta al Contenedor de Aplicaciones Web
15. El contenedor procesa la respuesta, para controlar la ejecución de la aplicación por si esta falla.
16. El Contenedor de Aplicaciones Web devuelve el fichero al servidor web.
17. El servidor Web devuelve los datos dentro de la respuesta HTTP al navegador web.
18. Cliente Web: Presenta (renderiza) el contenido HTML resultante.

Estructuras involucradas en la transferencia de datos entre computadoras:

Para comenzar vamos a definir *que “(...)* una red de computadoras, también llamada red de ordenadores, red de comunicación de datos o red informática, es un conjunto de equipos (nodos) y softwares conectados entre sí por medio de dispositivos físicos (cable) o inalámbricos (wifi) que envían y reciben impulsos eléctricos, ondas electromagnéticas o cualquier otro medio para el transporte de datos con la finalidad de compartir información recursos y ofrecer servicios (...)”*(Tanenbaum, 2003).*

* **Red del Hogar:** En la red del hogar podemos observar que entra el cable al modem wifi permitiendo que los equipos o nodos se conecten por la red inalámbrica, los dispositivos que se conectan  son notebook, cámaras, celulares, impresoras. También podemos ver que desde el modem sale un cable hacia una computadora de escritorio. Todo está conectado por la misma red.
* **Red Empresarial:** En la red empresarial entra un cable al modem principal que llega al primer y segundo edificio permitiendo conectar en red a los equipos o nodos  (notebook, pc de escritorio, impresora,  servidores). También observamos que desde el modem principal sale un cable de red que conecta con un modem wifi el cual permite conectar los equipos o nodos que tenga conexión a la red por medio de wifi.
* **Red Empresarial de Proveedores:** En la red de proveedor de internet podemos observar que distintas empresas se conectan entre sí formando una red en la que transmiten y reciben datos.

Objetivos de una red: El objetivo de las redes en general, consisten en "compartir recursos" entre computadoras de modo tal que los programas, datos y equipo estén disponibles para cualquier solicitud que los equipos de la red requieran, sin importar la ubicación o distancia física del recurso (equipo) y del usuario.

-Compartir recursos a grandes distancias.

-Asegurar la confiabilidad y disponibilidad de la información.

-Aumenta la transmisión de datos.

-Reducir costos tales como el procesamiento de información.

Funcionamiento de redes:

La comunicación por medio de una red se lleva a cabo en dos categorías diferentes: una capa denominada física y otra lógica.

Capa física:  Incluye todos los elementos de los que hace uso un equipo para comunicarse con otro equipo dentro de la red, como, por ejemplo, tarjetas de red, los cables, las antenas, etc.

Capa lógica: **Para formar una red se requieren elementos de hardware, software y protocolos**. Los elementos físicos se clasifican en dos grupos: los dispositivos de usuario final (llamados también Hosts) y los dispositivos de red. Entre los dispositivos de usuario final podemos enumerar computadoras, impresoras, escáneres, y demás elementos que brindan servicios directamente al usuario. Los segundos (dispositivos de red) son todos aquellos que conectan entre sí a los dispositivos de usuario finales posibilitando su intercomunicación.

Internet es un conjunto descentralizado de redes de comunicación interconectadas que utilizan la familia de protocolos [TCP / IP](https://www.avast.com/es-es/c-what-is-tcp-ip), lo cual garantiza que las redes físicas heterogéneas que la componen, constituyan una red lógica única de alcance mundial.

Protocolos de web: WWW

Protocolos de correos electrónicos: SMTP

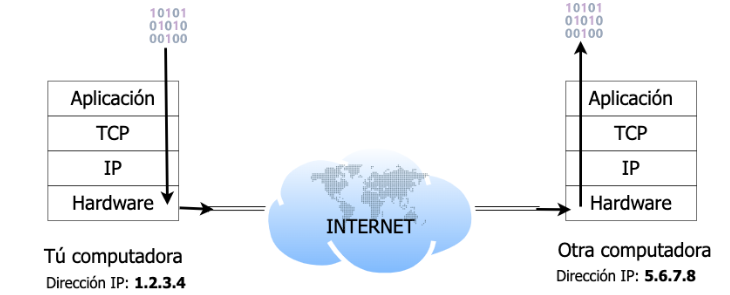
Protocolo para transmisión de archivos: FTP

TCP/IP: TCP (protocolo de control de transmisión) se usa para crear conexiones entre computadoras a través de las cuales pueden enviarse un flujo de datos. Por la forma en la que está implementado este protocolo los datos serán entregados en su destino sin errores y en el mismo orden en el que se transmitieron. ¿Esto qué quiere decir? que es un protocolo orientado a conexión, ya que el cliente y el servidor deben anunciarse y aceptar la conexión antes de comenzar a transmitir los datos entre ellos. Es decir que hay un intercambio de mensajes entre ellos para abrir una línea de conexión que permanece abierta durante toda la comunicación. Por otro lado, el protocolo IP es un protocolo cuya función principal es el uso direccional en origen o destino de comunicación para transmitir datos mediante un protocolo no orientado a conexión que transfiere paquetes conmutados a través de distintas redes previamente enlazadas según la norma OSI.

Entonces, en conclusión, se utiliza la combinación de estos dos protocolos para la comunicación en Internet, en donde TCP aporta la fiabilidad entre la comunicación e IP la comunicación entre distintas computadoras ya que las cabeceras de IP (cabecera por ser una parte el protocolo) contienen las direcciones de destino de las máquinas de origen y llamadas direcciones IP. Estas direcciones serán usadas por los routers para decidir el tramo de red por el que se enviarán los paquetes.

**Conclusión general: Dentro de la red de redes que es Internet debe existir un mecanismo para conectar dos computadoras. Este mecanismo lo proporciona el protocolo de Internet, el cual hace que un paquete de una computadora, llegue a la otra de manera segura a través del protocolo TCP y que llegue a destino a través de las direcciones IP.**

 Ya vimos que la estructura de red se maneja en capas. También mencionamos que hay una capa de red en dónde está el protocolo IP, una capa Superior de transporte en dónde está el protocolo TCP y ahora vemos una nueva capa que es la de aplicación en dónde se usa el protocolo HTTP.



HTTP (HyperText Transfer Protocol): HTTP define un conjunto de [métodos de **petición**](https://developer.mozilla.org/es/docs/Web/HTTP/Methods) para indicar la acción que desea realizar para un recurso determinado. Aunque estos también pueden ser sustantivos, estos métodos de solicitud a veces son llamados verbos HTTP. **Está orientado a transacciones y sigue el esquema de petición / respuesta entre un cliente y un servidor**. El cliente realiza una petición enviando un mensaje con cierto formato al servidor. El servidor le envía un mensaje de respuesta. Para hacerlo más concreto, un cliente podría ser un navegador web y un servidor podría ser una aplicación en un servidor web corriendo en Internet.

Es muy importante saber que HTTP es un protocolo sin estado, es decir, no guarda ninguna información sobre conexiones anteriores.}

Estructura del HTTP:

. - Primero, hay una línea inicial en donde se diferencian dependiendo de si son peticiones y respuestas. Para las solicitudes la línea comienza con una acción requerida por el servidor, a esto se le denomina método de petición seguido de la url del recurso y la versión http que soporte al cliente. Lo importante es el método de petición y la URL (Uniform Resource Locator o localizador de recursos uniforme). Para las respuestas, la línea comienza con la versión de HTTP seguido por un código de respuesta y con una frase asociada a dicho retorno.

. - También los mensajes tienen una cabecera que son metadatos con información diversa y el cuerpo de mensaje que es opcional. Típicamente este cuerpo tiene los datos que se intercambian entre el cliente y el servidor.

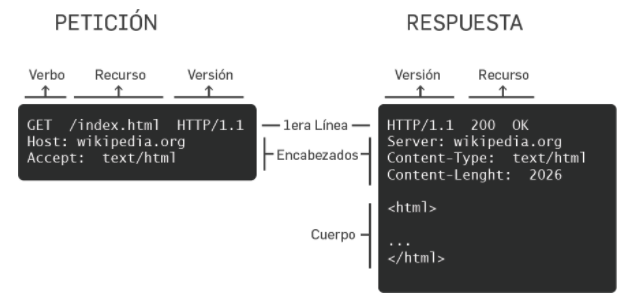
En el curso:

Los métodos que vamos a usar son los de **get** y **post**.

Get: solicita una representación del recurso especificado. Las solicitudes que usan **sólo deben recuperar datos** y no deben tener ningún otro efecto.

Post: **envía los datos para que sean procesados** ​​por el recurso identificado. Los datos enviados se incluirán en el de la petición. Esto puede resultar en la creación de un nuevo recurso o de las actualizaciones de los recursos existentes.

Vamos a ver que al presionar por ejemplo un enlace se hace una petición “get” al servidor para buscar otra página lo que resultará en una respuesta 200 si se encuentra la página o en un 404 si no se encuentra. También veremos que al llenar un formulario y enviarlo al servidor lo haremos a través de una petición post.



***GESTION DE TIEMPO***

*“El tiempo es un recurso escaso, limitado y finito por ello se debe gestionar. Se utiliza o se pierde, es tu elección y decisión que hacer con él”.*

Tiempo en función del objetivo:

•Tiempo laboral: El tiempo que invertimos o intercambiamos con otro por realizar un trabajo y obtener beneficios.

•Tiempo personal: Tiempo que decidimos dedicárselo a algo u a nosotros sin ningún tipo de barreras o compromisos.

•Tiempo familiar y social.

Aristóteles: *“Somos lo que hacemos día a día. De modo que la excelencia no es un acto, sino un hábito”.*

Impedimentos en la organización:

Déficit cognitivo: Dificultad en las funciones ejecutivas de la persona.

Rutinas incorrectas: Desbalanceada organización del tiempo, ya sea de forma física o mental.

Ciclos y hábitos:

Practica – Hábito – Rutina – Costumbre

En programación, esto sería un ciclo, es decir, es una sentencia que no se ejecuta hasta cumplir la condición.

Metodología de Organización del tiempo (GTD y TIMEBOXING):

Getting Things Done, es un método de gestión de las actividades. Se basa en que las personas deben liberar su mente de las tareas a realizar guardándolas en un lugar específico. De este modo no hay que estar pensando en que hay que hacer y se puede concentrar en las tareas que se realizan cuando corresponda. Es importante enlistar las tareas, ponerles una fecha, hora y prioridad.

TIME BOXING

Es una técnica de organización del tiempo orientada a objetivos y enfocada a la productividad evitando la procrastinación.

Cuando se fija un bloque de tiempo, se establece una meta a cumplir, realizar esa tarea dentro de la franja horaria que la delimita. En vez de preocuparse por menores, buscar aprobaciones, o documentos, el saber cuanto tiempo nos llevara la tarea nos permitirá utilizar eficientemente nuestro bloque de tiempo.

7 consejos del TB:

* 1. Priorizar los bloques de tiempo establecidos.
  2. Visualiza tu tiempo para entenderlo mejor.
  3. Configura un temporizador, del tiempo que hemos delimitado.
  4. Cuando comience un bloque que requiera una determinada concentración, desactiva las notificaciones. “No molestar”.
  5. Toma descansos entre cada bloque.
  6. Captura todo tu trabajo en una herramienta de gestión del trabajo.
  7. Agrupa las tareas similares.

Otros consejos:

-Definir objetivo y su consecuencia. ¿La tarea a realizar me brindara beneficios?

-Determinar cuando es el momento adecuado en la semana o en el día para realizar la tarea.

-Fija que cosas tienes que hacer en ese tiempo, cuales son las subtareas que conlleva la tarea en sí. Por ejemplo, si debo estudiar, cuales son los temas, que tan extensos son, etc.

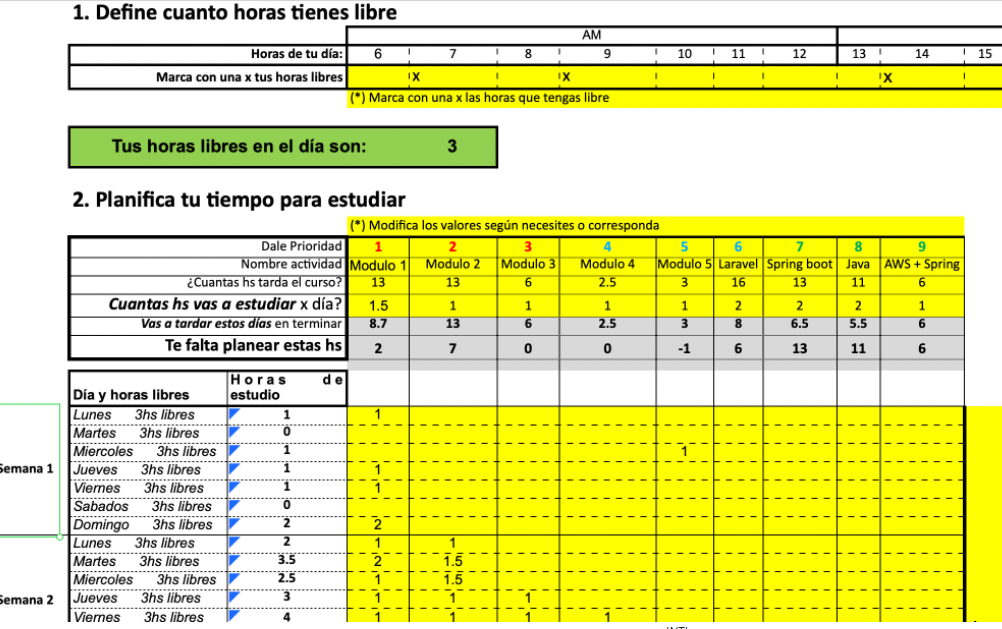
-Respetar el box, realizar las tareas indicadas en su tiempo y forma.

-En la práctica, debemos reconocer si nos alcanzo el tiempo asignado y realizar ajustes, por ejemplo, aumentar o disminuir el tiempo, saber si realmente es necesario mas o menos tiempo para la tarea a realizar.

-Es importante determinar cuanto tiempo tengo realmente, que tareas son fijas y las tareas que agreguemos delimitarlas por tiempo y prioridad.

-Es conveniente también, dividir las tareas por grupos.

REALIZAR EL EXCEL. 1er DESAFIO.



Eficiencia vs Eficacia

Eficiencia: Hacer las cosas bien independientemente del resultado. Se aprovechan los recursos de la mejor manera. En el caso que evaluamos el recurso primordial es el tiempo.

Eficacia: Consta de el resultado final sin importar el método o el proceso, es decir, se busca la aprobación sin tener en cuenta como se llego a ella. Por lo que, podemos ser eficaces, sin ser eficientes. Pero a su vez, no solo debemos ser eficientes, si no que el resultado debe tener un plus extra, y eso depende de si somos eficaces.

La ley de Parkinson: “El trabajo se expande hasta llenar el tiempo disponible para que se termine”.

Las ideas claves de la ley son: El trabajo (lo que mencionamos anteriormente), el tiempo (el tiempo que se le dedican a las diferentes tareas del día, no es proporcional al nivel de importancia de las mismas) y el gasto (el gasto aumentara hasta cubrir la totalidad de los ingresos).

Para cumplir con la ley, lo primordial es tener en cuenta un objetivo fijo para que la tarea no se dilate con el tiempo, por otra parte, las tareas que pueden realizarse deben hacerse sin planificación, de lo contrario seremos mas propensos a procrastinar o posponerlas generándonos problemas a futuro.

Por otra parte, es importante conocer nuestras horas productivas y sacarles el máximo provecho para mejorar nuestro potencial.

Principios de la eficacia:

Principio 1: Se alargan las tareas al interrumpirlas, pero el tiempo no.

Principio 2: Principio de planificación.

Principio 3: Priorización de tareas. Fijar prioridades claras, como metas, que se encaminen hacia el objetivo principal.

Principio 4: Incluir recompensas.

Principio 5: Principio de secuencias, la administración eficaz del tiempo te permitirá manejar la secuencia de eventos.

Modelos de estímulo:

Según los estímulos que recibamos, podemos ser reactivos o proactivos, en el primer caso no respondemos ante el estimulo y en el segundo brindamos una respuesta.

Urgente vs Importante

Hay que saber diferenciar entre lo urgente y lo importante, ya que muchas veces lo urgente no es lo importante. Evitar la tiranía de lo urgente para enfocarse en lo importante.

Tareas urgentes e importantes: Son tareas prioritarias porque se unen al objetivo a cumplir, y por diversas circunstancias no pueden postergarse más.

Tareas urgentes y no importantes: Se deben realizar ya, pero no tienen valor en cuanto a los objetivos principales.

Tareas no urgentes e importantes: Son del objetivo, pero se han podido planificar de manera correcta.

Tareas no urgentes y no importantes: No son importantes para el objetivo, ni apremiantes.

Consejos para ser alumno de alto rendimiento:

Distribución del tiempo: Emplean el estudio fomentando el hábito, dividen el tiempo de ocio del de estudio. Por ejemplo, dividir entre días que corresponden al ocio y otros al estudio.

Planificación de tareas: Realizan una previsión de las tareas y una jerarquización de las mismas. Organizan todas las tareas para irlas realizando poco a poco según su importancia.

Herramientas: Incluyen recursos tecnológicos, pero también acuden a una libreta.

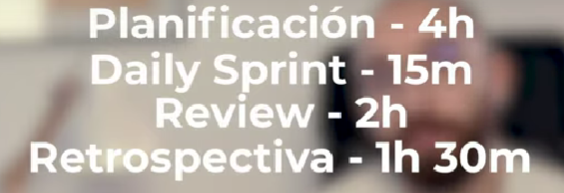
Valoran la administración del tiempo.

**SCRUM.**

Scrum es un marco de trabajo para desarrollo ágil de software que se ha expandido a otras industrias. Es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo y obtener el mejor resultado posible de proyectos.

El scrum permite promover innovación continua de forma incremental dentro de un equipo de trabajo. Debemos comprender que los proyectos varían según las necesidades y posibles problemas que surjan, por lo que nos permitirá realizar cambios a medida que el proyecto avanza.

El scrum esta diseñado para pequeños grupos, los cuales desgranaran el proyecto en diferentes partes, con sus respetivos objetivos y plazos a cumplir, conocidos como sprints. Para organízalos se deben realizar daily sprints con el equipo de trabajo. Luego de esto se realizarán reuniones de planificación, para definir los objetivos de los siguientes sprints. Mediante las review se analizan los anteriores sprints, estos nos permitirán analizar el progreso y encontrar posibles errores para evitar a futuro. Dentro de las review se mostrarán la demo, que permiten observar el proyecto, darle visibilidad. Al finalizar las reuniones se hacen las retrospectivas, que permiten analizar el flujo de trabajo del equipo.



Partes del equipo para llevar a cabo el scrum:

Product Owner: Es la voz de los clientes, y por lo tanto el responsable de el resultado de los negocios. Debe definir las características del producto y los porqués del mismo.

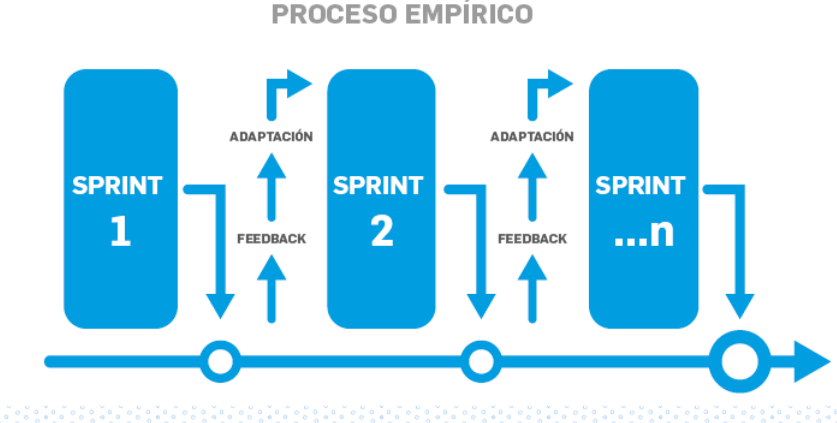
Scrum Master: Dirige las reuniones. Protege al equipo de interferencias externas. Facilita el trabajo, pero no dicta como debe ser realizado.

Dev Team: Es el propio equipo de desarrollo en sí, que trabaja en conjunto para la elaboración del producto final.

El software:

Es un producto intangible de alto conocimiento intelectual, que no sufre desgaste alguno y que puede ser potencialmente modificado permanentemente. No se manufactura si no que son desarrollados mediante proyectos de equipos altamente especializados. Es la parte del sistema que se puede codificar para ejecutarse en una computadora como un conjunto de instrucciones.

Los procesos pueden ser secuenciales, entrada, proceso y salida u empíricos:



Metodología en cascada:

Etapas definidas de orden secuencial, en la que cada etapa se inicia una vez finalizada y revisada la etapa anterior.

Sus etapas:

* 1. Análisis y definición de los requerimientos.
  2. Diseño del sistema de software.
  3. Implementación y pruebas unitarias.
  4. Integración y pruebas de sistema.
  5. Funcionamiento y mantenimiento.

Metodología iterativa e incremental:

Se diferencia del modelo en cascada ya que se admiten etapas que se solapan con el tiempo con la finalidad de flexibilizar el tiempo de desarrollo total y así alcanzar mejores resultados y antes de tiempo.

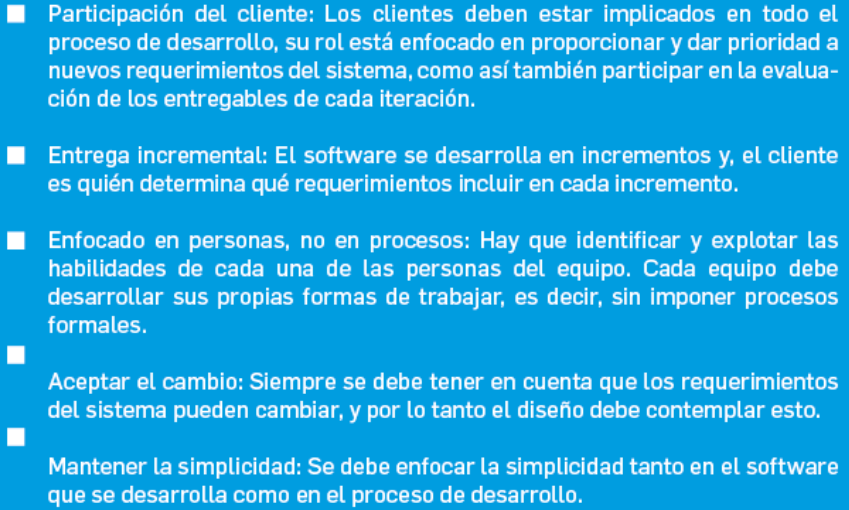
Incremental: Se busca desarrollar una parte del producto que se pueda integrar al conjunto a medida que se alcanza un cierto grado de completitud.

Iterativa: Se realizan ciclos donde se revisa y mejora el producto, mejorando la calidad del mismo.



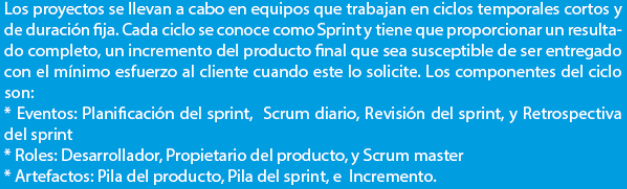
Procesos Agiles:

Los procesos o métodos agiles se consideran un enfoque iterativo para las etapas de especificación, desarrollo y entrega del software.



**Scrum:**

Es un marco de trabajo en el cual se aplican un conjunto de buenas practicas para trabajar de manera colaborativa y obtener así el mejor resultado posible en proyectos complejos que demanden constante adaptaciones y cambios.



\*Planificación variables: Alcance, estimación e importancia.

Alcance e Importancia – Propietario

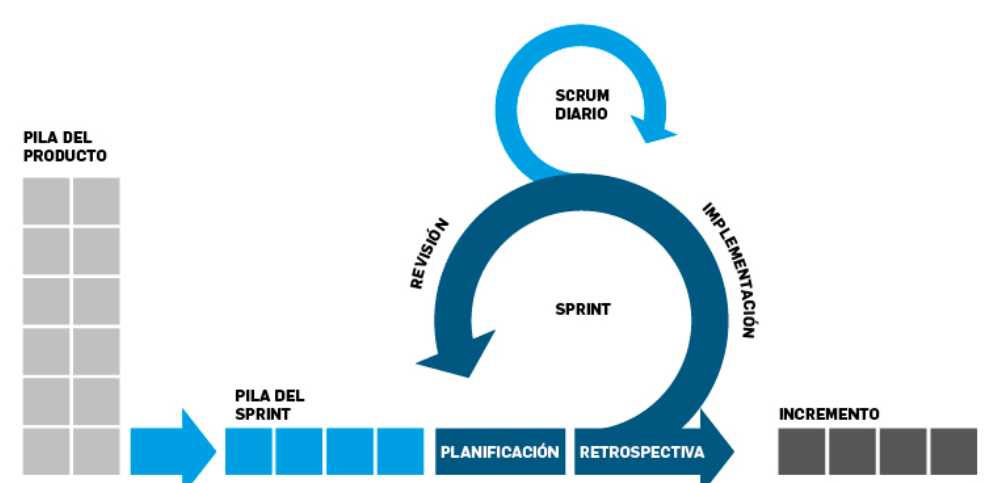
Estimación – Equipo

Sus pilares:

Transparencia: Debe existir una completa comunicación y entendimiento entre los participantes, el proyecto debe ser visible para todas las partes que trabajan en él.

Inspección: Se deben realizar diversas inspecciones en el proyecto pero que las mismas no estorben en el funcionamiento y progreso del mismo.

Adaptación: Durante el transcurso del proyecto las partes deben ser capaces de adaptarse a los posibles contratiempos y fallas que puedan surgir, y que en algunos casos, no puedan solucionarse.



**Eventos**:

•Planificación del sprint: Organización del trabajo a realizarse. En este sprint el propietario del producto explica el valor y utilidad del mismo. En este sprint el objetivo del proyecto y el terminado, deben ser definidos.

•Revisión de sprint: Son retrospectivas, de anteriores sprints, con la intención de mejorar la calidad y aprender de lo realizado para evitar futuras complicaciones.

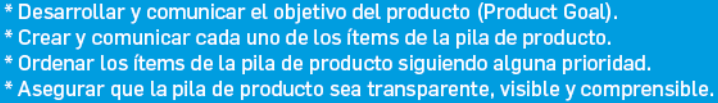
•Scrum diario: Son reuniones de nomas de 15m en donde se determina las tareas y objetivos a realizar en ese día puntual.

**Roles y Equipo (-10)**:

•Desarrolladores (developers): Son aquellos que producen cada incremento en los sptrints, son responsables de: Crear un plan para el sprint; Adoptar una definición de terminado; Adaptar el plan de cada día para alcanzar el objetivo del sprint.

•Scrum Master: Es el encargado de asegurar que la metodología se lleve a cabo tal y como esta definida. Esto se logra haciendo que cada persona del scrum entienda la teoría y la practica de la metodología. Dirige las reuniones. Protege al equipo de interferencias externas. Facilita el trabajo, pero no dicta como debe ser realizado.

•Product Owner: Es la voz de los clientes, y por lo tanto el responsable del resultado de los negocios. Debe definir las características del producto y los porqués del mismo. Es una sola persona y sus decisiones deben ser respetadas.



**Artefactos:**

•Pila de producto: Es una lista de lo que se necesita para mejorar el producto.

•Pila de sprint: Delimita el trabajo necesario a realizar para alcanzar un determinado incremento, y sirve para marcar el avance.

•Incremento: Es un paso concreto que acerca el desarrollo al objetivo del producto. El incremento debe ir de acuerdo con la definición de terminado para ser una suma al proyecto.

**Especificaciones y requerimientos**

La etapa de descubrir, analizar, documentar y verificar los servicios y restricciones del sistema, se la denomina ingeniería de requerimientos. Requierements Engineering.

Requerimientos Funcionales: Describen como debe reaccionar el sistema a entradas particulares o como debe comportarse bajo determinadas condiciones. Por ello debe ser completa, todos los servicios que debe proporcionar el sistema y consistente, por lo que los requerimientos no deben ser contradictorios.

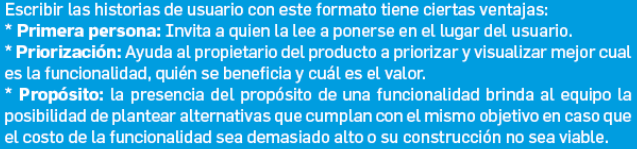
Requerimientos No Funcionales: Son aquellos requerimientos que refieren a propiedades del sistema, como fiabilidad, capacidad de almacenamiento, tiempos de respuesta, etc.

Requerimientos en Scrum:

Dentro de las metodologías agiles se suelen utilizar las historias de usuario como herramienta para definir los requerimientos que el sistema debe poseer. Estas son descripciones o especificaciones cortas de una funcionalidad validada por un usurario o cliente del sistema.

Las historias de usuario conforman la parte central de muchas metodologías de desarrollo ágil, tales como XP; Estas definen lo que se debe construir en el proyecto de software, tienen una prioridad asociada definida por el cliente de manera de indicar cuales son las más importantes para el resultado final, serán divididas en tareas y su tiempo será estimado por los desarrolladores. Generalmente se espera que la estimación de tiempo de cada historia de usuario se sitúe entre unas 10 horas y un par de semanas.

Estructura:

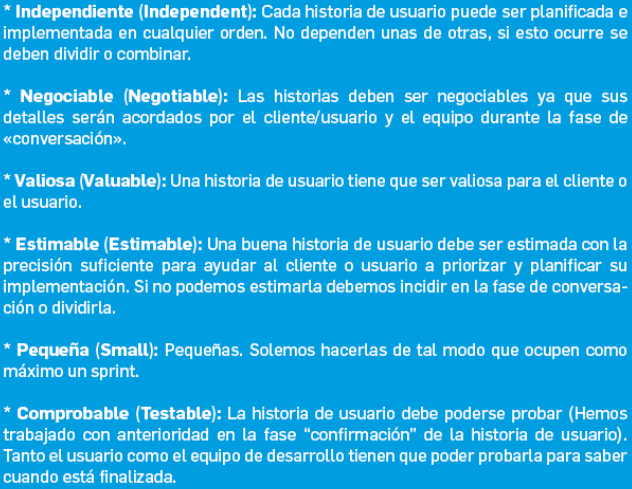


Tema, épicas e historias:

Durante el proceso de análisis de requerimientos se suele agrupar a las historias de usuario en épicas y a su vez, estas ultimas relacionadas a un tema particular. Las historias épicas son ingresadas a la pila de producto.

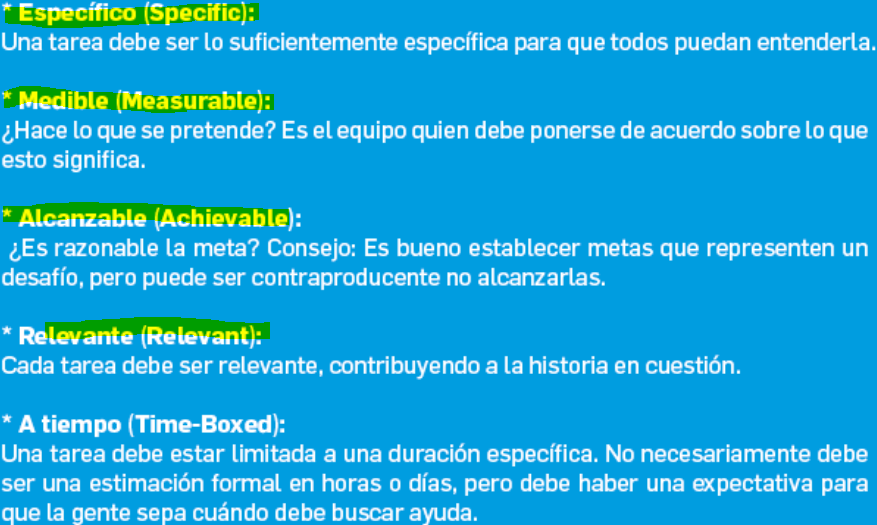
Método Invest:

Inventado por Bill Wake, permite asegurar la calidad de la escritura de las historias de usuario.

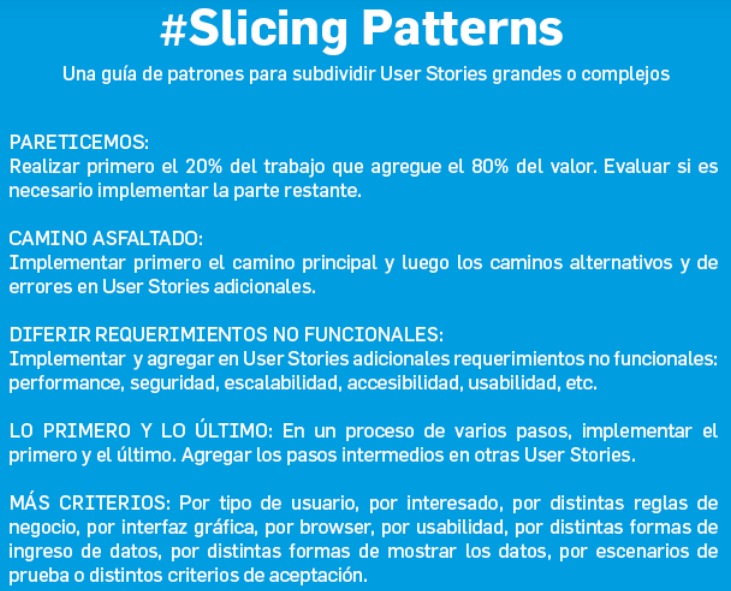


Método Smart:

Una vez compuestas las historias de usuario, mediante el método Smart se podrá evaluar si están definidas correctamente.



Subdivisión de historias de usuario:



“3 C”:

Card: Donde se escribe la historia de usuario de manera clara y con la menor ambigüedad posible.

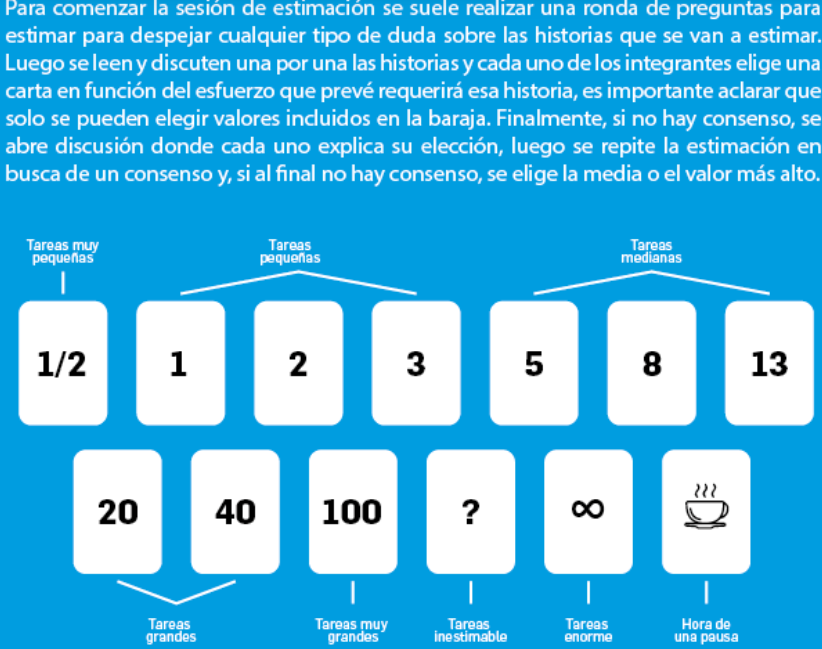
Conversation: Es importante validar y discutir con el cliente y el equipo de desarrollo, esto por lo general se realiza en la reunión de desarrollo.

Confirmation: Es la confirmación de que todos los miembros comprendieron los requisitos solicitados.

Puntos de historia:

Solo es valiosa para el grupo de scrum, y les permitirá determinar que tarea son valiosas para el sprint. Vistas por el product owner, mediante los puntos de historia podrá determinar las debilidades del equipo, reflejadas en las tareas que fueron más difíciles en completarse.

Planning Póker:



DESAFIO: <https://www.youtube.com/watch?v=ySJMHCI-8Uo>

<https://www.youtube.com/watch?v=ky6wFiF5vMk>

**GITHUB**

Sistema de control de versiones:

Estos sistemas son herramientas que permiten realizar un seguimiento de los cambios y también proteger el código de los posibles errores. Además, estos sistemas son de gran importancia a la hora de trabajar en conjunto a un equipo.

1. Gracias al historial de cambios se puede saber el autor, la fecha y notas escritas sobre los cambios realizados. También permite volver a versiones anteriores para ayudar a analizar causas raíces de errores y es crucial cuando hay que solucionar problemas de versiones anteriores.
2. Creación y fusión de ramas. Al tener varios integrantes del equipo trabajando al mismo tiempo, cada uno en una tarea diferente, pueden beneficiarse de tener flujos de trabajo independientes.
3. Trazabilidad de los cambios que se realizan en el software. Poder conectar el sistema de control con un software de gestión de proyectos y seguimiento de errores, ayuda con el análisis de la causa raíz de los problemas y con la recopilación de información.

Git:

Git es un proyecto de código abierto maduro y con un activo mantenimiento desarrollado originalmente por [Linus Torvalds](https://www.ecured.cu/Linus_Torvalds). Este sistema presenta una arquitectura distribuida, es decir que, cada desarrollador posee una copia del trabajo en un repositorio local donde puede albergar el historial completo de todos los cambios y, mediante comandos determinados, realiza sincronizaciones al repositorio remoto.

Sus principales características son el **rendimiento, la seguridad y la flexibilidad**.

Sectores fundamentales en Git:

**Directorio de trabajo (Working Area):** Es una copia de una versión del proyecto, archivos sacados de la base de datos comprimidos y colocados en el disco para ser utilizados o modificados.

**Area de preparación (Staging Area):** Es un archivo dentro del directorio de git que contiene información acerca de lo que va ir en la próxima confirmación.

**Directorio de Git (Local repository):** Es el lugar donde se almacenan los metadatos y la base de datos de objetos del proyecto. Es lo que se copia cuando se clona un repositorio desde otra fuente.

**Repositorio remoto (Remote repository):** Es el repositorio que se encuentra en un servidor remoto y donde los integrantes del equipo sincronizan sus trabajos.

El objetivo es que los archivos que ya sabemos que están funcionando correctamente queden en el Repositorio Local y en caso que se quiera compartir el código debemos enviarlo al Repositorio remoto.

**Comandos básicos:**

● **git init** es el comando para inicializar un directorio como repositorio Git, se ejecuta dentro del directorio del proyecto, y como resultado crea un subdirectorio **.git** que contiene todos los archivos para poder realizar el seguimiento de los cambios, etiquetas, etc.

● **git add <archivo>** luego de la creación, modificación o eliminación de un archivo, los cambios quedan únicamente en el área de trabajo, por lo tanto, es necesario pasarlos al área de preparación mediante el uso del comando **git add**, para que sea incluido dentro de la siguiente Confirmación (*cometer*).

●**git status** es un comando que permite conocer en qué estado se encuentran los archivos

●**git commit**, con este comando se confirman todos los cambios registrados en el área de preparación, o lo que es lo mismo, se pasan los cambios al repositorio local.

●**git push** es el comando que se utiliza para enviar todas las confirmaciones registradas en el repositorio local a un repositorio remoto.

●**git pull** funciona al inverso de **git push**, trayendo todos los cambios al repositorio local, pero también dejándolos disponibles directamente para su modificación o revisión en el área de trabajo. Es importante mencionar que se utiliza cuando ya se tiene un repositorio local vinculado a uno remoto, al igual que con el comando **git push**.

●**git clone**, en el caso de necesitar "bajar" un repositorio remoto de algún proyecto ya existente se puede ejecutar este comando. Genera un directorio (con el nombre del repositorio o uno especificado explícitamente) que contiene todo lo propio al proyecto, además del subdirectorio **.git** necesario para poder gestionar los cambios y todo lo pertinente al repositorio Git.

**GITHUB:**

GitHub es una plataforma de colaboración formal e informal de desarrollo de software (conocida también como plataforma de social coding). en esta se pueden publicar repositorios remotos que funcionan bajo el sistema de control de versiones Git. La plataforma configura los proyectos nuevos como de código abierto, por lo que cualquier persona puede verlos, pero esto es configurable.