Programación Web y Conceptos básicos relacionados a Internet

Introducción

Internet se podría definir como una red de redes de ordenadores cuya finalidad es permitir el intercambio libre de información entre todos sus usuarios. Pero sería un error considerar Internet únicamente como una red de computadoras. Podemos considerar las computadoras simplemente como el medio que transporta la información. En este caso Internet sería una gran fuente de información práctica y divertida.
Con Internet podemos enviar mensajes (chatear), programas ejecutables, ficheros de texto, consultar, pedir libros, hacer compras, etc.
Internet se basa "básicamente" en cuatro servicios:

- el servicio de correo electrónico (e-mail) que transmite y recibe mensajes: nos podemos poner en contacto con cualquier otro usuario mediante el intercambio de mensajes.
- servicio de noticias (news): te suscribes a un grupo de noticias y recibis información sobre ese tema.
- acceso remoto (telnet): podes conectarte como terminal y establecer una sesión de trabajo en cualquier ordenador (ordenador remoto) de la red si dispones de los permisos de acceso necesarios para acceder a él.
- transferencia de ficheros (ftp , File Transfer Protocol o Protocolo de Transferencia de Archivos) que permite transferir archivos de una computadora a otra.

Servidor

Es el ordenador encargado de proporcionar al navegador del cliente los documentos y medios que éste solicita.

Servidor Web o Servidor HTTP

Es un programa informático que procesa una aplicación del lado del servidor, realizando conexiones bidireccionales o unidireccionales y síncronas o asíncronas con el cliente y generando o cediendo una respuesta en cualquier lenguaje o aplicación del lado del cliente. El código recibido por el cliente es renderizado por un navegador web. Para la transmisión de todos estos datos suele utilizarse algún protocolo. Generalmente se usa el protocolo HTTP para estas comunicaciones, perteneciente a la capa de aplicación del modelo OSI. El término también se emplea para referirse al ordenador donde se ejecutan esas tareas.

Dominios y URL

Los servidores se identifican con un valor llamado IP (Internet Protocol). Esta IP es única para cada ordenador y, por lo tanto, trabaja como una dirección que permite ubicar a un ordenador dentro de una red. Cuando el navegador tiene que acceder a un servidor para descargar el documento solicitado por el usuario, primero busca el servidor a través de esta dirección IP y luego le pide que

le envíe el documento. Las direcciones IP están compuestas por números enteros entre 0 y 255 separados por un punto, o números y letras separadas por dos puntos, dependiendo de la versión (IPv4 o IPv6). Por ejemplo, la dirección 216.58.198.100 corresponde al servidor donde se encuentra alojado el sitio web de Google. Si escribimos esta dirección IP en la barra de navegación de nuestro navegador, la página inicial de Google se descarga y muestra en pantalla.
Los sitios web están compuestos por múltiples archivos, por lo que debemos agregar el nombre del archivo al dominio para indicar cuál queremos descargar. Esta construcción se llama URL e incluye tres partes:

Navegador Es el programa que se utiliza para acceder a los contenidos de Internet. Debe ser capaz de comunicarse con un servidor y comprender el lenguaje de todas las herramientas que manejan la información de Web. Los navegadores más populares son Internet Explorer, Mozilla Firefox, NetScape, Opera, Safari, etc.

HTML (HyperText Markup Language)

Es un lenguaje compuesto por un grupo de etiquetas definidas con un nombre rodeado de paréntesis angulares. Los paréntesis angulares delimitan la etiqueta y el nombre define el tipo de contenido que representa. Por ejemplo, la etiqueta indica que el contenido es código HTML. Algunas de estas etiquetas son declaradas individualmente y otras son declaradas en pares, que incluyen una de apertura y otra de cierre, como (en la etiqueta de cierre el nombre va precedido por una barra invertida). Las etiquetas individuales y las de apertura pueden incluir atributos para ofrecer información adicional acerca de sus contenidos. Las etiquetas individuales y la combinación de etiquetas de apertura y cierre se llaman elementos. Los elementos compuestos por una sola etiqueta se usan para modificar el contenido que los rodea o incluir recursos externos, mientras que los elementos que incluyen etiquetas de apertura y cierre se utilizan para delimitar el contenido del documento.

```
<!DOCTYPE html>
<html lang="es">
        <head>
                <title>Mi primer documento HTML</title>
        </head>
        <body>
 <p>HOLA MUNDO!</p>
        </body>
</html>
```

Breve historia de HTML

La historia completa de HTML es algo larga, por lo que se muestra resumida a partir de la información que se puede encontrar en Wikipedia.

El origen de HTML se remonta a 1980, cuando el físico Tim Berners-Lee, investigador del CERN (Organización Europea para la Investigación Nuclear) propuso un nuevo sistema de "hipertexto" para compartir documentos.

Los sistemas de hipertexto habían sido desarrollados años antes. En el ámbito de la informática, el hipertexto permite que los usuarios accedan a la información relacionada con los documentos electrónicos que visualizan. En cierta manera, los primitivos sistemas de hipertexto podrían asimilarse a los enlaces de las páginas web actuales.

Tras finalizar el desarrollo de su sistema, Tim Berners-Lee lo presentó a una convocatoria organizada para desarrollar un sistema de hipertexto para internet. Después de unir sus fuerzas con el ingeniero de sistemas Robert Cailliau, presentaron la propuesta ganadora llamada WorldWideWeb (W3). El primer documento formal con la descripción de HTML se publicó en 1991 bajo el nombre "HTML Tags" (Etiquetas HTML) y todavía hoy puede ser consultado en línea a modo de reliquia informática.

La primera propuesta oficial para convertir HTML en un estándar se realizó en 1993 por parte del organismo IETF (Internet Engineering Task Force). Aunque se consiguieron avances significativos (en esta época se definieron las etiquetas para imágenes, tablas y formularios) ninguna de las dos propuestas de estándar, llamadas HTML y HTML+ consiguieron convertirse en estándar oficial.

En 1995, el organismo IETF organizó un grupo de trabajo de HTML y el 22 de septiembre publicaron el estándar HTML 2.0. A pesar de su nombre, HTML 2.0 es el primer estándar oficial de HTML.

A partir de 1996, los estándares de HTML los publica otro organismo de estandarización, el W3C. La versión HTML 3.2 se publicó el 14 de Enero de 1997 y es la primera recomendación de HTML publicada por el W3C. Esta revisión incorpora los últimos avances de las páginas web desarrolladas hasta 1996, como applets de Java y texto que fluye alrededor de las imágenes.

HTML 4.0 se publicó el 24 de Abril de 1998 (versión corregida de la publicación del 18 de Diciembre de 1997) y supuso un gran salto desde las versiones anteriores. Entre sus novedades más destacadas se encuentran las hojas de estilos CSS, la posibilidad de incluir pequeños programas o scripts en las páginas web, mejora de la accesibilidad de las páginas diseñadas, tablas complejas y mejoras en los formularios.

La última especificación oficial de HTML se publicó el 24 de diciembre de 1999 y se denomina HTML 4.01. Se trata de una revisión y actualización de la versión HTML 4.0, por lo que no incluye novedades significativas.

Desde la publicación de HTML 4.01, la actividad de estandarización de HTML se detuvo y el W3C se centró en el desarrollo del estándar XHTML. Por este motivo, en el año 2004, las empresas Apple, Mozilla y Opera mostraron su preocupación por la falta de interés del W3C en HTML y decidieron organizarse en una nueva asociación llamada WHATWG (Web Hypertext Application

Technology Working Group). La actividad actual del WHATWG se centra en el estándar HTML5, cuyo primer borrador oficial se publicó el 22 de enero de 2008. Debido a la fuerza de las empresas que forman el grupo WHATWG y a la publicación de los borradores de HTML 5.0, en marzo de 2007 el W3C decidió retomar la actividad estandarizadora de HTML.

HTML5 ya es un estándar (recomendado), el consorcio W3C, con el inventor de la Web Sir Tim Berners-Lee, presentó el 27 de octubre de 2014 la versión final. Su intención es que se construya lo que se ha llamado Plataforma Web abierta, donde el HTML5, junto a Javascript y CSS3, se podrá utilizar para el desarrollo de aplicaciones multiplataforma (Linux, Windows, Android, iOS..).

Como parte de la estrategia para vincular a más gente en el proceso actual, el director ejecutivo del Consorcio, el Dr. Jeff Jaffe (MIT) ha publicado un texto acerca de las prioridades del mismo bajo el título de 'Fundamentos de Aplicación' que sostienen todo el trabajo orientado hacia la plataforma actual y de "nueva generación":
Seguridad y privacidad, todo lo relacionado con autenticaciones, encriptación, protección de identidad y actividad en línea.
Diseño y desarrollo de la web, en cuanto a estilo, formato, gráficos, animación y tipografía.
Interacción con distintos equipos como sistemas de sensores y Bluetooth.
Ciclo de uso de aplicación para administración de tareas fuera de conexión y sincronización.
Medios y comunicaciones en tiempo real, para efectos, por ejemplo, de transmisiones en vivo (streaming).
Desempeño y afinación de la capacidad y precisión en la respuesta y descarga de sitios web con sus funciones.
Usabilidad y accesibilidad, para un web internacional, multilingüe y de acceso para personas con distintas discapacidades.
Servicios como pagos y web social.

De forma paralela a su actividad con HTML, W3C ha continuado con la estandarización de XHTML, una versión avanzada de HTML y basada en XML. La primera versión de XHTML se denomina XHTML 1.0 y se publicó el 26 de Enero de 2000 (y posteriormente se revisó el uno de Agosto de 2002).

XHTML 1.0 es una adaptación de HTML 4.01 al lenguaje XML por lo que mantiene casi todas sus etiquetas y características, pero añade algunas restricciones y elementos propios de XML. La versión XHTML 1.1 ya ha sido publicada en forma de borrador y pretende modularizar.

HTML5

Incorpora tres características (estructura, estilo, y funcionalidad), con lo que integra tres lenguajes de programación independientes: HTML, CSS, y JavaScript. Estos lenguajes están compuestos por grupos de instrucciones que los navegadores pueden interpretar para procesar y mostrar los documentos al usuario. Para crear nuestros documentos, tenemos que aprender todas las instrucciones incluidas en estos lenguajes y saber cómo organizarlas.

CSS (Cascading Style Sheets)

Es el lenguaje que se utiliza para definir los estilos de los elementos HTML, como el tamaño, el color, el fondo, el borde, etc.
color: #FF0000;
Las propiedades CSS se pueden agrupar usando llaves. Un grupo de una o más propiedades se llama regla y se identifica por un nombre llamado selector.

```
body {
        width: 100%;
 margin: 0px;
        background-color: #FF0000;
}
```

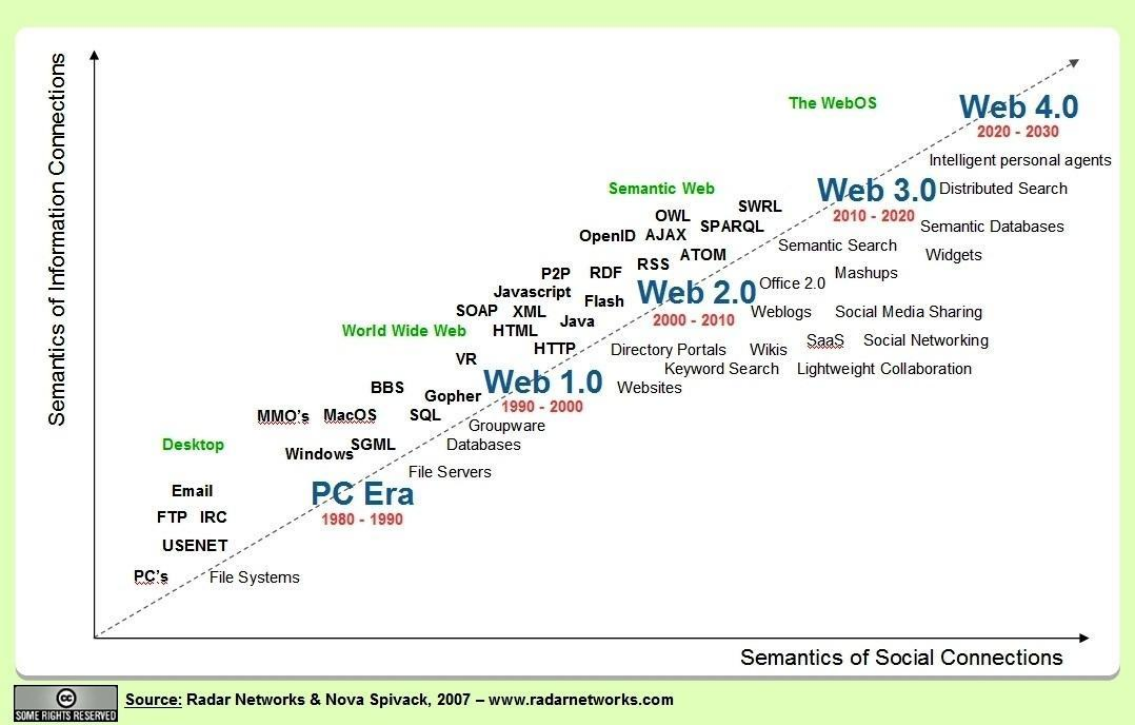
JavaScript

A diferencia de HTML y CSS, JavaScript es un lenguaje de programación. Para ser justos, todos estos lenguajes pueden ser considerados lenguajes de programación, pero en la práctica existen algunas diferencias en la forma en la que suministran las instrucciones al navegador.

```
<script>
        function cambiarColor() {
                document.body.style.backgroundColor = "#0000FF";
        }
        document.addEventListener("click", cambiarColor);
</script>
```

Los códigos desarrollados en HTML, CSS, y JavaScript son ejecutados por el navegador en el ordenador del usuario (el cliente). Esto significa que, después de que los archivos del sitio web se suben al servidor, permanecen inalterables hasta que se descargan en un ordenador personal y sus códigos son ejecutados por el navegador. Aunque esto permite la creación de sitios web útiles e interactivos, hay momentos en los cuales necesitamos procesar la información en el servidor antes de enviarla al usuario. El contenido producido por esta información se denomina contenido dinámico, y es generado por códigos ejecutados en el servidor y programados en lenguajes que fueron especialmente diseñados con este propósito (lenguajes de servidor). Cuando el navegador solicita un archivo que contiene este tipo de código, el servidor lo ejecuta y luego envía el resultado como respuesta al usuario. Estos códigos no solo se utilizan para generar contenido y documentos en tiempo real, sino también para procesar la información enviada por el navegador, almacenar datos del usuario en el servidor, controlar cuentas, etc. Existen varios lenguajes disponibles para crear código ejecutable en los servidores. Los más populares son PHP, Ruby, y Python.

WEB



La web 1.0

La Web 1.0 (1991-2003) es la forma más básica que existe, con navegadores de sólo texto bastante rápidos ya que es de sólo lectura. El usuario no puede interactuar con el contenido de la página (nada de comentarios, respuestas, citas, etc), estando totalmente limitado a lo que el Webmaster sube a ésta.

Algunos elementos de diseño típicos de un sitio Web 1.0 incluyen:
Páginas estáticas en vez de dinámicas por el usuario que la visita.
El uso de framesets o Marcos.
Extensiones propias del HTML como <bilnk> y <marquee>, etiquetas introducidas durante la guerra de navegadores web.
Libros de visitas online o guestbooks
botones GIF, casi siempre a una resolución típica de 88x31 pixels en tamaño promocionando navegadores web u otros productos.
formularios HTML enviados vía email. Un usuario llenaba un formulario y después de hacer clic se enviaba a través de un cliente de correo electrónico, con el problema que en el código se podía observar los detalles del envío del correo electrónico.
No se podian adherir comentarios ni nada parecido
Todas sus páginas se creaban de forma fija y muy pocas veces se actualizaban.
No se trata de una nueva versión, sino de una nueva forma de ver las cosas.

La web 2.0

El término Web 2.0 comprende aquellos sitios web que facilitan el compartir información, la interoperabilidad, el diseño centrado en el usuario y la colaboración en la World Wide Web. Un sitio Web 2.0 permite a los usuarios interactuar y colaborar entre sí como creadores de contenido generado por usuarios en una comunidad virtual, a diferencia de sitios web estáticos donde los usuarios se limitan a la observación pasiva de los contenidos que se han creado para ellos. Ejemplos de la Web 2.0 son las comunidades web, los servicios web, las aplicaciones Web, los servicios de red social, los servicios de alojamiento de videos, las wikis, blogs, mashups y folcsonomías.

Servicios asociados:
Blogs: Un blog es un espacio web personal en el que su autor puede escribir cronológicamente artículos, noticias...(con imágenes y enlaces).
Wikis: Una wiki es un espacio web corporativo, organizado mediante una estructura hipertextual de páginas donde varias personas elaboran contenidos de manera asíncrona.
Redes sociales: Sitios web donde cada usuario tiene una página donde publica contenidos y se comunica con otros usuarios. Ejemplos: Facebook, Twitter, Tuenti, Hi5, Myspace, etc.
Entornos para compartir recursos: Entornos que nos permiten almacenar recursos o contenidos en Internet, compartirlos y visualizarlos cuando nos convenga. Existen de diversos tipos, según el contenido que albergan o el uso que se les da:
Documentos: Google Drive y Office Web Apps (SkyDrive), en los cuales podemos subir nuestros documentos, compartirlos y modificarlos.
Videos: Youtube, Vimeo, Dailymotion, Dalealplay... Contienen miles de vídeos subidos y compartidos por los usuarios.
Fotos: Picassa, Flickr... Permiten disfrutar y compartir las fotos también tenemos la oportunidad de organizar las fotos con etiquetas, separándolas por grupos como si fueran álbumes, podemos seleccionar y guardar aparte las fotos que no queremos publicar.
Agregadores de noticias: Digg, Meneame... Noticias de cualquier medio son agregadas y votadas por los usuarios.
Almacenamiento online: Dropbox, Google Drive, SkyDrive
Presentaciones: Prezzi, Slideshare.
Plataformas educativas
Aulas virtuales (síncronas)
Encuestas en línea

La web 3.0

Web 3.0 es una expresión que se utiliza para describir la evolución del uso y la interacción de las personas en internet a través de diferentes formas entre los que se incluyen la transformación de la red en una base de datos, un movimiento social hacia crear contenidos accesibles por múltiples aplicaciones non-browser, el empuje de las tecnologías de inteligencia artificial, la web semántica, la Web Geoespacial o la Web 3D.

Se basa en la idea de añadir metadatos semánticos y ontológicos a la World Wide Web. Esas informaciones adicionales —que describen el contenido, el significado y la relación de los datos— se deben proporcionar de manera formal, para que así sea posible evaluarlas automáticamente por

máquinas de procesamiento. El objetivo es mejorar Internet ampliando la interoperabilidad entre los sistemas informáticos usando "agentes inteligentes". Agentes inteligentes son programas en las computadoras que buscan información sin operadores humanos. Con la web 3.0 se busca que los usuarios puedan conectarse desde cualquier lugar, cualquier dispositivo y a cualquier momento.

Entre sus innovaciones destacan:

Bases de datos
Inteligencia artificial
Web semántica y SOA
Evolución al 3D

La web 4.0

las aplicaciones ya no estarán en nuestras PC's, estarán en la Internet y por ende en todos lados. Pasaremos de una red "tonta" a una red "inteligente" donde el objetivo primordial será el de unir las inteligencias donde tanto las personas como las cosas se comuniquen entre sí para generar la toma de decisiones. Para el 2020 o quizás antes se espera que haya "agentes" en la Web que conozcan, aprendan y razonen como lo hacemos las personas.
La Web Ubicua es un concepto que está aún en desarrollo, pero me llama curiosamente la atención como se van complementando algunas tecnologías que nos permiten imaginar o soñar lo que podemos esperar en un futuro no muy lejano. Imagínese recibir información en su celular en la calle por la que camina y que su propio equipo le haga una reservación en el restaurante de la esquina con solo saber sus gustos.

Estándares

World Wide Web Consortium (WWW), en inglés: World Wide Web Consortium (W3C), es un consorcio internacional que propone recomendaciones y estándares que aseguran el crecimiento de la World Wide Web a largo plazo. Fue creado en octubre de 1994 y está dirigido por Tim Berners-Lee, el creador original del URL (Uniform Resource Locator, Localizador Uniforme de Recursos), del HTTP (HyperText Transfer Protocol, Protocolo de Transferencia de HiperTexto) y del HTML (Hyper Text Markup Language, Lenguaje de Marcado de HiperTexto), que son las principales tecnologías sobre las que actualmente se basa la Web.

El W3C está integrado por:

- Miembros: a 11 de febrero de 2019 contaba con 451 miembros.
- Equipo (W3C Team): 65 investigadores y expertos de todo el mundo.
- Oficinas (W3C Offices): centros regionales establecidos en Alemania y Austria (oficina conjunta), Australia, Benelux (oficina conjunta), China, Corea del Sur, España, Finlandia, Grecia, Hong Kong, Hungría, India, Israel, Italia, Marruecos, Suecia y Reino Unido e Irlanda (oficina conjunta).

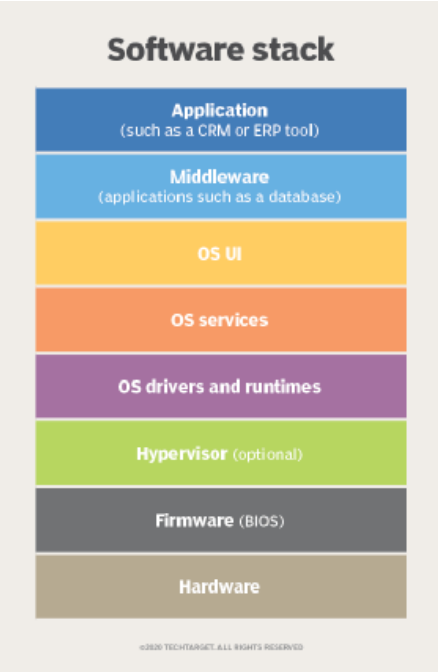
Procesos de desarrollo de software

Introducción

Antes de adentrarnos en los procesos y protocolos para desarrollar software, debemos pensar las características de la tecnología web en la actualidad, ya que a partir de esto es qué se definen muchos de estos procesos.

Software en capas.

Una de las principales características de los sistemas modernos, es qué se encuentran desarrollados a partir de capas, cada una con prestaciones y objetivos específicos. Para visualizar esto de mejor manera, podemos pensar en la integridad de un sistema, como a una pila (stack) de componentes, donde la información va subiendo y bajando de un nivel al otro, de acuerdo con lo qué esté sucediendo.



Microservicios

Otra característica remarcable de los sistemas actuales es qué los mismos se encuentran funcionales a través de la estructura de microservicios. Este concepto, será material de estudio en el futuro. Pero, por el momento, debemos comprender qué es una forma de implementar software a partir de pequeños módulos, relacionados entre sí.

Conclusión

Tanto la lógica de capas, como la arquitectura de microservicios, van en pos de desarrollos más modulares, con mayor capacidad de corrección y reutilización, haciendo qué el proceso de fabricación software implique una ingeniería menos “pesada” y de constante evolución.

Estructura básica de un proyecto Web

Dentro de las capas previamente mencionadas, podemos identificar tres etapas muy marcadas, más allá de qué estás después puedan estar divididas en más subcapas.

Front-end (Capa de presentación): Es la que ve el usuario (también se la denomina «capa de usuario»), presenta el sistema, comunica la información y captura la información del usuario en un mínimo de procesos (realiza un filtrado previo para comprobar que no hay errores de formato). También es conocida como interfaz gráfica y debe tener la característica de ser «amigable» (entendible y fácil de usar) Esta capa se comunica únicamente con la capa de negocio.

Back-end (Capa de negocio): es donde residen los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos almacenar o recuperar datos de él. También se consideran aquí los programas de aplicación.

Capa de datos: es donde residen los datos y es la encargada de acceder a los mismos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

¿Cómo se compone un equipo de desarrollo?

Por supuesto, qué la respuesta a esta pregunta no es algo estanco, ni del todo estandarizado, ya qué depende muchísimo de la compañía, el proyecto en cuestión y el estadio del mismo, entre muchos otros factores. Pero a partir de todo lo enunciado hasta acá, podemos establecer ciertos patrones en común, encontrando las siguientes posiciones:

Desarrolladores: Se encargan de desarrollar a nivel código e infraestructura, las distintas etapas involucradas en el proyecto. Partiendo de lo mencionado párrafos atrás, es común encontrar desarrolladores focalizados al back-end o al front-end. Se denomina full-stack a quienes se dedican a proyectos qué involucren ambas etapas en su conjunto.

Por otro lado, con los avances de los sistemas de software es común encontrar, en un mismo proyecto, muchos desarrolladores de front-end, pero dedicados a plataformas específicas, web, android, Roku, etc. También aparece el grupo de back-end, donde no es tan habitual encontrar la multiplicidad de tecnologías.

Analistas de calidad: En conjunto con los desarrolladores, trabajan los analistas de calidad denominados QA (quality analist) o QC (quality control) ellos se encargan de evaluar exhaustivamente los desarrollos logrados por el conjunto previamente mencionado. Con el fin de prevenir cualquier posible error en instancias productivas.

Gestiones Técnicas: De las siguientes posiciones, suele haber un único miembro por equipo, de hecho es común, qué una de estas posiciones, esté dedicada a más de un proyecto por persona. Aquí aparecen:

TL (Team Leader), mayormente enfocado a liderar el grupo a nivel técnico. Debe tener pleno conocimiento en todo lo que se está desarrollando, sumado a un contacto con el cliente o entidad reguladora del desarrollo en cuestión, de modo qué los objetivos sean técnicamente alcanzables en el plazo estipulado.

Arquitecto: Mas qué nada en proyectos de back-end, se posee un arquitecto qué diseña la infraestructura de lo qué se va a desarrollar, en gran medida, esto implica gestionar como, lo desarrollado será puesto en producción.

DevLead: Así como hay un líder de equipo, es común encontrar un líder exclusivo de los desarrolladores, suele ser quien más conocimiento y/o experiencia posee en la tecnología o en el desarrollo en cuestión.

DevOps: (development operations, operaciones desarrollo): Su función es la de gestionar a nivel plataforma lo establecido por el arquitecto y desarrollado por los devs. Suele ser la persona qué maneja plataformas tales como Azure, AWS, entre otras.

Gestiones de negocio: En esta etapa se involucran cuestiones qué tienen qué ver con la organización del proyecto a nivel negocio, tanto por el lado de los recursos qué se disponen para el mismo, como de los tiempos y formas de entrega. Ponderadamente, en esta sección aparecen:

- **Business Analytics (BA):** Este se encarga de gestionar información del y para el proyecto, tal como documentación, herramientas de seguimiento etc. Debe tener muy buen conocimiento de lo qué se desarrolla, a nivel objetivo, no técnico.

- **Project Manager (PM):** Se debe encargar de qué las condiciones para qué el objetivo se cumpla, estén dadas, por lado de los recursos, como de la motivación y dedicación del equipo. Parte de esto último implica gestionar reuniones apropiadas y eventos especiales para constituir un ciclo de trabajo apropiado.

Metodologías Ágiles

Habiendo definido las etapas y características de un proceso de desarrollo de software, procedemos a enunciar como todo esto se conjuga y organiza dentro de lo qué son las metodologías ágiles. Brevemente podemos establecerlas como a una serie de protocolos y ceremonias qué se llevan adelante dentro de un marco de trabajo basado en iteraciones.

Primero qué nada, separemos este último detalle. ¿Qué es un desarrollo basado en iteraciones?

Desarrollo en cascada

Para ser más claros en esto, pensemos lo contrario. Desarrollo convencional, mayormente implementado en las ingenierías duras, como ser edilicia, electrónica, mecánica, entre otras. Este tipo de desarrollo, antagónico al desarrollo iterado, se conoce como desarrollo en cascada.

Conectar dos dispositivos en cascada (o en serie), implica qué la salida de un elemento se conecta a la entrada del otro. Llevado a procesos, implica poder comenzar con un proceso, una vez finalizados los procesos previos necesarios. Esto se puede ver en el diagrama de Gantt, herramienta empleada para dar seguimiento a este tipo de desarrollos.

Desarrollo iterado

En oposición a esto, tenemos el desarrollo iterado. Bajo este criterio siempre estaremos volviendo sobre un mismo componente, solo qué vamos aumentando su complejidad y prestaciones. Esto permite revisiones constantes, mayor flexibilidad y una mejor capacidad de adaptación a distintos cambios qué puedan ocurrir.

Como es de esperar, existen múltiples metodologías que califican dentro del paradigma, a continuación iremos enunciando detalles que son comunes a la mayoría de los casos.

Cuestiones de tiempo

Sprint: En inglés podría traducirse como “pique” o “corrida”, de esta forma se segmentan los periodos de tiempo de desarrollo, suelen ser de 1 o 2 semanas, dependiendo el proyecto y/o compañía en cuestión. La idea es qué sprint a sprint, se vaya revisando y probando los distintos desarrollos a medida que el proyecto crece.

Estimaciones y Fibonacci: La secuencia de Fibonacci se establece a partir de números qué se obtienen de sumar los dos números previos, 0, 1, 1, 2, 3, 5, 8, 13… Se usan estos valores para asignar complejidad estimada a una tarea. También es a criterio del proyecto y/o empresa, establecer qué complejidad conlleva cada valor de Fibonacci. Se supone qué a mayor experiencia del desarrollador, mayor cantidad de puntos puede abarcar en un sprint.

Protocolos y ceremonias

Estas etapas están muy asociadas a la metodología agile en cuestión. Probablemente SCRUM sea la más usada y difundida. Dentro de esta forma de trabajar, se encuentran las siguientes ceremonias a realizarse una vez en cada Sprint, a excepción de las daily.

Daily (diaria): Como su nombre lo indica, se lleva a cabo diariamente. En esta reunión se pone al tanto del estatus de cada miembro del equipo. Se debe hablar menos de un minuto por persona, debiendo focalizar en tareas realizadas, a realizar y si se tiene algún bloqueo.

Demo (demostración): Consiste en presentar lo desarrollado durante el sprint hacia el cliente o quien está interesado en qué el proyecto se lleve adelante, además de una demostración es un

espacio para recibir devoluciones y probar de manera tangible el funcionamiento de la etapa en cuestión.

Planning (planeamiento): Se lleva adelante con el grupo de desarrollo, en esta etapa se planea cómo abordar el Sprint en cuestión, desde cómo abordar las tareas, repartirlas dentro del grupo, asignarles complejidad, etc.

Retro (Retrospectiva): Es una reunión con el objetivo de qué el equipo mejore, valorando lo qué resultó bien, notando lo qué fue mal, y exponiendo alternativas acerca de cómo mejorar lo malo y potenciar lo bueno

A continuación, se establece un resumen acerca de las metodologías Agile

¿Qué son las metodologías ágiles o Agile?

Las metodologías ágiles no son apenas herramientas, sino estrategias integrales - estrategias de marketing, gestión de servicios y más - que impulsan a las organizaciones a gestionar sus proyectos con rapidez y flexibilidad. Estas metodologías ayudan en el desarrollo de proyectos que necesitan mayor enfoque para adecuarse a las necesidades del cliente.

Es decir, una metodología ágil es una innovadora forma de trabajar y organizar flujos, que divide los proyectos en partes, permite adaptarse sobre la marcha, complementa y resuelve etapas en poco tiempo.

Con las metodologías ágiles, no se planifica ni se diseña el proyecto por adelantado, es decir, a medida que ellas se van desarrollando se va definiendo el proyecto. Siendo así, los involucrados trabajan por períodos específicos, mientras que cada miembro del equipo debe ejecutar una serie de tareas. Al final de la ejecución de las tareas, cada miembro o equipo entrega los avances, recibe devoluciones y comienza otra vez el proceso, lo que permite que los cambios necesarios sean implementados.

Los 12 principios de la metodología Agile

El concepto de metodología ágil nació en 2001 y fue pensado por un grupo de 17 líderes de la informática, desarrolladores de software y administradores, que se juntaron para discutir los métodos de desarrollo de software ligero.

De ese encuentro, surgieron una serie de principios que conforman el manifiesto Agile, que determina las bases primordiales que debe cumplir cualquier método Agile. En un primer momento, los principios fueron pensados para los programadores, pero se adaptaron y hoy son valiosos para los equipos de distintas áreas.

A continuación están los principios adaptados:

- El cliente en primer lugar a través de entregas tempranas y continuas: la entrega temprana y continua aumenta la probabilidad de satisfacer las demandas de los clientes y contribuye a la generación de un retorno de la inversión más rápido.

- Estar siempre abierto a cambios: las solicitudes de cambio deben ser siempre bienvenidas, incluso en las últimas etapas de la ejecución del proyecto. A diferencia de la gestión de proyectos tradicional, en las metodologías ágiles, los equipos tienen como objetivo aceptar la incertidumbre y reconocer que incluso un cambio tardío puede tener mucho valor para el cliente final.

- Entregas con valor: originalmente fue establecido para las metodologías ágiles "entregar software que funcione con frecuencia, desde un par de semanas hasta un par de meses, con una preferencia hacia la escala de tiempo más corta". Aquí el principal objetivo es reducir el tamaño de los lotes que utilizas para procesar el trabajo.

- Eliminar silos organizacionales entre los equipos: en este punto el objetivo es crear una sincronización entre las personas que crean valor y las que lo planifican o venden. Sólo así lograrás una colaboración interna fluida, lo que mejorará el rendimiento de tu proceso.

- Motivación a los involucrados: al reducir la microgestión y potenciar a los miembros del equipo motivados, los proyectos se completan más rápido y con mejor calidad.

- La comunicación, de preferencia, debe ocurrir cara a cara: la comunicación en persona reduce el tiempo de respuesta. Pero, con el desarrollo de la tecnología y equipos trabajando remotamente, podemos adaptar este principio ágil para: comunicación directa. Es decir, cuando tengas una manera de llegar rápidamente a tu equipo para discutir asuntos de trabajo, hazlo.

- Un software de trabajo es la principal medida de progreso: no importa cuántas horas de trabajo hayas invertido en tu proyecto: si el resultado no es lo que tu cliente espera que sea, tienes problemas.

- Mantén un ritmo de trabajo sostenible: al utilizar una metodología ágil, uno de los objetivos es evitar sobrecargas y optimizar la forma de trabajo de tus equipos para que puedas realizar entregas con frecuencia y responder a los cambios.

- La excelencia continua mejora la agilidad: al mantener la excelencia operativa, tus equipos tienen menos problemas para reaccionar a los cambios y mantienen la agilidad.

- La simpleza es importante: siempre que puedas hacer algo de manera sencilla, hazlo, al final ¿por qué perder tiempo complicándote?. Ten esto en consideración al implementar una metodología ágil y evita hacer algo por el simple hecho de hacerlo.

- Equipos más libres generan más valor: equipos motivados generan el máximo valor para el cliente. Fíjate: Si tienes que conducir hacia adelante a tus equipos, tal vez aún no es el mejor momento para implementar una metodología ágil.

- Reflexiona sobre tu forma de trabajar para aumentar la productividad: tú y tus equipos deben reflexionar sobre cómo ser más eficaces, discutan lo que salió mal y hagan los ajustes necesarios para volver a empezar. Sólo así, serás capaz de experimentar y mejorar tu rendimiento continuamente.

¿Qué ventajas tienen las metodologías ágiles?

Más allá de apenas ayudar en la organización del desarrollo del proyecto, las metodologías ágiles poseen otras ventajas, como:

- Agilizar entregas rápidas y continuas: una de las principales y más importantes características de las metodologías ágiles son sus entregas rápidas y continuas. Con el uso de estas metodologías, es posible determinar un intervalo de tiempo donde todos los equipos deben realizar sus entregas, que puede ser una vez a la semana o una vez a cada 15 días.

- Mejorar la calidad del producto: a través de las metodologías ágiles, los equipos enfocan sus trabajos en la búsqueda por la excelencia del producto, lo que mejora el producto final.

- Aumentar la motivación de los trabajadores: los equipos de trabajo autogestionados, facilitan el desarrollo de la capacidad creativa y de innovación entre sus miembros.

- Estimular el trabajo colectivo: con las metodologías ágiles, distintos equipos y reuniones frecuentes son necesarias. Esto permite una mejor organización del trabajo colaborativo entre equipos de distintas áreas de la empresa.

- Predecir resultados y minimizar los riesgos: gracias a revisiones continuas y a la posibilidad de cambios, es posible una mirada predictiva sobre el resultado y esto minimiza los riesgos de cometer errores inmodificables.

- Reducir los costos: la gestión continua del proyecto en las metodologías ágiles elimina la posibilidad de fracaso absoluto, una vez que los errores se van identificando a lo largo del desarrollo y pueden ser corregidos, lo que evita pérdidas.

Metodologías ágiles más utilizadas

Existen diversas metodologías ágiles, incluso muchas empresas deciden combinarlas con la finalidad de mejorar su día a día. Abajo te mostramos las más utilizadas:

Scrum: esta metodología se lleva adelante en “Sprints”, es decir, procesos de trabajo que deben ser lo más cortos posibles. Al final de cada sprint, el equipo debe entregar una versión mejorada

del proyecto para que sea analizada por todos los interesados; luego de eso, los evaluadores dan una devolución, lo que da inicio al proceso de mejora.

Kanban: Kanban es una palabra japonesa que en español significa “tarjeta visual”. Esta metodología sugiere una comunicación en tiempo real y controla el trabajo a través de una línea de producción. Es decir, se crean tres columnas: pendientes, en proceso y terminadas. De esa forma, es posible clasificar las tareas y visualizar fácilmente sus avances.

Hay otras herramientas online similares a Kanban, como Trello y Monday.

Extreme Programming o XP: esta es una metodología ágil creada para responder a ambientes muy cambiantes donde se necesita una retroalimentación permanente. Ella busca enfatizar la adaptabilidad de un proyecto, sólo así se conseguirá el resultado esperado.

Quienes trabajan con esta metodología deben entender que los cambios son inevitables y, muchas veces, más beneficiosos que un crecimiento estático.

Lectura requerida:
Gauchat, J. D. (2017). El gran libro de HTML5, CSS3 y JavaScript. Barcelona: Marcombo.

Lectura ampliatoria:
https://sites.google.com/site/programacionwebhegm. (s.f.). Obtenido de https://sites.google.com/site/programacionwebhegm/unidad-1-arquitectura/1-1-evolucion-de-las-aplicaciones-web

https://www.um.es. (s.f.). Obtenido de https://www.um.es/docencia/barzana/DAWEB/2017-18/daweb-tema-1-introduccion-html-css.html

https://www3.uji.es. (s.f.). Obtenido de https://www3.uji.es/~pacheco/INTERN~1.html