



**UNIDAD I: ANÁLISIS DE UN PROYECTO DE ESPECIALIDA
INNOVADOR**

ASIGNATURA: PROYECTO DE TÍTULO

Consideraciones Previas

Sobre las fuentes utilizadas en el material

El presente Material de Estudio constituye un ejercicio de recopilación de distintas fuentes, cuyas referencias bibliográficas estarán debidamente señaladas al final del documento. Este material, en ningún caso pretende asumir como propia la autoría de las ideas planteadas. La información que se incorpora tiene como única finalidad el apoyo para el desarrollo de los contenidos de la unidad correspondiente, respetando los derechos de autor ligados a las ideas e información seleccionada para los fines específicos de cada asignatura.

Introducción

Antes de comenzar sean muy bienvenidos a la segunda semana de la asignatura Proyecto de Título en la que continuaremos definiendo nuestro proyecto innovador dentro de la especialidad.

En esta semana, abordaremos aspectos clave en el diseño y modelado de sistemas, pilares fundamentales en el desarrollo de soluciones informáticas robustas y alineadas con las necesidades del usuario final. Los contenidos que estudiaremos esta semana se encuentran diseñados para proporcionar una visión integral de las herramientas y metodologías necesarias para planificar, representar y estructurar proyectos tecnológicos de manera efectiva.

Estudiaremos los tipos de bases de datos (SQL y NoSQL), analizando sus características, aplicaciones y cómo elegir la más adecuada para diferentes contextos, profundizaremos en la representación de procesos mediante diagramas de flujo y BPMN, herramientas esenciales para modelar las operaciones internas de un sistema y los flujos de negocio asociados a la solución propuesta.

También conoceremos la importancia del árbol funcional y los prototipos de interfaz de usuario, aplicando principios de experiencia de usuario (UX) para garantizar que las soluciones sean intuitivas, accesibles y funcionales, revisaremos metodologías de desarrollo de software, como enfoques ágiles y tradicionales, que facilitan la gestión y ejecución efectiva de proyectos informáticos.

A lo largo del estudio de esta semana, utilizaremos ejemplos prácticos y herramientas automatizadas para plasmar visualmente las soluciones propuestas, consolidando un enfoque práctico y orientado a resultados.

Ideas fuerza

En esta semana encontraremos ideas fundamentales, a las cuales debemos prestar especial atención:

- Selección de Bases de Datos (SQL y NoSQL): Comprender las características, ventajas y aplicaciones de los tipos de bases de datos para seleccionar la opción más adecuada según las necesidades y objetivos del proyecto.
- Modelado de Procesos: Utilizar diagramas de flujo y BPMN para representar procesos de manera clara, estructurada y comprensible, facilitando la comunicación entre los involucrados y la planificación efectiva.
- Diseño de Interfaces centradas en el Usuario: Crear prototipos de interfaz (mockups) aplicando principios de UX, asegurando que las soluciones sean intuitivas, accesibles y alineadas con las expectativas del usuario final.
- Árbol Funcional: Desarrollar una representación jerárquica de las funciones del sistema, facilitando la comprensión de su alcance y los flujos de navegación.
- Representación Gráfica del Diseño: Aplicar herramientas automatizadas para visualizar las estructuras y componentes del proyecto, permitiendo una validación temprana de la solución y mejorando la toma de decisiones.
- Metodologías de Desarrollo de Software: Analizar y aplicar enfoques como metodologías ágiles y tradicionales para gestionar de manera eficiente las etapas del desarrollo de proyectos informáticos, promoviendo la adaptabilidad y la colaboración en los equipos de trabajo.

Índice

Consideraciones Previas.....	2
Introducción	3
Ideas fuerza	4
Índice	5
Desarrollo	6
3. Diseño de sistemas y modelado.....	6
3.1 Tipos de bases de datos (SQL vs NoSQL) y sus aplicaciones	6
3.2 Diagramas de flujo para representar procesos.....	14
3.3 Diagramas BPMN para modelar procesos de negocio relacionados con la solución propuesta.....	16
3.4 Árbol funcional y prototipo de interfaz de usuario (mockups aplicando principios de UX).....	19
El árbol funcional.....	20
Prototipo de interfaz de usuario (mockups aplicando principios de UX)	21
Herramientas automatizadas	22
Representación gráfica del diseño de proyecto.	24
3.5 Metodologías de desarrollo de software.....	28
Conclusión	33
Bibliografía	¡Error! Marcador no definido.

Desarrollo

3. Diseño de sistemas y modelado

El diseño de sistemas y modelado constituye una etapa fundamental en el desarrollo de un sistema web, ya que permite estructurar, representar y planificar los componentes esenciales del sistema, asegurando su correcto funcionamiento y alineación con los objetivos planteados. Para la carrera de análisis de sistemas, este proceso se enfoca en aplicar metodologías y herramientas que permitan diseñar soluciones informáticas sólidas y escalables, adaptadas a las necesidades específicas del proyecto.

En esta etapa se abordan elementos clave como los tipos de bases de datos (SQL y NoSQL) y sus aplicaciones prácticas, facilitando la selección adecuada para el almacenamiento y gestión de datos. Además, se desarrolla un diagrama entidad-relación (ER) para modelar visualmente las relaciones entre las entidades del sistema, junto con un diagrama de casos de uso y actores que define las interacciones y roles dentro de la solución propuesta. Por último, se consideran metodologías de desarrollo de software que guían el proceso desde la conceptualización hasta la implementación del sistema web, garantizando un desarrollo eficiente y colaborativo.

Este enfoque integral en el diseño y modelado no solo optimiza la estructura técnica del proyecto, sino que también proporciona un marco claro para su implementación.



Figura 1: Diseño de software
Fuente: (varadero, s.f.)

3.1 Tipos de bases de datos (SQL vs NoSQL) y sus aplicaciones

En el desarrollo de sistemas web, la selección del tipo de base de datos es una decisión crucial, ya que influye directamente en el rendimiento, la escalabilidad y la estructura del sistema. Existen dos enfoques principales: bases de datos relacionales (SQL) y bases de datos no relacionales (NoSQL). Ambas tienen características distintivas que las hacen adecuadas para diferentes tipos de aplicaciones.

Bases de Datos SQL (Relacionales)

Las bases de datos SQL están basadas en un modelo relacional, donde los datos se organizan en tablas con filas y columnas. Utilizan un lenguaje estándar llamado SQL (Structured Query Language) para realizar operaciones como consultas, inserciones y actualizaciones.

Características principales:



Figura 2: Características principales bases de datos relacionales.

Fuente: (Villamar, 2024)

Aplicaciones típicas:

- Sistemas de gestión empresarial: ERP, CRM, sistemas contables.
- E-commerce: Gestión de inventarios, transacciones y clientes.
- Aplicaciones financieras: Bancos, pagos en línea y manejo de datos sensibles.

Bases de Datos NoSQL (No Relacionales)

Las bases de datos NoSQL ofrecen una estructura más flexible que no requiere un esquema fijo. Se adaptan mejor a grandes volúmenes de datos no estructurados o semi-estructurados.

Características principales:

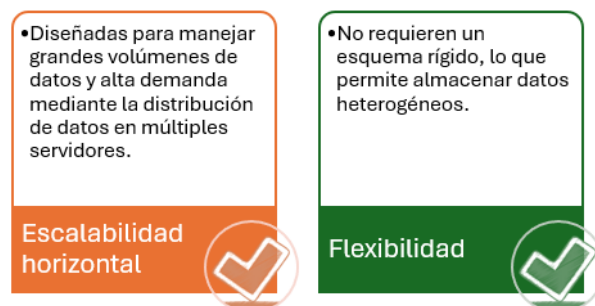


Figura 3: Características principales bases de datos no relacionales.

Fuente: (Villamar, 2024)

Tipos de bases de datos NoSQL:



Ilustración 8. Ejemplo gráfico de base de datos NoSQL de tipo documental

Figura 4: Ejemplo gráfico de base de datos NoSQL documental.

Fuente: (Lázaro, 2019).

Documentales:

Almacenan datos en formato JSON o BSON (por ejemplo, MongoDB).

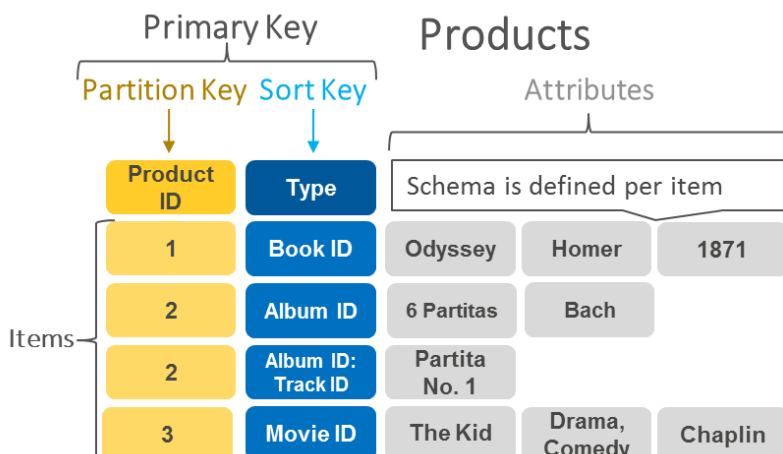


Figura 5: Ejemplo base de datos NoSQL clave-valor.

Fuente: (AWS, s.f.).

Claves-valor: Ideales para accesos rápidos basados en una clave (por ejemplo, Redis).

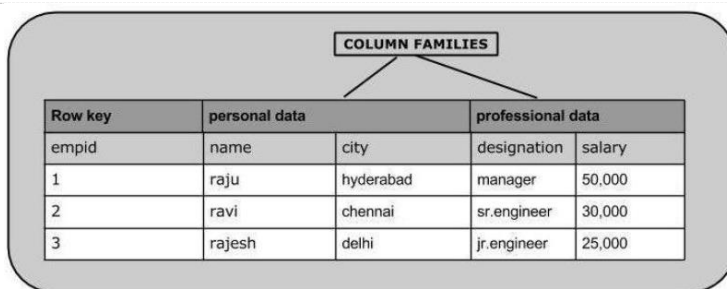


Figura 6: Ejemplo base de datos NoSQL columnas anchas.

Fuente: (AWS, s.f.).

Columnas anchas:

Organizan datos en columnas, lo que es eficiente para análisis masivos (por ejemplo, Apache Cassandra).

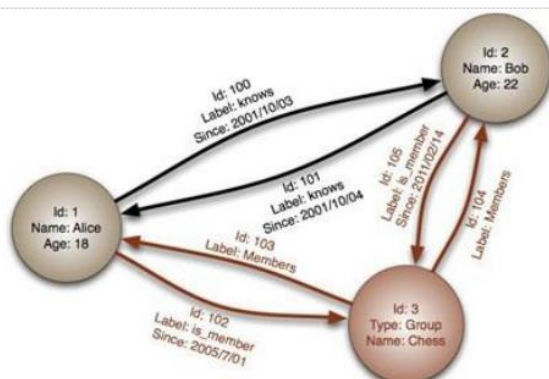


Figura 7: Ejemplo base de datos NoSQL en grafos.

Fuente: (Acens, s.f.)

Grafos: Modelan relaciones complejas entre datos (por ejemplo, Neo4j).

A continuación, revisaremos unos ejemplos que presenta (Taboada, Casal, & Fuente, 2024):

Rich Queries	<ul style="list-style-type: none"> Find Paul's cars Find everybody in London with a car built between 1970 and 1980
Geospatial	<ul style="list-style-type: none"> Find all of the car owners within 5km of Trafalgar Sq.
Text Search	<ul style="list-style-type: none"> Find all the cars described as having leather seats
Aggregation	<ul style="list-style-type: none"> Calculate the average value of Paul's car collection
Map Reduce	<ul style="list-style-type: none"> What is the ownership pattern of colors by geography over time? (is purple trending up in China?)

MongoDB

```

{
  first_name: 'Paul',
  surname: 'Miller',
  city: 'London',
  location:
    [45.123, 47.232],
  cars: [
    { model: 'Bentley',
      year: 1973,
      value: 100000, ... },
    { model: 'Rolls Royce',
      year: 1965,
      value: 330000, ... }
  ]
}
```

mongoDB

Figura 8: MongoDB: NoSQL documental.

Fuente: (Taboada, Rubén, & Fuente, gltaboada, 2024)

In-memory data structure store, útil para base de datos de login-password, sensor-valor, URL-respuesta, con una sintaxis muy sencilla:

El comando SET almacena valores
 SET server:name "luna"
 Recuperamos esos valores con GET
 GET server:name
 INCR incrementa atómicamente un valor
 INCR clients
 DEL elimina claves y sus valores asociados
 DEL clients
 TTL (Time To Live) útil para cachés
 EXPIRE promocion 60

Figura 9: Redis: NoSQL key-value.

Fuente: (Taboada, Rubén, & Fuente, gltaboada, 2024)

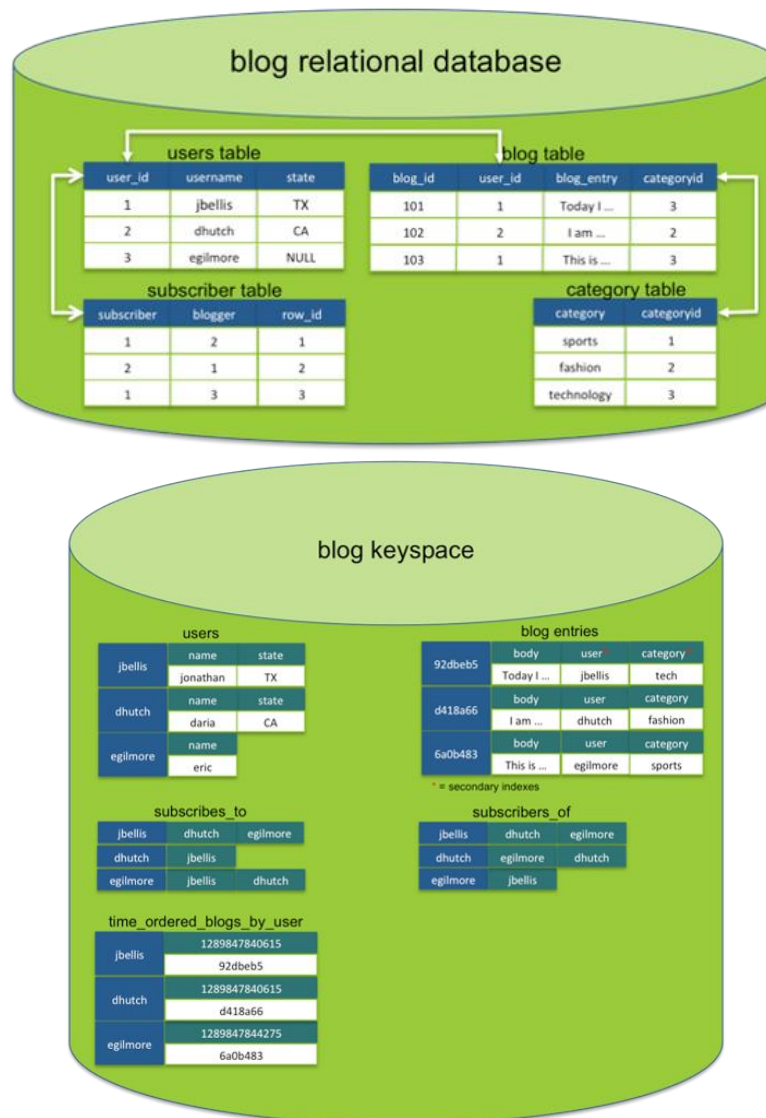
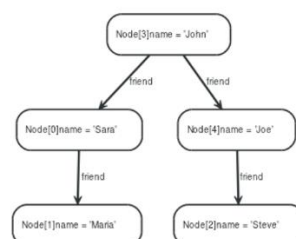


Figura 10: Cassandra: NoSQL columnar.

Fuente: (Taboada, Rubén, & Fuente, gltaboada, 2024)



```
MATCH (john {name: 'John'})-[:friend]->()-[:friend]->(fof)
RETURN john, fof
```

Figura 11: Neo4j: NoSQL grafos.

Fuente: (Taboada, Rubén, & Fuente, gltaboada, 2024)

Aplicaciones típicas:



Figura 12: Aplicaciones típicas.

Fuente: (Villamar, 2024)

La siguiente figura nos presenta un resumen con las principales diferencias entre bases de datos SQL versus NoSQL:



Figura 13: Principales diferencias entre SQL versus NoSQL.

Fuente: (Levo, 2024)

Types of Databases

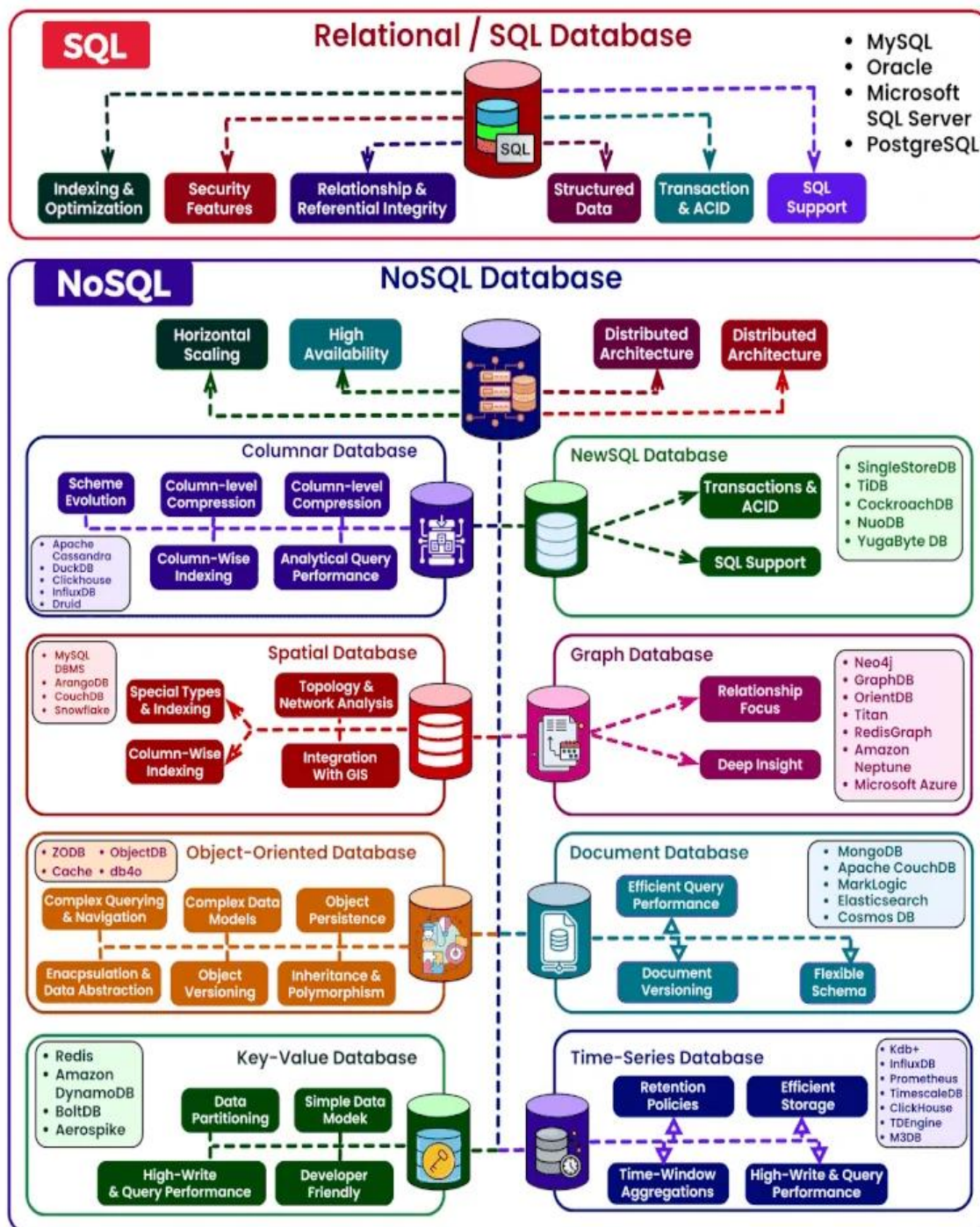


Figura 14: Tipos de base de datos SQL / NoSQL

Fuente: (reddit, s.f.)

TYPES OF DATABASES

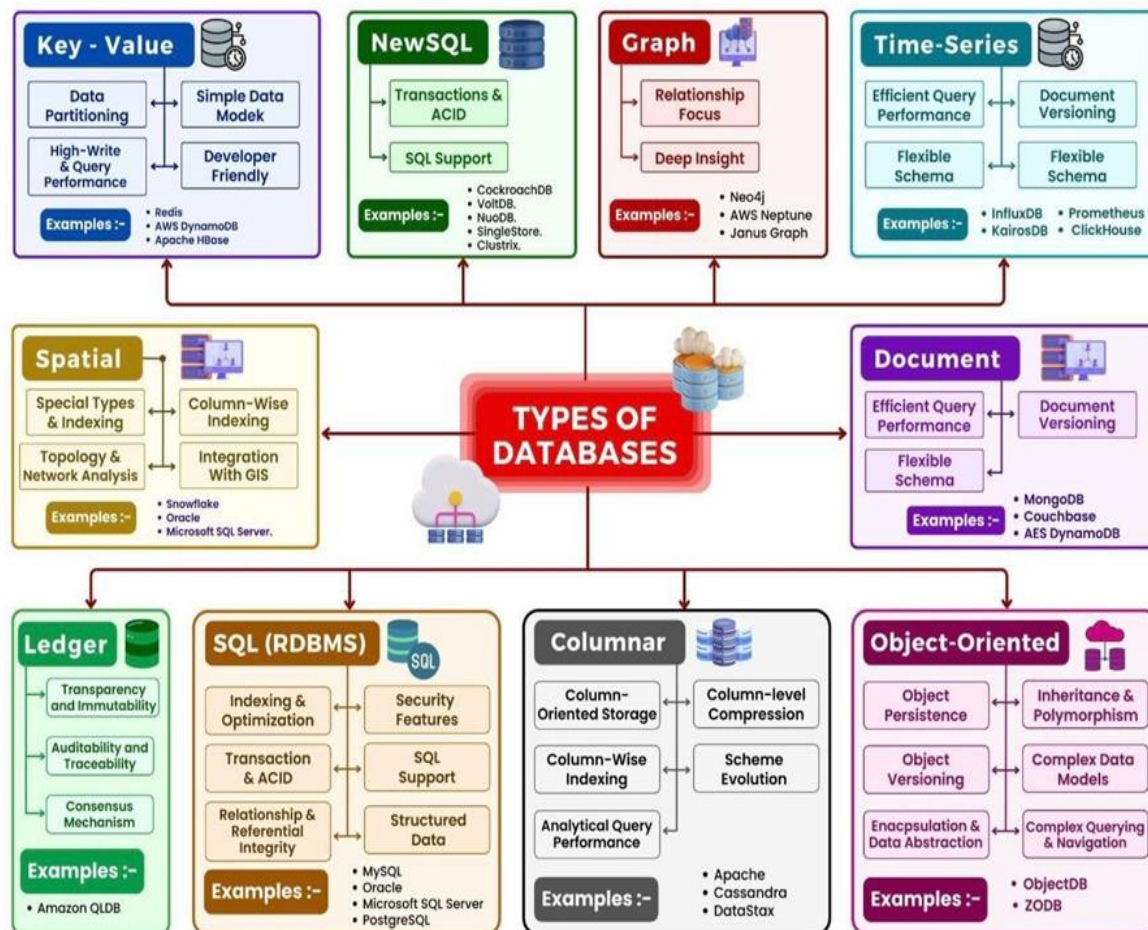


Figura 15: Tipos de base de datos
Fuente: (amigoscode, s.f.)

3.2 Diagramas de flujo para representar procesos

Los diagramas de flujo son herramientas gráficas ampliamente utilizadas para representar, analizar y comunicar procesos o sistemas de manera clara y estructurada. Mediante el uso de símbolos estandarizados, como rectángulos, flechas, rombos y óvalos, se describen los pasos secuenciales, decisiones y actividades involucradas en un proceso, facilitando su comprensión tanto para equipos técnicos como no técnicos.

En cuanto a su utilidad y aplicaciones, los diagramas de flujo son esenciales en diversos contextos, como el diseño de sistemas, la optimización de procesos, la documentación técnica y la toma de decisiones. Permiten identificar redundancias, detectar inefficiencias y proponer mejoras en procesos complejos. Además, son herramientas clave para capacitar a equipos, dado que transforman procedimientos abstractos en representaciones visuales fáciles de seguir.

En la siguiente figura encontramos los componentes principales:

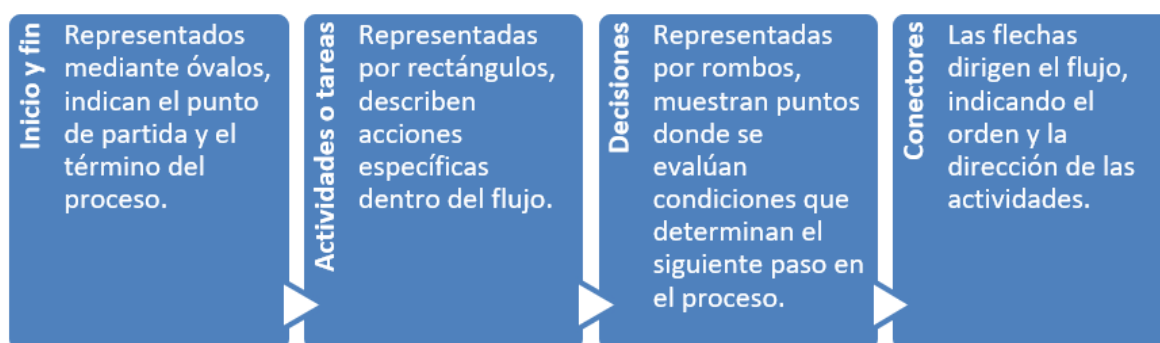


Figura 16: Componentes Principales.

Fuente: (Villamar, 2024)

En el contexto informático, los diagramas de flujo son fundamentales para mapear algoritmos, modelar la lógica de sistemas y procesos de negocio, y facilitar el entendimiento previo al desarrollo de soluciones. Su claridad y flexibilidad los convierten en un recurso valioso durante las etapas iniciales de diseño y planificación de proyectos.

El uso adecuado de diagramas de flujo no solo asegura una mejor comunicación entre los equipos de trabajo, sino que también garantiza que las soluciones propuestas se diseñen con una visión integral y estructurada.

Como menciona (Saucedo, 2015),

“En los diagramas de flujo de datos (abreviado DFD), el flujo precisamente está indicado por una línea descendente que indica la secuencia de ejecución de las instrucciones. En la figura siguiente podemos observar todos los elementos del diagrama de flujo que maneja el programa PselInt. En el siguiente capítulo iremos analizando con ejemplos la funcionalidad de cada elemento del diagrama.”

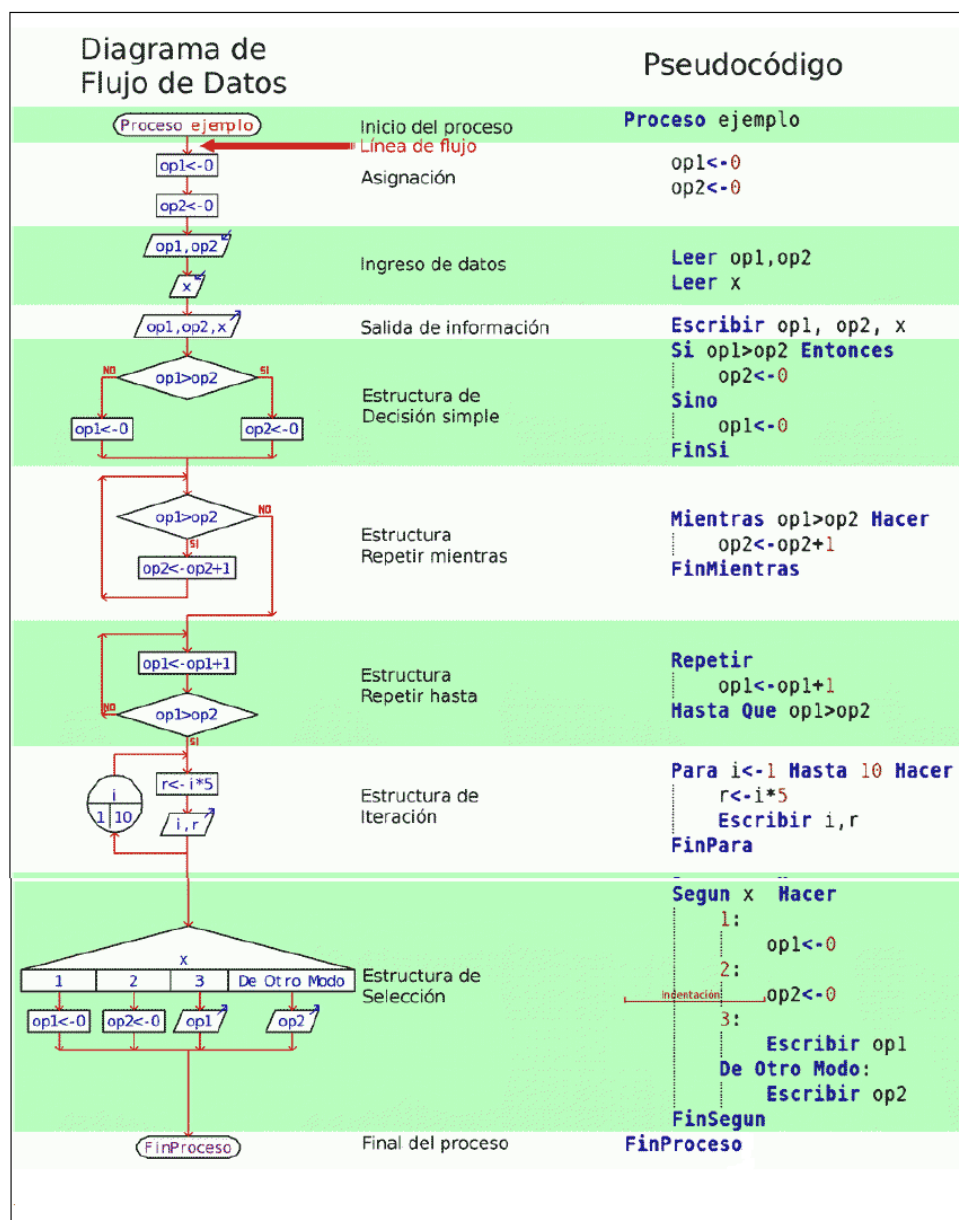


Figura 17: Componentes Principales.

Fuente: (Saucedo, 2015)

3.3 Diagramas BPMN para modelar procesos de negocio relacionados con la solución propuesta

Los diagramas BPMN (Business Process Model and Notation) son herramientas gráficas ampliamente utilizadas para modelar procesos de negocio de manera clara, estandarizada y comprensible para diferentes públicos, desde expertos técnicos hasta actores estratégicos. Su uso permite representar de manera precisa las actividades, roles, decisiones y flujos de información involucrados en un sistema o solución propuesta, facilitando el análisis, diseño y optimización de los procesos.

Podemos ver las características principales del BPMN en la siguiente figura:

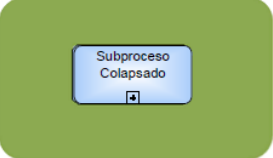
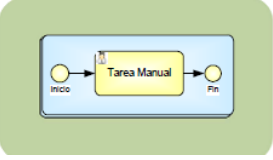












Figura 18: Características principales del BPMN.
Fuente: (Villamar, 2024)

Elementos de los diagramas según (ANALÍTICA, 2024):

Eventos	TIPO EVENTO	NOMBRE BPMN	DEFINICIÓN	NOTACIÓN
	Inicio	Start	Representa el inicio de un proceso	
	Intermedio	Intermidate	Detiene el flujo hasta que ocurra una condición o dispara acciones de excepción	
	Fin	End	Indica cuando finaliza un proceso en ejecución	

Eventos de inicio	<div>  <p>NONE</p> <ul style="list-style-type: none"> ● No tiene establecida una condición o requisito para dar inicio al proceso o subproceso </div> <div>  <p>MESSAGE</p> <ul style="list-style-type: none"> ● Un proceso o aplicativo envía un mensaje específico para dar inicio a un proceso </div> <div>  <p>TIMER</p> <ul style="list-style-type: none"> ● Se puede fijar una hora-fecha específica en la que se activará el inicio del proceso. </div>
Eventos intermedios	<div>  <p>MESSAGE</p> <p>Es usado tanto para enviar o recibir un mensaje de otros procesos o aplicativos, y debe tener el mismo nombre en el mensaje.</p> </div> <div>  <p>TIMER</p> <p>Es un mecanismo de retraso dentro del proceso. Este tiempo puede ser definido en una expresión fecha o unidad de tiempo.</p> </div> <div>  <p>LINK</p> <p>Permite conectar dos secciones de un proceso para crear situaciones de bucle o para evitar líneas de secuencia de flujo largas o cruzadas y están limitados a un nivel de proceso.</p> </div>
Eventos de fin	<div>  <p>NONE</p> <ul style="list-style-type: none"> ● No tiene establecida ninguna condición o requisito para finalizar el proceso o subproceso </div> <div>  <p>MESSAGE</p> <ul style="list-style-type: none"> ● Un proceso o aplicativo envía un mensaje específico para dar fin a un proceso. </div>
Tarea	<div>  <p>USER</p> <ul style="list-style-type: none"> ● Es una tarea donde interviene un humano para su ejecución y presenta información para la ejecución de la tarea. </div> <div>  <p>SERVICE</p> <ul style="list-style-type: none"> ● Es toda aquellas tareas que realiza el sistema sin intervención humana, como lo puede ser: enviar un email o invocar web service </div>

Subproceso	<div>  <p>●COLAPSADO</p> <p>●Los detalles del subproceso no pueden ser visualizados. El signo más (+) indica que la actividad es un subproceso y que tiene un nivel más bajo de detalle. , Esta asociado a un solo rol.</p> </div> <div>  <p>●EXPANDIDO</p> <p>●Los detalles del subproceso pueden ser visualizados, es decir, esta en el mismo nivel de detalle del proceso y tiene un evento de inicio y fin de proceso. Puede estar asociado a uno o varios roles.</p> </div>
Gateway (compuerta)	<div>  <p>EXCLUSIVA</p> <p>●Divergente: son decisiones que toma el usuario del sistema para decir el camino a seguir. ●Convergente: Sincroniza los caminos salientes, al cumplirse una condición de negocio</p> </div> <div>  <p>COMPLEJA</p> <p>●Se da en un punto del proceso donde aparecen varios caminos y solo uno de ellos es válido. Esta decisión esta basada en la información registrada en Metadata.</p> </div> <div>  <p>PARALELA</p> <p>●Indica un punto del proceso donde pueden ser llevadas a cabo actividades en forma concurrente y sincroniza los caminos que parten de una compuerta paralela</p> </div>
Objetos conectores	<div>  <p>SECUENCIA</p> <p>●Muestra el orden de los eventos, actividades y decisiones que se realizan dentro del proceso.</p> </div> <div>  <p>MENSAJE</p> <p>●Indica el flujo de mensaje entre las distintas entidades de los procesos.</p> </div> <div>  <p>ASOCIACIÓN</p> <p>●Asociar diferentes artefactos con objetos de flujo.</p> </div>
Swimlanes (canales)	<div>  <p>LANE</p> <p>●Representa un participante dentro un proceso, el cual contiene un conjunto de actividades asociadas a este rol.</p> </div> <div>  <p>POOL</p> <p>●Representa los actores externos con los cuales interactúa un proceso, estos actores pueden ser un proceso o aplicativo</p> </div>
Artefactos	<div>  <p>GRUPOS</p> <p>●Se utiliza para agrupar un conjunto de actividades, ya sea para efectos de documentación o análisis.</p> </div> <div>  <p>ANOTACIONES</p> <p>●Son un mecanismos para que el modelador pueda dar información textual adicional.</p> </div>

A continuación, revisaremos en ejemplo que nos presenta (Hitpass, 2014) Modelo de proceso de contratación de personal con cada participante en Pool propio:

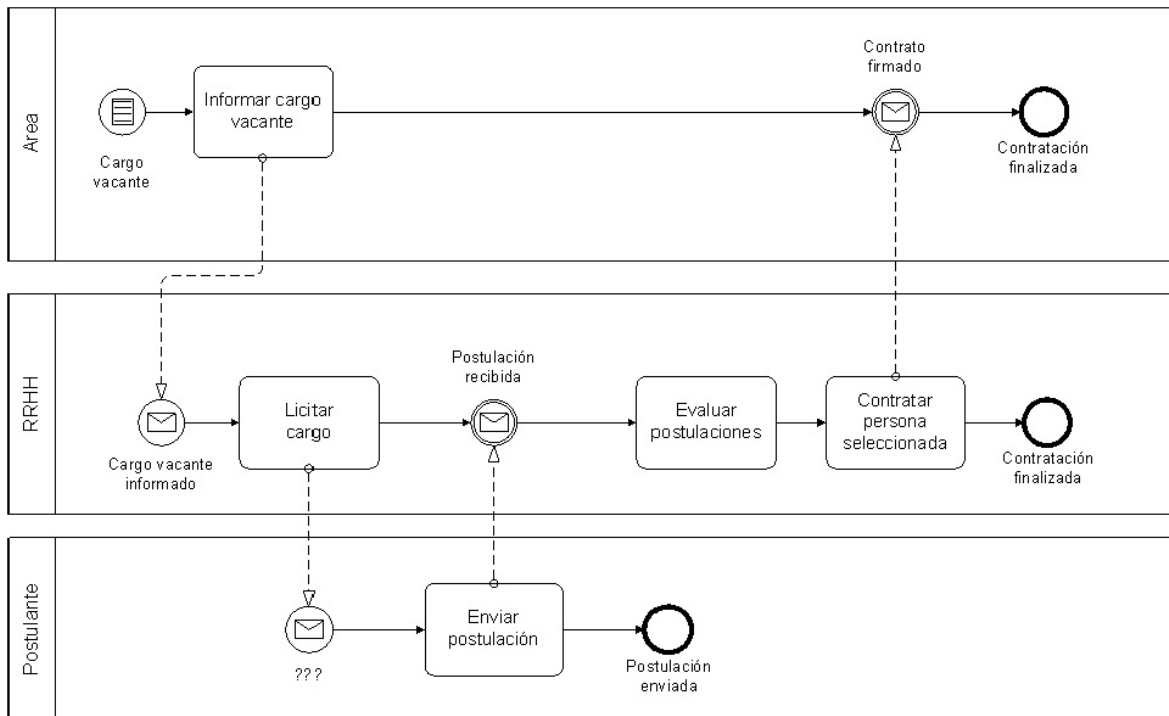


Figura 19: Modelo de proceso de contratación de personal.

Fuente: (Hitpass, 2014)

3.4 Árbol funcional y prototipo de interfaz de usuario (mockups aplicando principios de UX)

El diseño del árbol funcional y la interfaz de usuario son etapas esenciales en el desarrollo de sistemas web, ya que establecen cómo el sistema funcionará y cómo interactuarán los usuarios con él. La interfaz de usuario es el diseño gráfico y funcional que permite a los usuarios interactuar con el sistema. Incluye elementos visuales como botones, formularios y menús, así como la experiencia de uso (UX).

Aspectos clave:

Usabilidad	Consistencia	Estética y funcionalidad	Herramientas comunes para el diseño
<ul style="list-style-type: none"> La interfaz debe ser intuitiva y fácil de usar. 	<ul style="list-style-type: none"> Mantener un diseño uniforme en todas las pantallas. 	<ul style="list-style-type: none"> Diseñar elementos atractivos y funcionales. 	<ul style="list-style-type: none"> Figma, Adobe XD, Sketch.

Figura 20: Aspectos clave interfaz de usuario.

Fuente: (Villamar, 2024)

¿Cuál es la Relación entre Árbol Funcional y UI?

El árbol funcional define la estructura lógica del sistema, y la UI traduce esta estructura en una experiencia visual e interactiva. Juntos aseguran que el sistema sea funcional, navegable y centrado en las necesidades del usuario.

El árbol funcional

El árbol funcional es un método para precisar y categorizar las funciones de un sistema. Se comienza por uno o varios objetivos clave que se desagrega continuamente en funciones, subfunciones hasta que un determinado nivel no se pueda descomponer en más elementos que puedan ser redactados como funciones.

El árbol funcional es una representación jerárquica que organiza las funcionalidades del sistema en niveles. Cada nodo del árbol representa una funcionalidad o módulo, y sus subnodos, las funciones asociadas.

Propósito:



Figura 21: Propósito árbol funcional.

Fuente: (Villamar, 2024)

Todo elemento del sistema ejecuta sus propias funciones, y el sistema también realiza funciones con su entorno.

Como todas las herramientas, el árbol funcional tiene ventajas y desventajas. Es una herramienta simple de explicar y de utilizar, pero sus pocas reglas permiten que para un mismo propósito se logren resultados muy distintos, en todo caso, su uso es bastante limitado y cuenta con objetivos muy claros, como son:

- Definir las funciones que debe desarrollar el sistema.
- Identificar modos de fallo para esas funciones.

Ahora bien, si lo aplicamos en un desarrollo web podemos definir cuantas secciones tendrá el sitio y con cuantos niveles contará, agrupar los contenidos en función de su lógica de actividad o tareas dentro del sistema. La figura 10 presenta ejemplos de árbol funcional:

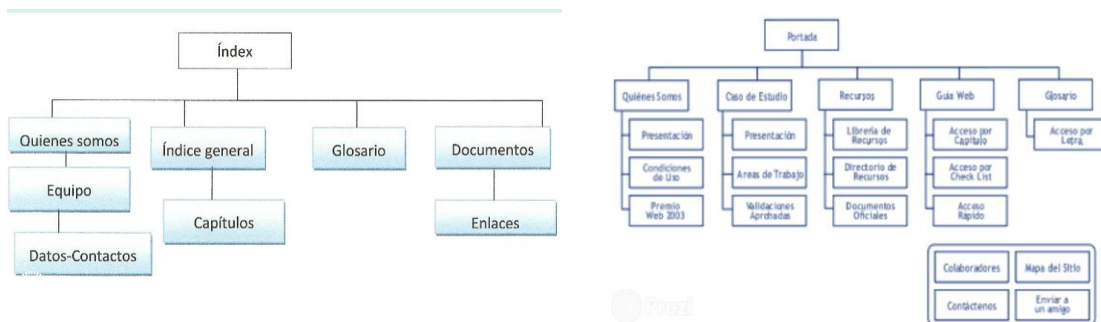


Figura 22: Ejemplo Árbol funcional.
Fuente: (Villaseñor, 2013)

Prototipo de interfaz de usuario (mockups aplicando principios de UX)

El diseño de un prototipo de interfaz de usuario (UI) es una etapa fundamental en el desarrollo de soluciones tecnológicas, ya que permite representar visualmente cómo los usuarios interactuarán con el sistema. Los mockups, como prototipos de media o alta fidelidad, son herramientas gráficas que combinan diseño visual con funcionalidad simulada, proporcionando una visión cercana al producto final. Cuando se desarrollan aplicando principios de experiencia de usuario (UX), estos prototipos garantizan interfaces intuitivas, accesibles y centradas en las necesidades del usuario.

Características de los Mockups en el Contexto de UX

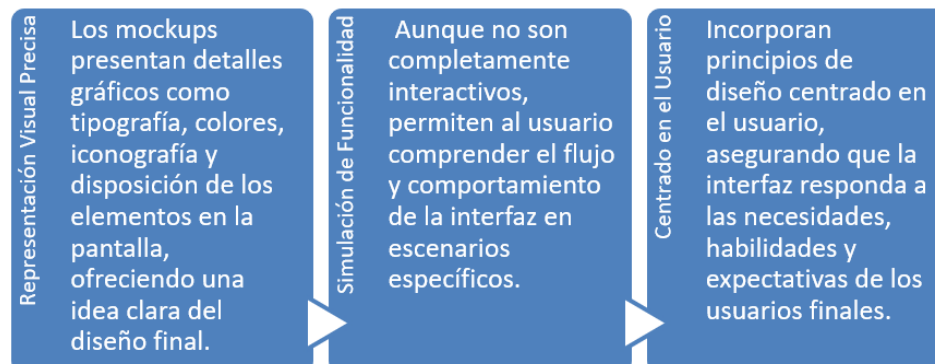


Figura 23: Características de los Mockups en el Contexto de UX.
Fuente: (Villamar, 2024)

Aplicación de Principios de UX en el Desarrollo de Mockups:

- 1 Usabilidad: Diseñar interfaces que sean intuitivas y fáciles de navegar, reduciendo la curva de aprendizaje para los usuarios.
- 2 Consistencia Visual: Garantizar una uniformidad en estilos, colores, tipografía y disposición para proporcionar una experiencia fluida y coherente.
- 3 Jerarquía de la Información: Organizar los elementos de la interfaz según su importancia, utilizando técnicas como contraste y tamaño para guiar la atención del usuario hacia las acciones principales.
- 4 Accesibilidad: Asegurar que los mockups consideren principios de diseño universal, permitiendo que la interfaz sea utilizable por personas con diferentes capacidades.
- 5 Pruebas y Retroalimentación: Permitir iteraciones basadas en pruebas con usuarios reales para identificar áreas de mejora y ajustar el diseño antes del desarrollo final.

Figura 24: Aplicación de Principios de UX.

Fuente: (Villamar, 2024)

Los prototipos de interfaz de usuario, al proporcionar una representación clara del producto final, facilitan la comunicación efectiva entre desarrolladores, diseñadores y otras partes interesadas, además, permiten la detección temprana de problemas de diseño, evitando costos adicionales al identificar y resolver inconvenientes antes de avanzar en el desarrollo técnico. Estos prototipos también son esenciales para validar los requerimientos del proyecto, asegurando que la solución propuesta cumpla con los objetivos establecidos y las expectativas del usuario, fomentan la colaboración al actuar como una herramienta clave para la discusión y toma de decisiones durante las etapas iniciales, optimizando así el proceso de desarrollo.

Herramientas automatizadas

En el proceso de desarrollo de software la creación de prototipos resulta ser una parte fundamental en el diseño, entregando la posibilidad de revisar conceptos y compartir información durante las primeras fases del desarrollo.

La ventaja de crear un prototipo durante el desarrollo de un sitio web, un sistema o una aplicación móvil, es que se podrán identificar deficiencias en el funcionamiento y la funcionalidad del diseño antes de invertir una cantidad elevada de tiempo o dinero en el desarrollo.

Si bien es cierto en el momento de diseñar prototipos se cuenta con distintas necesidades que deben ser cubiertas, se cuenta con herramientas que brindan funcionalidad y flexibilidad necesarios para la creación de conceptos interactivos. A continuación, revisaremos algunas herramientas automatizadas empleadas en la creación de prototipos:

 <p>Figura 25: Logo y pantalla Moqups. Fuente: (moqups, s.f.)</p> <p>Enlace de descarga: https://www.moqups.com/</p>	<p>moqups Moqups Una aplicación web optimizada que ayuda a crear y colaborar en tiempo real en estructuras alámbricas, maquetas, diagramas y prototipos. Fuente: (moqups, s.f.)</p>
 <p>Figura 26: Logo y pantalla Marvel. Fuente: (marvelapp, s.f.)</p> <p>Enlace de descarga: https://marvelapp.com</p>	<p>Marvel Marvel: tiene todo lo que necesitas para dar vida a tus ideas y transformar la forma en que creas productos digitales con tu equipo. Poniendo el poder del diseño en manos de todos. Fuente: (marvelapp, s.f.)</p>
	<p>JUST IN MIND JustInMind: Herramienta de diseño y creación de prototipos para aplicaciones web y móviles. Plataforma de diseño de UI y UX todo en uno para crear activos, prototipos y simulaciones de UI.</p>

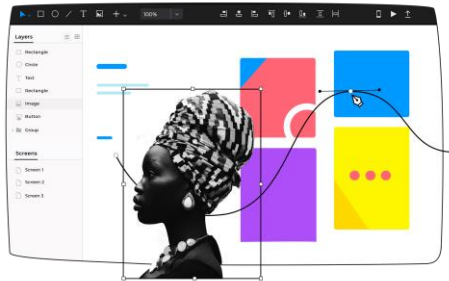


Figura 27: Logo y pantalla Just in Mind.

Fuente: (justinmind, s.f.)

Enlace de descarga: <https://www.justinmind.com/>

Fuente: (justinmind, s.f.)

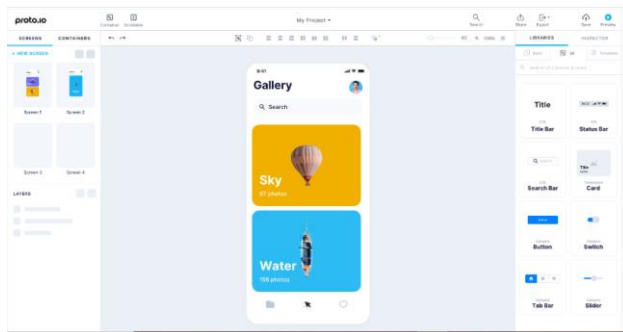


Figura 28: Logo y pantalla Proto.

Fuente: (proto, s.f.)

Enlace de descarga: <https://proto.io/>

proto.io **Proto:** Prototipos para todos. Da vida a tu idea en poco tiempo. La solución de creación de prototipos para todas sus necesidades. Para diseñadores de UX, empresarios, gerentes de producto, especialistas en marketing y cualquier persona con una gran idea. Fuente: (proto, s.f.)

Representación gráfica del diseño de proyecto.

La representación gráfica la podemos realizar por medio de los prototipos, como hemos visto resultan importantes ya que nos permiten tener una documentación visual de lo que se necesita desarrollar, permite medir la dimensión del sistema, realizar estimados realistas, lo que da la posibilidad de contar con un desarrollo organizado y seguro, si hacemos un símil es lo equivalente a los planos en la construcción.

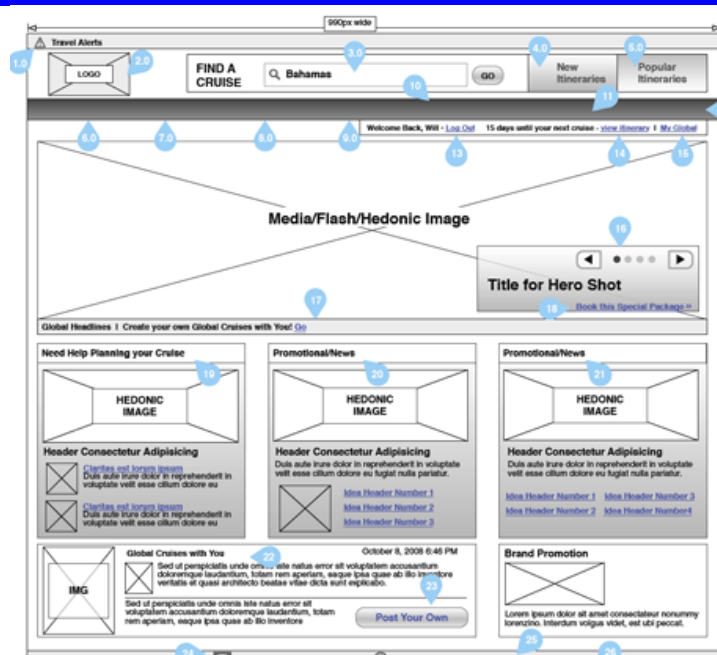


Figura 29: Ejemplo diseño representación gráfica, pantalla.
Fuente: (vexsoluciones, s.f.)



Figura 30: Ejemplo diseño representación gráfica, pantalla.
Fuente: (vexsoluciones, s.f.)



- 1 For Q1 release, music search only
- 2 Related artists determined by user purchasing data mining
- 3 Album art to be approved by legal

Figura 31: Ejemplo diseño representación gráfica, pantalla.
Fuente: (vexsoluciones, s.f.)

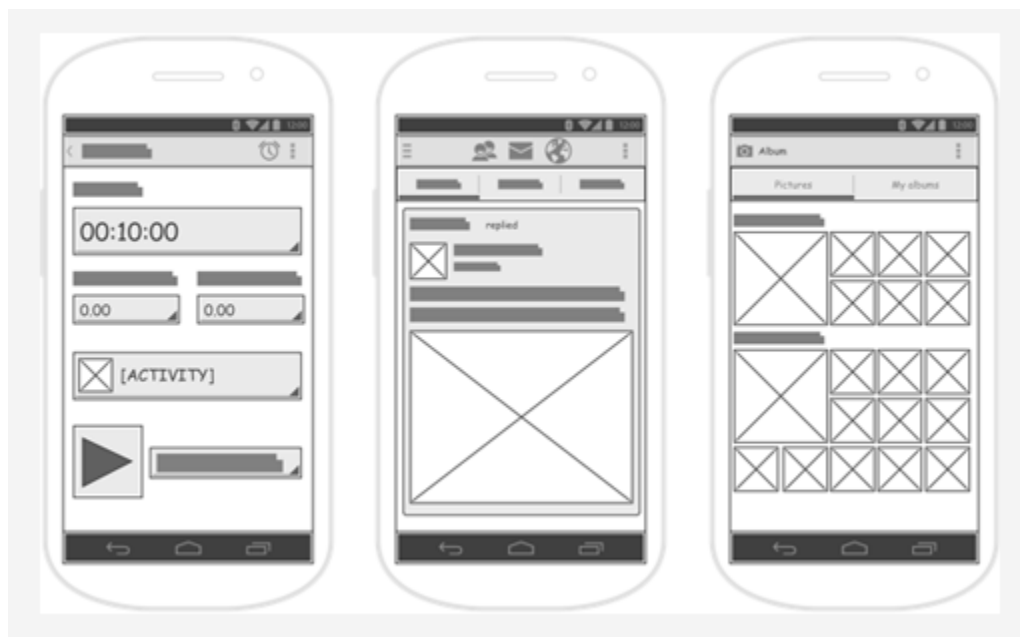


Figura 32: Ejemplo diseño representación gráfica, pantalla móvil.
Fuente: (vexsoluciones, s.f.)

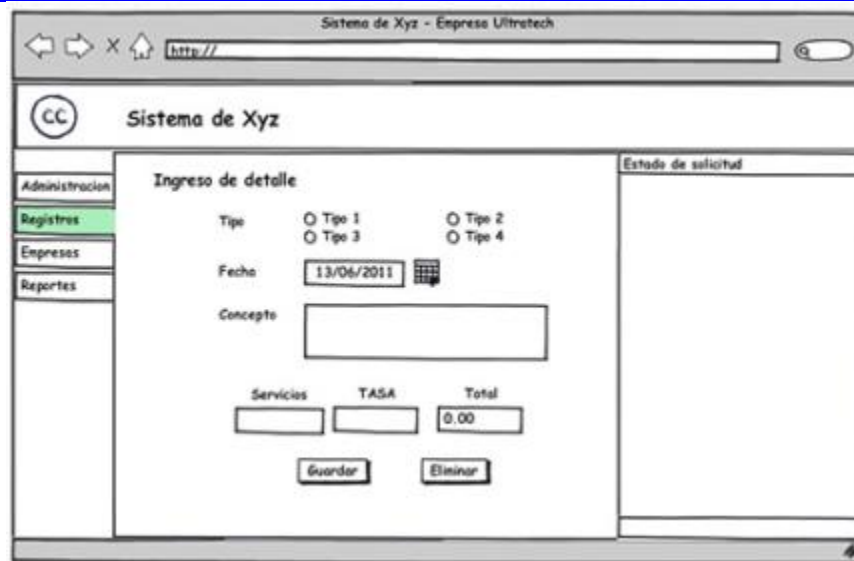


Figura 33: Ejemplo diseño representación gráfica, pantalla.
Fuente: (sv.all.biz, s.f.)



Figura 34: Ejemplo diseño representación gráfica, mockup incluye elementos de sketch y wireframe.
Fuente: (mosaic.uoc.edu/, s.f.)

3.5 Metodologías de desarrollo de software

Las metodologías de desarrollo de software son enfoques estructurados que guían el proceso de creación de aplicaciones o sistemas, asegurando una planificación eficiente, una ejecución organizada y la entrega de productos de alta calidad. Estas metodologías establecen marcos de trabajo que incluyen prácticas, principios y herramientas diseñadas para gestionar proyectos de manera efectiva, desde la recopilación de requisitos hasta la implementación y mantenimiento del software.

Entre las más utilizadas se encuentran las metodologías tradicionales, como el modelo en cascada, que sigue una secuencia lineal de fases; y las ágiles, como Scrum o Kanban, que promueven la iteración, la adaptabilidad y la colaboración en equipos multidisciplinarios. Cada metodología se elige según las características del proyecto, el entorno de desarrollo y las necesidades específicas del cliente o usuario final. Su correcta aplicación mejora la comunicación entre los equipos, minimiza los riesgos y asegura la alineación del producto con los objetivos planteados.

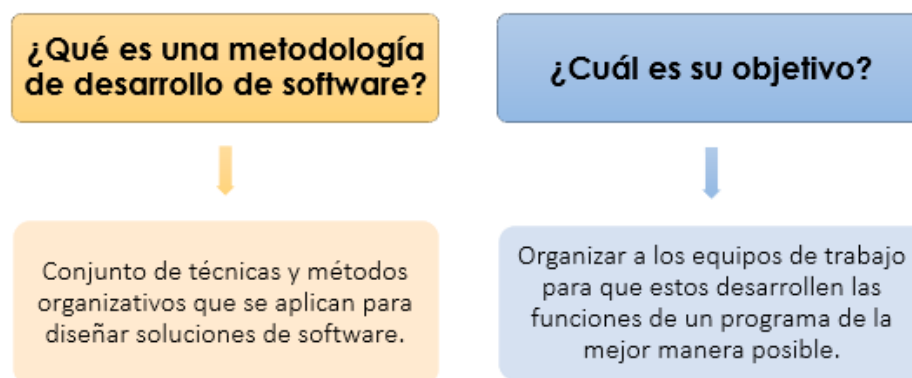
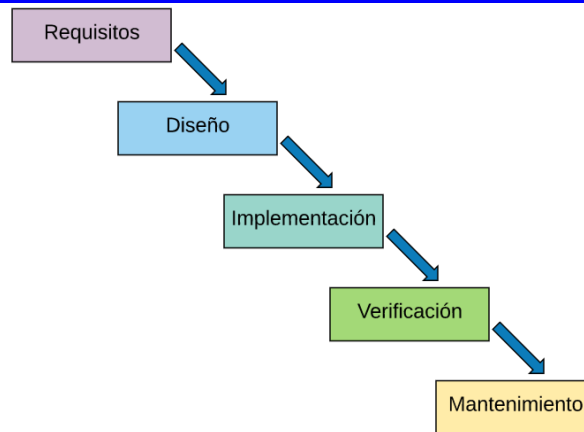


Figura 35: Metodología de desarrollo, definición y objetivo.

Fuente: (Villamar, 2022)

¿Cómo funcionan las metodologías clásicas?

Cuando se aplica una metodología clásica en el desarrollo de un sistema, se sigue una secuencia lógica y cada etapa depende de que finalice la etapa anterior para comenzar.



Las etapas del modelo en cascada

Figura 36: Etapas modelo clásico

Fuente: (Leon, 2018)

Hoy en día se dispone de una gran cantidad de metodologías para el desarrollo de software, las que podemos clasificar en dos grandes grupos:

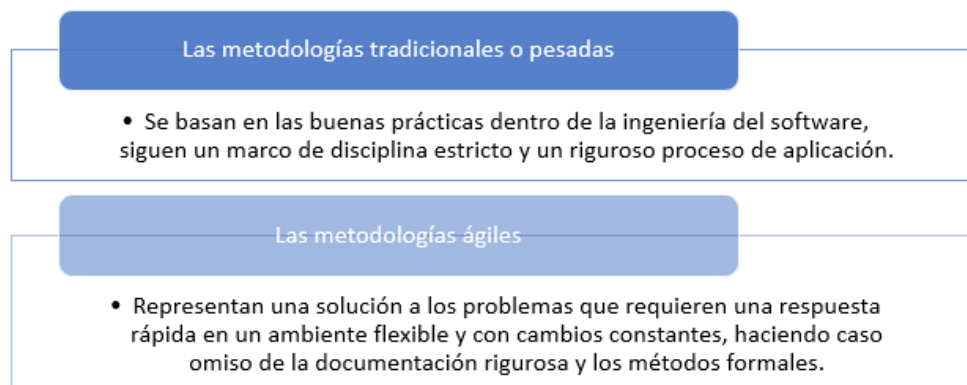


Figura 37: Clasificación de las metodologías de desarrollo

Fuente: (Villamar, 2022)



Figura 38: Cuadro comparativo metodologías.
Fuente: (Universidades, 2020)

Las metodologías ágiles: Se desarrollan de manera iterativa, repitiéndose etapas cortas en las que se realizan pequeñas secciones del proyecto.



Figura 39: Como implementar metodología ágil en 7 pasos.
Fuente: (Roca, s.f.)

Diferencia entre modelos ágiles v/s modelos no ágiles

Parámetros	Modelo Ágil	Modelos no Ágil
Enfoque de esta metodología	Muy flexible y ajustable Se puede adaptar a las necesidades del proyecto	No es tan flexible como ágil Difícil acomodar los cambios en el proyecto
Medición del éxito	Se mide por el valor comercial entregado	Se mide por la conformación a planear
Tamaño del proyecto	Generalmente pequeño	Grande
Estilo de gestión	No centralizado Se distribuye entre los integrantes del equipo	Dictatorial Solo una persona toma decisiones y el resto lo sigue
Capacidad de adaptación al cambio	Se aceptan cambios Se adapta según necesidades del proyecto	Cambios no se aceptan fácilmente en las ultimas etapas del desarrollo
Documentación requerida	Menor documentación	Mayor cantidad de documentación

Figura 40: Modelos ágiles v/s modelos no ágil.

Fuente: (Villamar, 2022)

Siete criterios a tomar en cuenta para definir si un determinado proyecto debe utilizar metodología ágil o metodología cascada.

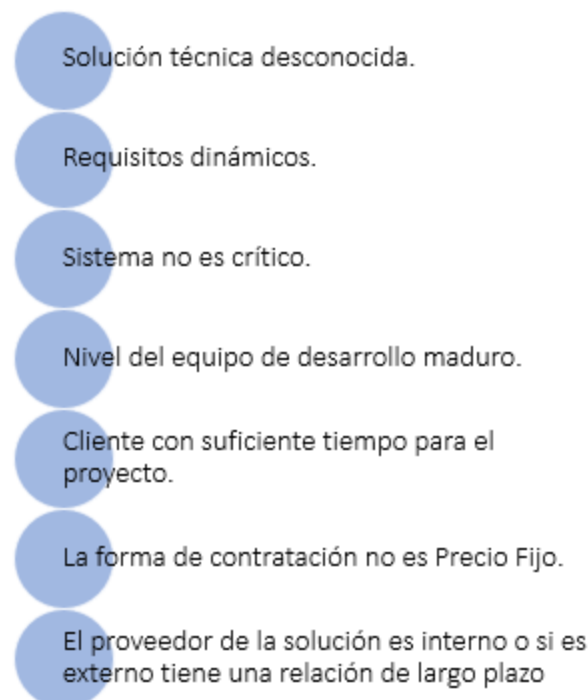


Figura 41: Como implementar metodología ágil en 7 pasos.

Fuente: (Lazarte, 2018)

Preguntas de reflexión

¿Cómo puede el diseño del árbol funcional garantizar que todas las funcionalidades necesarias del sistema estén organizadas de manera lógica y accesible para los usuarios?

¿De qué manera una interfaz de usuario bien diseñada puede influir en la experiencia del usuario y el éxito del sistema web en su entorno de uso?



Conclusión

A lo largo de esta semana, hemos estudiado elementos clave en el diseño, modelado y desarrollo de sistemas, destacando su importancia en la construcción de soluciones tecnológicas efectivas. La comprensión de las diferencias entre bases de datos SQL y NoSQL nos ha permitido identificar aplicaciones específicas según las necesidades del proyecto, mientras que el uso de diagramas de flujo y BPMN nos proporciona herramientas visuales para modelar procesos de manera clara y colaborativa.

Además, el diseño centrado en el usuario, a través de mockups y árboles funcionales, garantiza interfaces intuitivas y funcionales que responden a las expectativas y necesidades de los usuarios. Complementando estas técnicas, las herramientas automatizadas para la representación gráfica del diseño del proyecto y las metodologías de desarrollo de software consolidan un marco estructurado para la planificación, ejecución y validación de proyectos.

En conjunto, estos conceptos fortalecen nuestra capacidad para abordar el diseño y desarrollo de sistemas desde una perspectiva integral, alineada con las mejores prácticas y orientada al éxito de los proyectos informáticos.

Referencias

- Acens. (s.f.). Obtenido de <https://www.acens.com/comunicacion/wp-content/images/2014/02/bbdd-nosql-wp-acens.pdf>
- amigoscode. (s.f.). Obtenido de <https://www.threads.net/@amigoscode/post/Cul2XpLABYo>
- ANALÍTICA. (2024). *Sistema de Gestión de Procesos*. Obtenido de https://www.analitica.co/website/images/stories/documentosTecnicos_SGP/Manual%20de%20Diagramacion%20de%20Procesos%20Bajo%20Estandar%20BPMN.pdf
- AWS. (s.f.). Obtenido de <https://aws.amazon.com/es/nosql/key-value/>
- Hitpass, B. (2014). *BPMN 2.0 Manual de Referencia y Guía Práctica*. Santiago de Chile: Edición hispana.
- justinmind. (s.f.). Obtenido de <https://www.justinmind.com/>
- Lázaro, H. I. (20 de febrero de 2019). Obtenido de <https://bddocumentales.blogspot.com/2019/02/bases-de-datos-documentales-una-base-de.html>
- Lazarte, R. (21 de 05 de 2018). Obtenido de <https://www.linkedin.com/pulse/siete-criterios-para-definir-cu%C3%A1ndo-usar-metodolog%C3%ADa-da-ruben/?originalSubdomain=es>
- Leon, A. I. (2018). Obtenido de <https://ivan395.github.io/Web/modelos.html>
- Levo, L. (16 de 05 de 2024). *Levo Learning*. Obtenido de <https://levolearning.edu.pe/shop/marvelapp>
- marvelapp. (s.f.). Obtenido de <https://marvelapp.com/>
- moqups. (s.f.). Obtenido de <https://www.moqups.com/>
- mosaic.uoc.edu/. (s.f.). Obtenido de <https://mosaic.uoc.edu/2015/09/15/proceso-de-desarrollo-de-un-proyecto-digital/>
- proto. (s.f.). Obtenido de <https://proto.io/>
- reddit. (s.f.). Obtenido de https://www.reddit.com/r/coolguides/comments/17wi130/a_cool_guide_to_sql_vs_nosql_databases/?show=original
- Roca, C. (s.f.). *thepowermba*. Obtenido de <https://www.thepowermba.com/es/blog/implementar-metodologias-agiles>
- Saucedo, R. (2015). *Manual de Programación Lógica*. julio: 30.
- sv.all.biz. (s.f.). Obtenido de <https://sv.all.biz/prototipos-de-software-s117>
- Taboada, G. L., Casal, R. F., & Fuente, M. O. (2024). *Prácticas de Tecnologías de Gestión y Manipulación de Datos*. 09: 15.
- Taboada, G. L., R. F., & Fuente, M. O. (15 de 09 de 2024). *gltaboada*. Obtenido de <https://gltaboada.github.io/tgdbook/index.html>
- Universidades, S. (21 de 12 de 2020). Obtenido de <https://www.becas-santander.com/es/blog/metodologias-desarrollo-software.html>
- varadero. (s.f.). Obtenido de <https://www.varadero.es/servicios-it/desarrollo-de-software/>
- vexsoluciones. (s.f.). Obtenido de <https://www.vexsoluciones.com/desarrollo-prototipos-de-software/>
- Villaseñor, A. (03 de 11 de 2013). Obtenido de <https://prezi.com/bmohxejf556j/arbol-funcional/>



 **Iplacex**
enovus

4 INSTITUCION
ACREDITADA
NIVEL AVANZADO
AÑOS Hasta octubre 2025


Comisión Nacional
de Acreditación
CNA - Chile

GESTIÓN INSTITUCIONAL Y DOCENTE DE PREGRADO