

Alex Stryer Diaz  
A01707173  
10/6/2025

### Lecturas(Preguntas en la Hoja):

Consultas en SQL usando Roles y Subconsultas

Roles: aparece cuando una tabla aparece más de una vez en una consulta

```
SELECT idviaje, origen.nombre, destino.nombre, Roles
  FROM viajes, ciudades origen, ciudades destino
 WHERE viajes.idorigen = origen.idciudad
   AND viajes.iddestino = destino.idciudad;
```

CREATE SYNONYM origen FOR ciudades;

Consultas reflexivas (Recursivas)

```
SELECT e.nombre empleado, j.nombre jefe
  FROM empleados e, empleados j
 WHERE e.jefe = j.idempleado;
```

CREATE SYNONYM jefes FOR empleados

Subconsultas:

- Comparar filas vs conjuntos
- Filtrar por condiciones complejas
- Comparar contra valores agregados

```
SELECT idproducto, descripcion
  FROM productos
 WHERE idproducto NOT IN (SELECT idproducto
                            FROM ventasdetalle
                           WHERE NOT EXISTS
                                 WHERE 1000 > (
```

operadores:

Sintaxis: 1. Select      • ANY • SOME • ALL  
2. FROM  
3. WHERE

Comparadores: =, !=, <, >, <=, >=

Step three: Identify data values to test

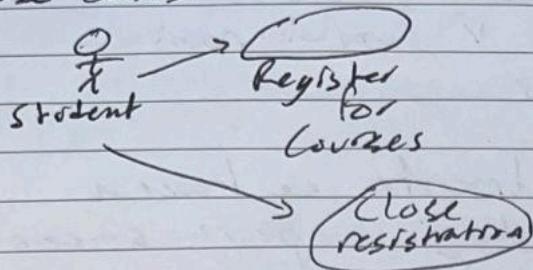
Matrix with data values  
change the V with real  
data values.

Pregunta: ¿Realmente se hacen  
procesos tan largos para sacar  
una buena app?

ID	Barcode	Stolen	Password	Course	Prerequisites	Course Name	Schedule Type	Enrollment	Open	Result
P15	Kunst	No	123456	Math 101	None	Math 101	Evening	Yes	Full	Pass
P16	enroll	No	123456	Math 101	None	Math 101	Morning	Yes	Full	Pass
P17	Pharma	Yes	123456	Chem 101	None	Chem 101	Evening	Yes	Full	Pass
P18	Pharma	Yes	123456	Chem 101	None	Chem 101	Morning	Yes	Full	Pass

Metodología para diseñar casos de prueba a partir de casos de uso

Use Cases:



Use Case:

- Name
- Brief Desc
- Flow of events
- Special requirements
- Preconditions
- Post conditions

Test Cases

Step 1: identify main and alt flows

Name	Flow	Alt
Registration	Basic Flow	A1
Unidentified	Basic Flow	A2

Step 2: Identify test cases

ID Scenario	Student ID	Password	Courses Selected	Prerequisites	Course Done	Expectation Result
A1	≈	V	V	V	V	xx
A2	≈	I	N/A	N/A	N/A	N/A

## Normalización

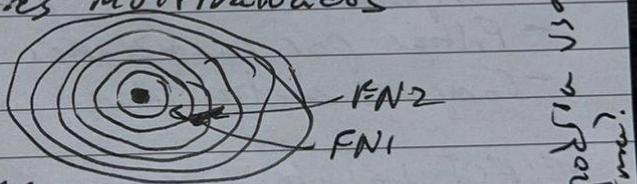
Que es?:

Es el proceso de estructurar los datos para evitar redundancias, errores y dependencias innecesarias, dividiendo las relaciones en tablas más pequeñas y específicas

- Eliminar anomalías
- Mejorar integridad de datos
- Mantenimiento
- Evitar redundancia e inconsistencias

Pasos:

1. Descomponer los datos en tablas simples
2. Eliminar lo que no depende de la clave primaria
3. Eliminar dependencias transitivas
4. Eliminar valores multivaluados



## Formas Normales

1FN Celdas atómicos, sin repeticiones

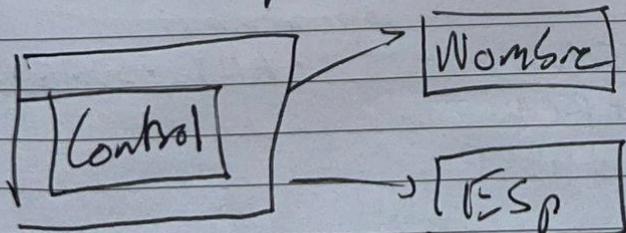
2FN Dependencia total de la clave primaria

3FN Elimina dependencias transitivas

4FN BCNF todo determinante (clave candidata)

4FN Elimina valores multivaluados

5FN Elimina dependencias de producto



Algebra relacional, SQL básico y  
funciones agregadas

Algebra relacional es la teoría de  
los lenguajes de consulta

1. Ø → SELECT \* FROM R WHERE  
condición;
2. Π → Select columna1, columna2  
FROM R;
3. ∪ → SELECT... FROM R UNION  
SELECT...;
4. - → Except / MINUS
5. ∩ → INTERSECT
6. × → SELECT \* FROM A, B;

1. SN

2. SL

3.

Filtrar ↗

Derivar otras  
columnas ↗

Combinar  
tuplas ↗

En una tabla  
pero no en otra

tuplas  
comunes  
en 2 tablas ↗

Combinación ↗  
todas las de  
con B

¿En algún punto la álgebra relacional  
será obsoleta?

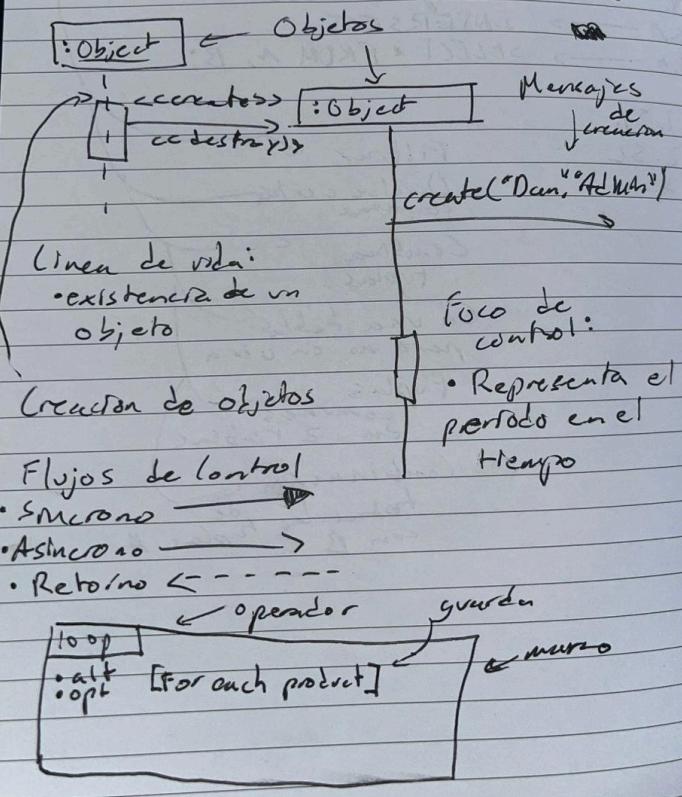
### Diagramas de secuencia

- Diagrama de interacción, muestra el flujo temporal de mensajes entre objetos en un sistema
- Se usa para modelar escenarios específicos como casos de uso

### Estructura

eje x = objetos involucrados  
eje y = mensajes ordenados

¿Esra ideal uno  
enorme?



## Modelo Relacional

### Algebra relacional

- **Dominio:** conjunto de valores válidos
- **Atributo:** toma valores de un dominio
- **Clave primaria:** identificador de forma única
- **Clave compuesta:** varias columnas para garantizar unicidad

### Clave Foranea:

- Atributo que referencia la Clave primaria de otra tabla

### Algebra relacional

$$atb = s \quad a, b \quad S, P$$

$$ab = p$$

### Leyes:

$a+b = b+a$  Comutatividad de la suma

$ab = ba$  Comutatividad del producto

$(a+b)+c = a+(b+c)$  Asociatividad de la suma

$a(b+c) = ab+ac$  Distributividad del producto

Lenguaje formal que actúa sobre tablas

### Operadores:

- Unión ( $\cup$ )      • Producto cartesiano ( $\times$ )
- Intersección ( $\cap$ )      • Diferencia ( $\setminus$ )
- Diferencia ( $-$ )
- Proyección ( $\pi$ )
- Selección ( $\sigma$ )

¿Cuando conviene usar Theta-Join?

## Gestión de la Comunicación

### Procesos

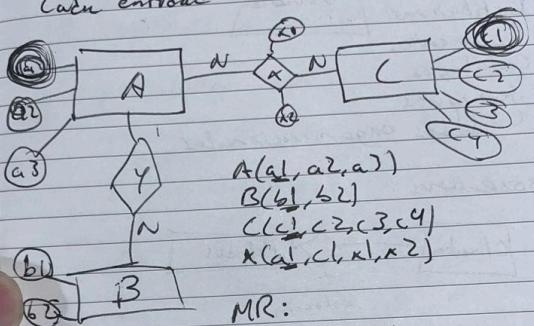
1. Los interesados (Identificar)
  - interno/externo
2. Planear la comunicación
  - definir medios, canales, responsables
3. Distribuir la info
  - Presentaciones, reportes, reuniones
4. Administrar expectativas
  - Negociación, atención, aclaraciones
5. Reportar
  - Informar sobre avances
  - Estado actual
  - Riesgos

¿Como se maneja la situación si hay un cambio de enfoque?

## Traslado MER a IER en tablas

Es necesario trasladar el MER a IER para implementar una base de datos relacional

Procedimiento:  
Cada entidad → tabla (PK se subraya)

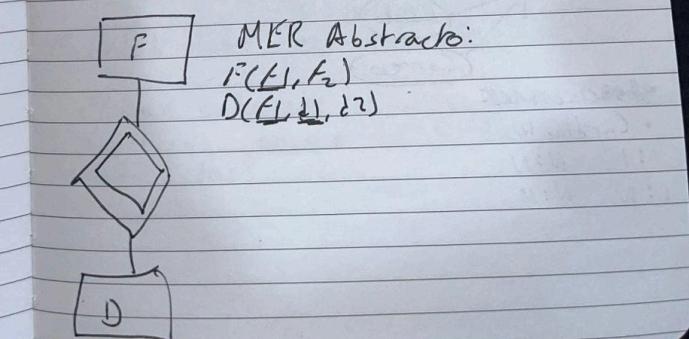


MR:

$A(\underline{a}_1, a_2, a_3)$   
 $B(\underline{b}_1, b_2, a_1)$   
 $C(c_1, c_2, c_3, \underline{c}_4)$   
 $K(k_1, k_2, \underline{x}_1, x_2)$

MER Abstracto:

$I^*(\underline{f}_1, f_2)$   
 $D(\underline{f}_1, \underline{f}_2, f_2)$



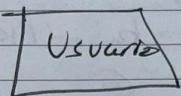
## Notación M.E.R

- Crear esquemas sin redundancia
- Minimizar problemas

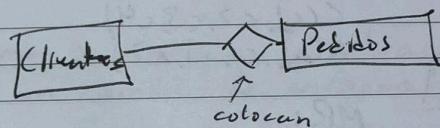
## Componentes MER

### • Entidades:

- Personas
- Instituciones
- Documentos
- Catálogos
- = Secciones
- Unidades organizacionales

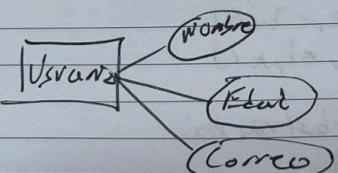


### • Asociaciones:



### • Atributos:

- Las características de la entidad



### • Atributos:

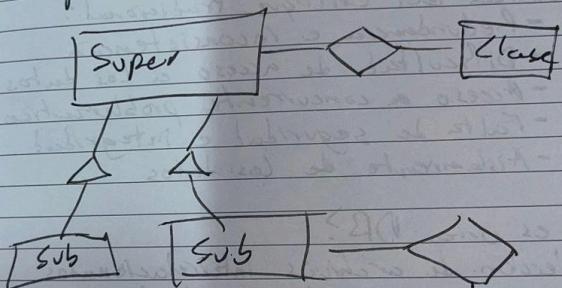
- Cardinalidad

1:1    N:N

1:N    N:M

- Metodología para MÉR
1. Identificar entidades
  2. Incorporar Atributos
  3. Determinar identificadores
  4. Identificar asociaciones
  5. Determinar cardinalidad
  6. Atributos a las asociaciones
  7. Verificar con los req

Superclase y Subclase



Entidad Fuerte: Existe por Clase

si ademas

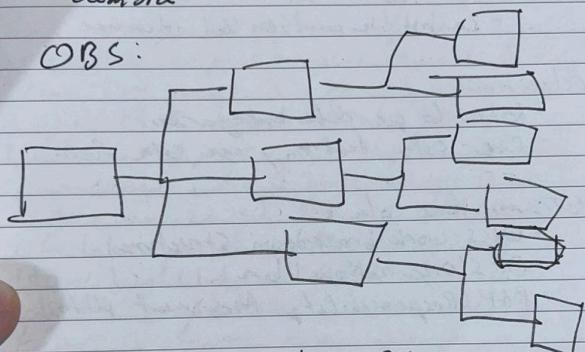
Entidad débil: depende de otra ent.

¿Como sabemos cuándo usar subclases en vez de solo atributos?

Buenas Prácticas:

- Descomponer entregables para permitir control sin ser excesivo
- Reglas claras
- Activarizar si el alcance cambia

OBS:



¿Como ayuda la WBS a prevenir el "Scope Creep" durante la ejecución del proyecto?

## DD vs DBMS

Antes los datos estaban atados a programas individuales → generaba

Como cada programa tiene sus propios archivos, se crean versiones distintas de un mismo dato

• redundancia

• errores

• contradicciones

Problemas del enfoque tradicional:

- Redundancia e inconsistencia
- Dificultad de acceso a los datos
- Acceso concurrente problemático
- Falta de seguridad e integridad
- Aislamiento de los datos

¿Qué es una DB?

Colección de archivos interrelacionados diseñados para reducir redundancia y representar integralmente a una organización

DBMS: Sistema de gestión de base de datos

DBMS:

- Facilita acceso a datos
- Supervisa: Integridad, seguridad, respaldo, recuperación y control de concurrencia

¿Qué impacto tuvo en empresas y trabajos el cambio a DB?

## Gestión del alcance

### Administración del alcance

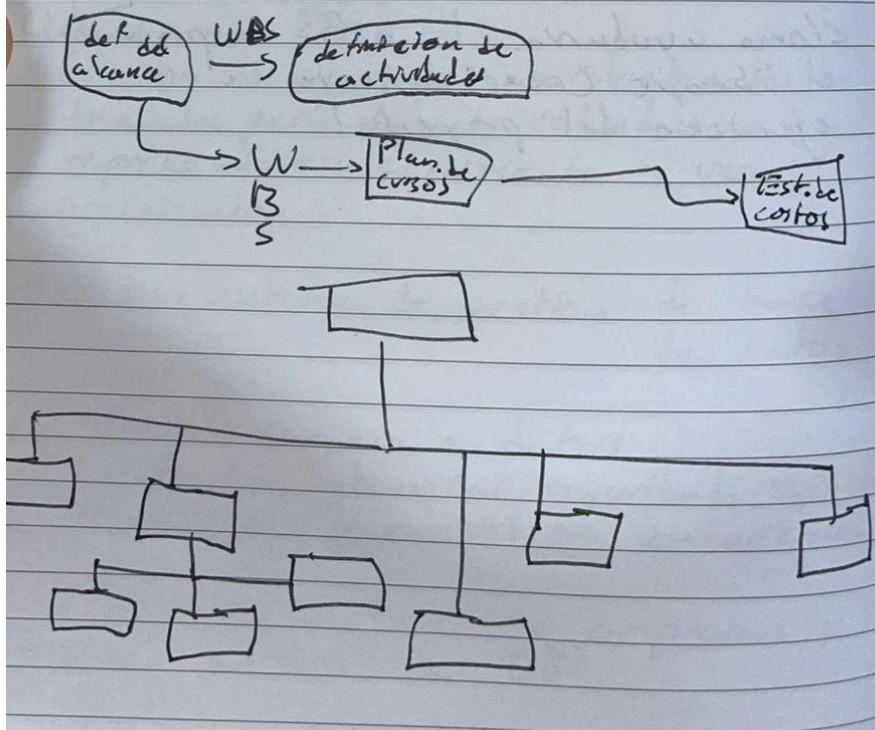
- Definición
- Planificación del alcance
- Definición del alcance
- Verificación del alcance
- Control de cambios del alcance

## Alcance

- Todo lo que debe entregarse
- Que está dentro y que está fuera

## Herramientas clave:

- WBS (Work breakdown Structure)
- OBS (Organizational Breakdown Structure)
- RAM (Responsibility Assignment Matrix)



## Sistema de Información

Conjunto de componentes interrelacionados que recopilan, procesan, almacenan y distribuyen información para la toma de decisiones.

### Elementos clave del sistema

- Entrada
- Procesamiento
- Salida
- Retro

### Importancia

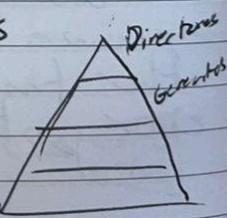
- tareas cotidianas
- gestión estratégica

Dato: representación simbólica

Información: organismo de datos

### Tipos de Sistemas

- TPS → Procesos operativos
- OAS → tareas de oficina
- KWS → Trabajo especializado
- MIS → informes y control
- DSS → Soporte en decisiones
- GDSS → toma de decisiones en grupo
- ES → Simular Razonamiento
- EIS → decisiones estratégicas



¿Cómo ha evolucionado el papel del director de informática?

## Gestión de proyectos Guía del PMBOK

- Entorno organizacional de los proyectos

9 áreas de conocimiento

- Integración alcance, tiempo, costo, calidad, recursos humanos, comunicación, riesgos y aprovisionamiento

- Describe el entorno

- Describe y organiza las características

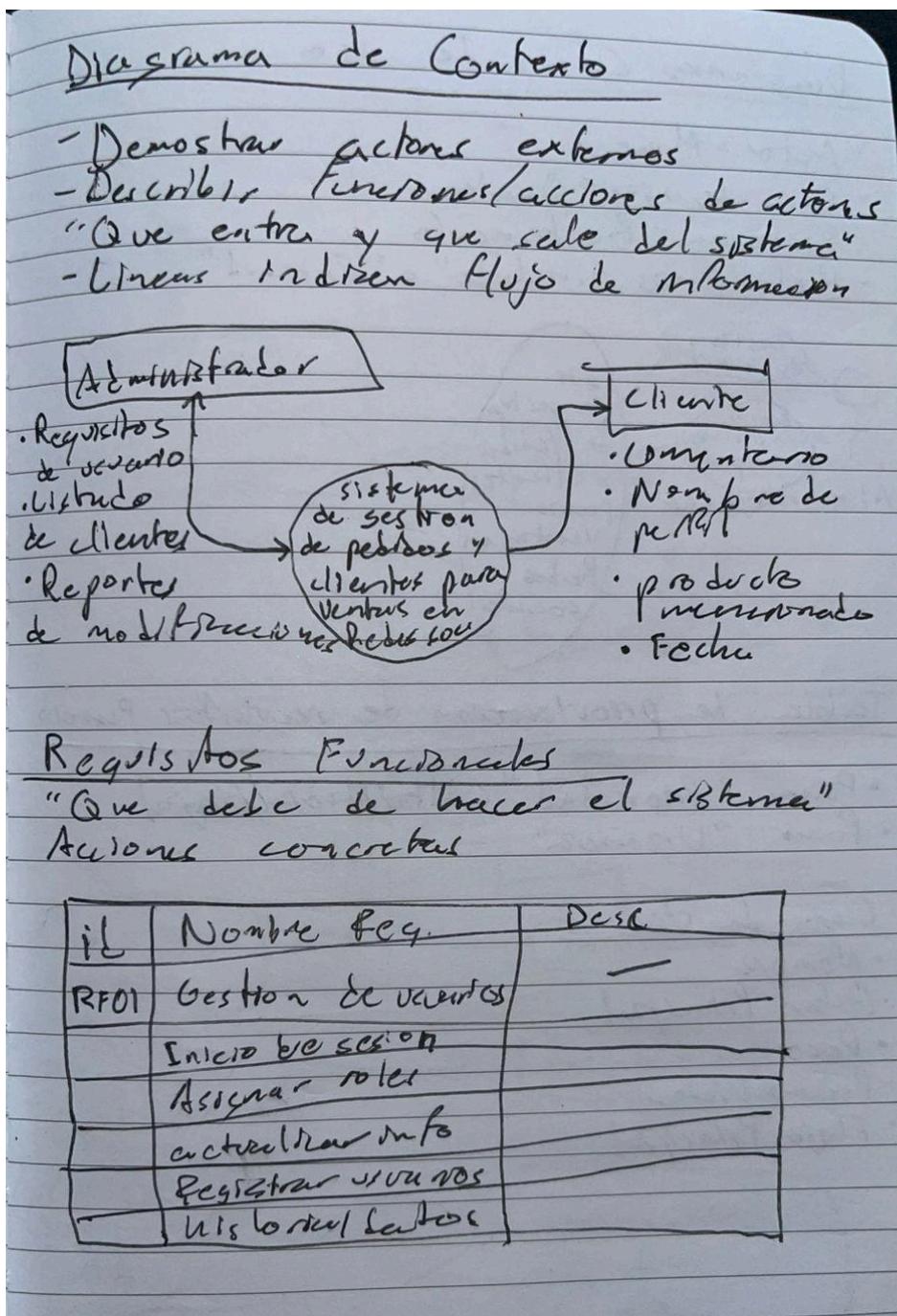
- Describe el conocimiento

Causas de fracaso

- Requerimientos cambiantes o incompletos
- Poca participación de usuarios
- Débil gestión del proyecto
- Falta de planificación, comunicación, soporte gerencial, conocimientos técnicos

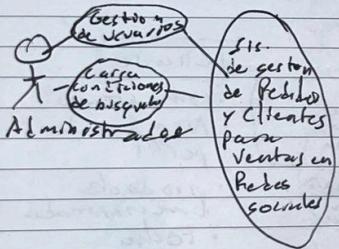
¿Cómo se ha desarrollado el PMBOK  
a través de los años?

Notas:



## Diagramas Casos de uso

- Actor = Muñeco
- Caso de uso = Ovalo
- Sistema = Rectángulo
- Relaciones "include" o "Extend"



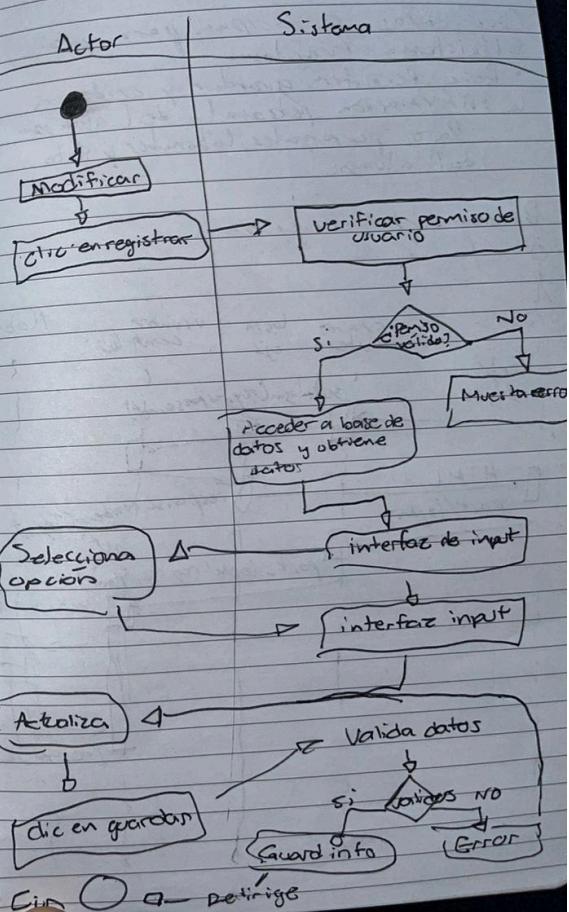
## Tabla de priorización de requisitos Puros

- Poner "Prioridad" (Alta/Mediana/Baja)
- Poner "Urgencia"

## Caso de Uso

- Nombre
- Actor Principal
- Desc.
- Pre condición
- Flujo Principal

Día

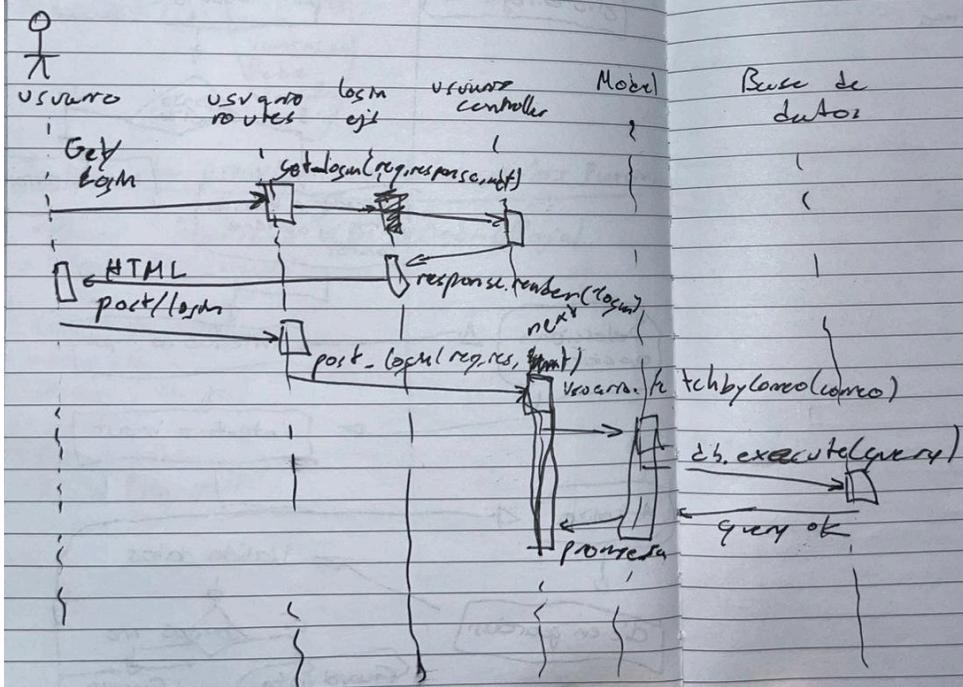


## Requisitos de Información

Que datos se ocupan para que el sistema funcione

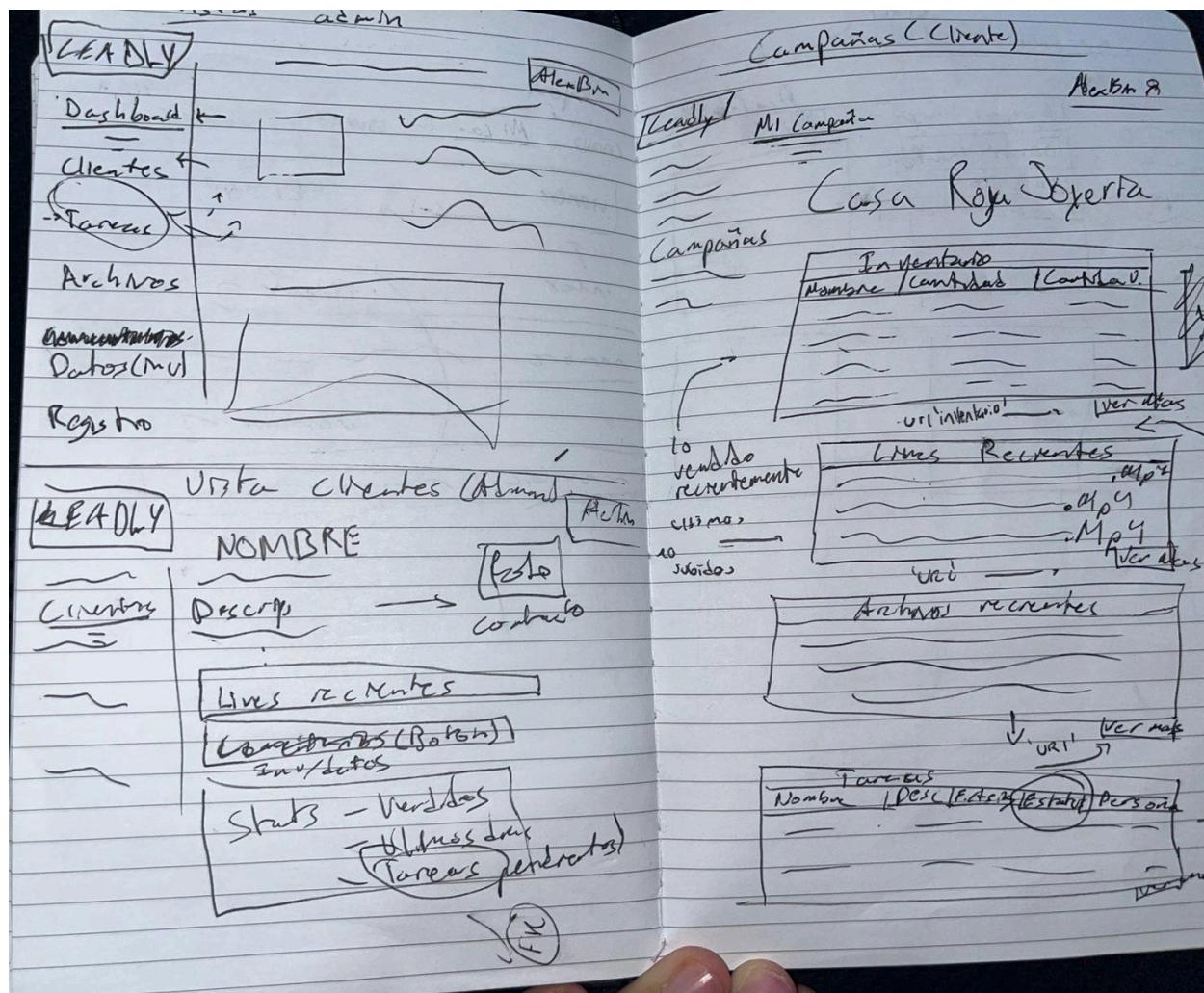
- Debe permitir guardar y acceder a la información personal del alumno
- Datos personales, labores y foto del alumno

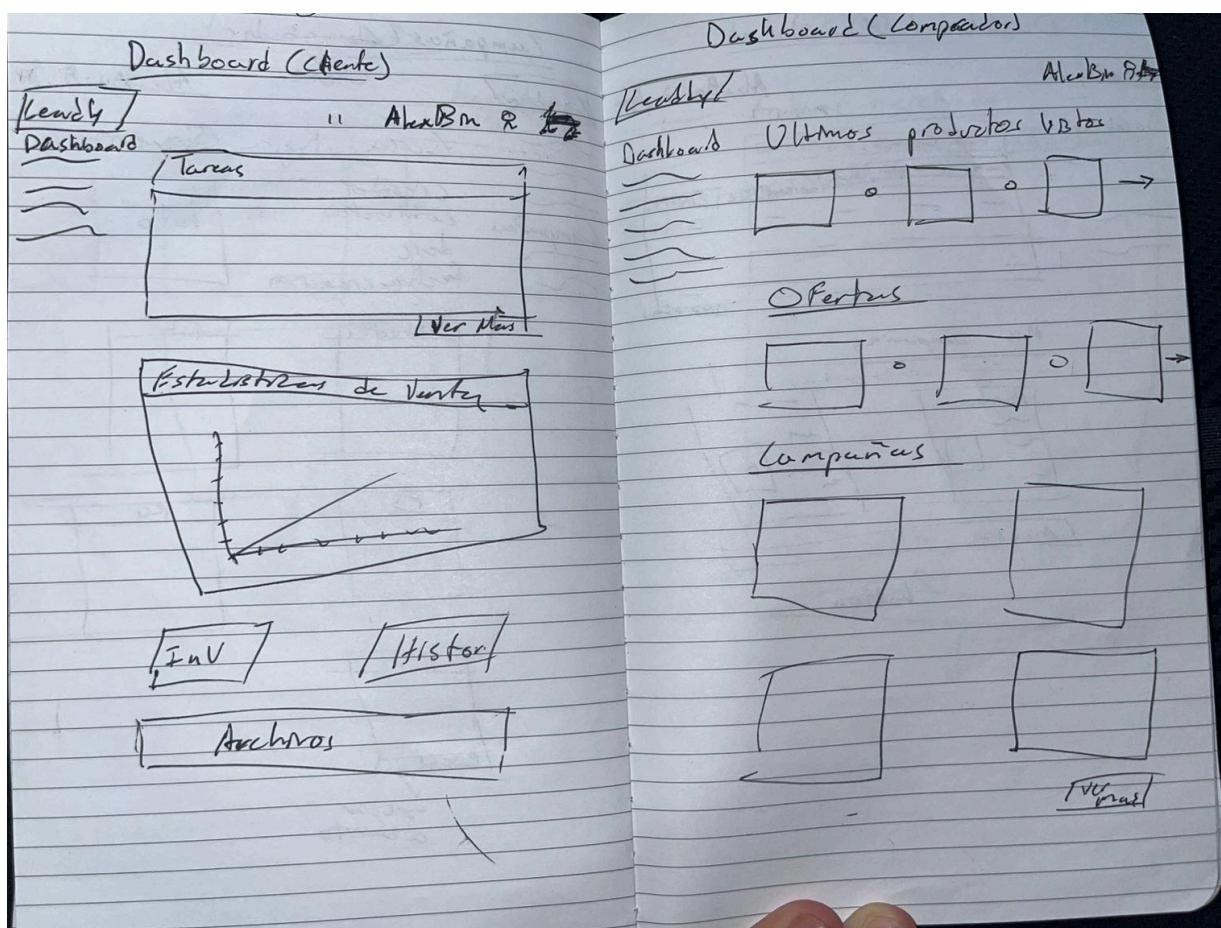
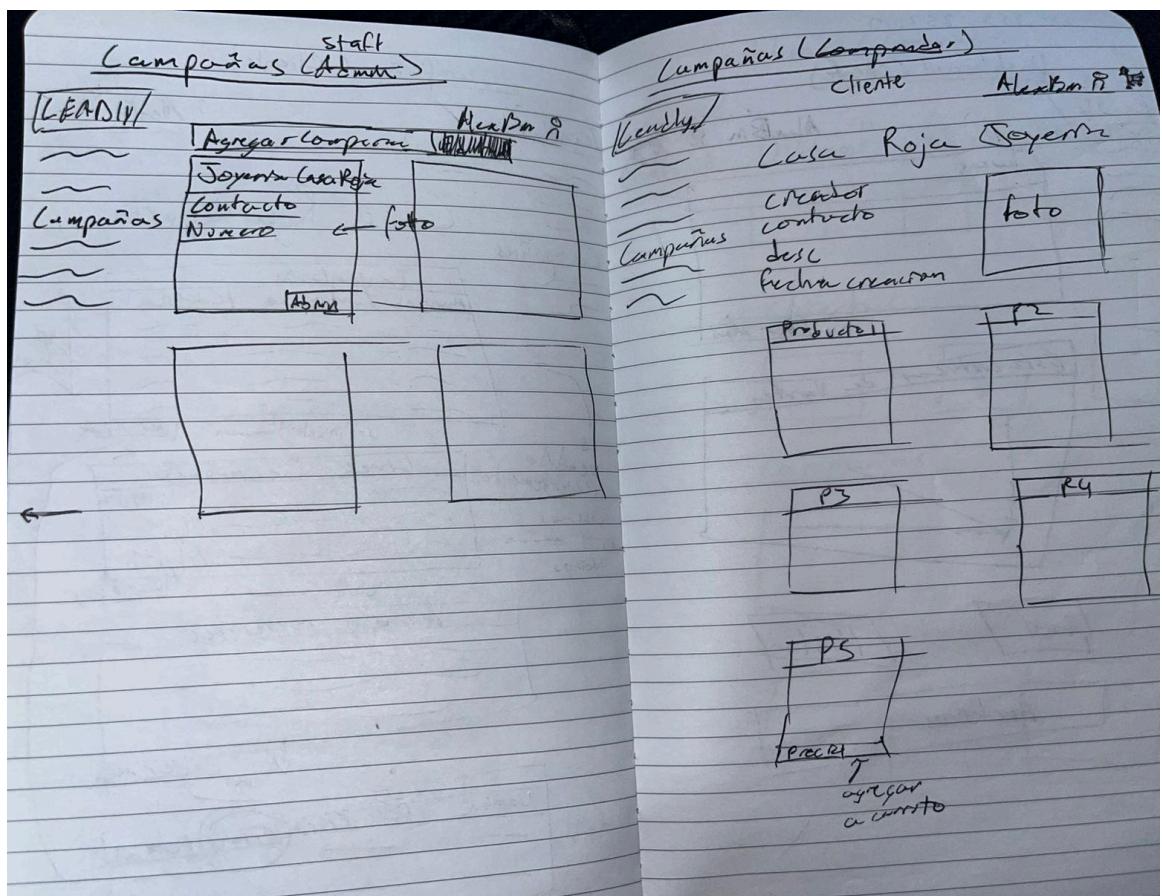
## Diagrama de secuencia

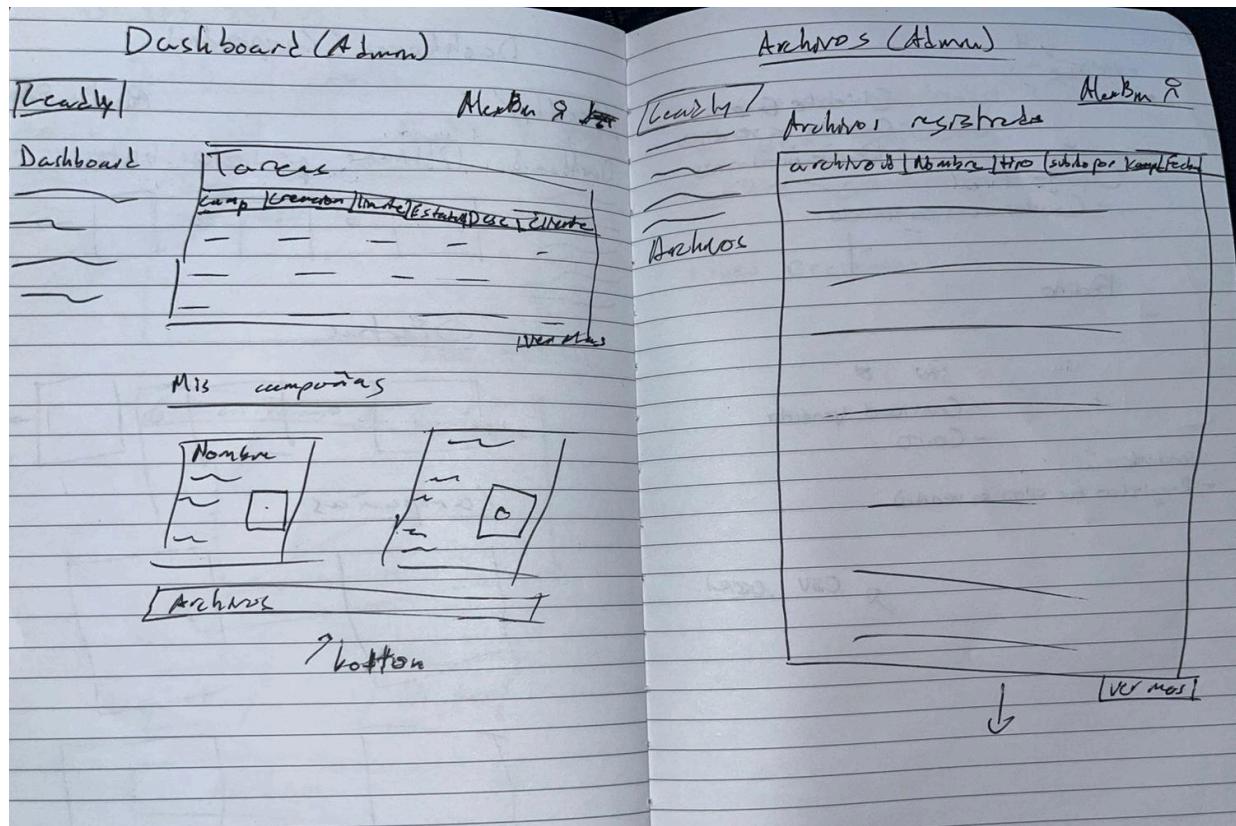


## Documentación de Restricciones

R1 → Un usuario no puede registrar más de 10 apuestas pendientes al mismo tiempo





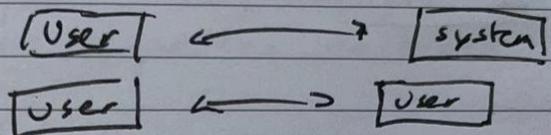


## Requerimientos de Software

- Descripción detallada de lo que hace el sistema (Funciones) y cómo debe comportarse (restricciones)

### Requerimientos Funcionales

- Que hace el sistema
- Interacciones:



ej.: "El sistema debe enviar notifica... autent."

### Requerimientos NO Funcionales

- Como debe comportarse el sistema
- Restricciones
- Calidades

ej. "Interfaz en ESP y en INGLES"

### Otros Tipos

- Requisitos del usuario: Deseos y necesidades del usuario
- Requisitos del sistema: Versión Vista para Windows
- Requisitos de interfaz: Protocolos, formularios, API's
- Requisitos de dominio: Reglas del negocio

## Características

- Correcto
- Claro
- Verificable
- Consistente
- Completo
- Rastreable
- Modelable

## Ciclo de vida

Elicitación

Análisis

Especificación

Validación

Gestión de cambios

## Diagrama de contexto

Ventura

Sistema  
de  
pedidos

## Diagrama de Casos de Uso

Alma

Gestión  
de cambios

Sistema  
de  
pedidos

## Requisitos Funcionales: Acciones

- Debe permitir crear una cuenta
- Tras registro manda correo de confirmación

## Requisitos NO Funcionales: como debe comportarse el sistema

- debe estar disponible 24/7
- El cliente puede recuperar su contraseña desde el app las respuestas tardan mas de 2 seg

## Requisitos del Usuario

- Supervisor puede crear un manual PDF
- El cliente puede recuperar su contraseña desde el app

## Requisitos de negocio: objetivos o metas que justifican el proyecto

→

- Debe mirar sesión con google algo

↓

## Requisitos de Información: Que se va guardar y como

Diagrama de actividad

## Desarrollo de Componentes de Software

Convención de GoDaddy:

Para el equipo usamos cosas ~~de~~ estandar

- Para variables " \_ "
- Clases con uppercase
- nombres simples y coherentes
- Usa PascalCase

Front End:

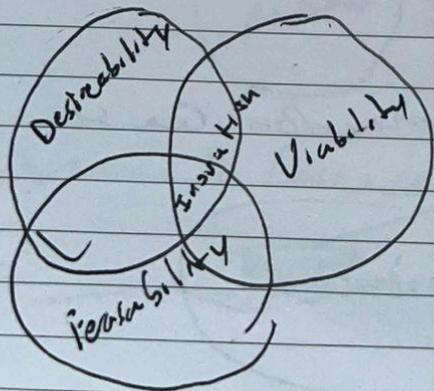
Se utilizan frameworks modernos de estilo junto con Mockups en Canva.

Metodologías

- SCRUM

Arquitectura:

- Cliente - Servidor
- División de responsabilidades por modelos



## Diseno de Componentes de Software

- Unidades funcionales
- responsabilidades claras, una interfaz  
y debe ser reutilizable
- ej. Modulo de pagos

↳ Se debe crear

### Requerimientos

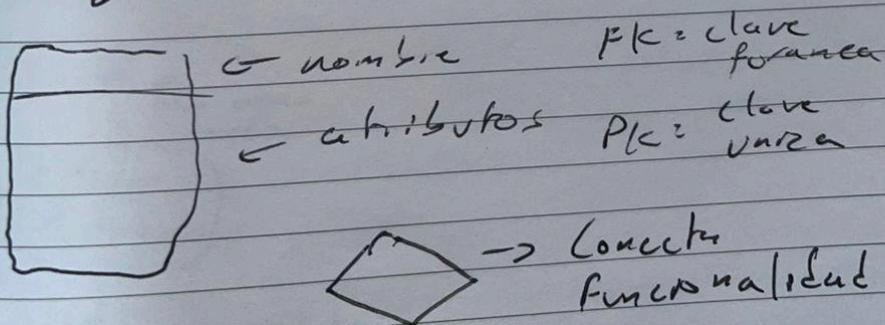
- Funcionales
- No Funcionales

### Arquitectura:

- Arquitectura MVC

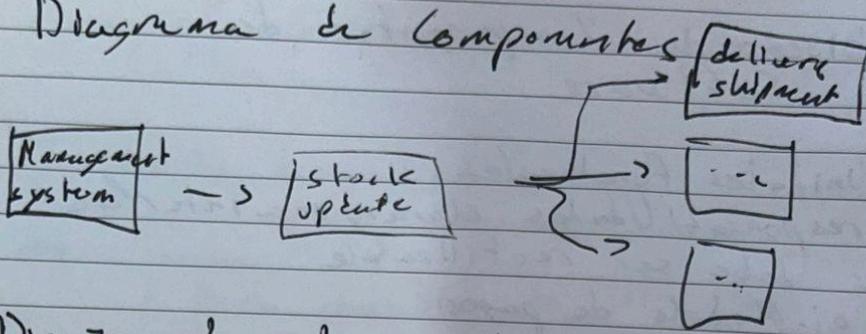
Cliente → Controlador → Modelos  
(HTML, CSS, JS)      (View)      (Base de datos)

### Diagramas: UML



N:1 }  
1:1 } etc (logica)  
1:N }

## Diagrama de Componentes



Diseñar basado en la base de datos

un buen diseño

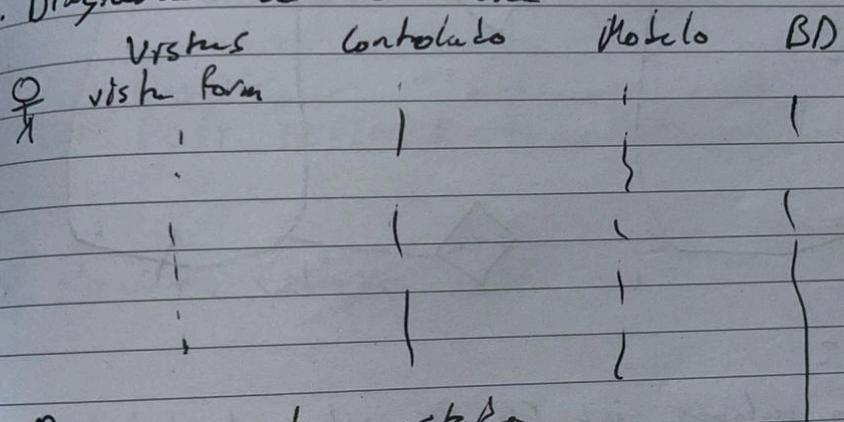
- Reduce errores en desarrollo
- Ahorra tiempo de prueba
- Facilita mantenimiento

1. Sistema Arquitectónico (MVC)
- Interfaz: HTML/CSS
  - Backend: node.js express
  - Base: MongoDB/SQL
  - Externos: API

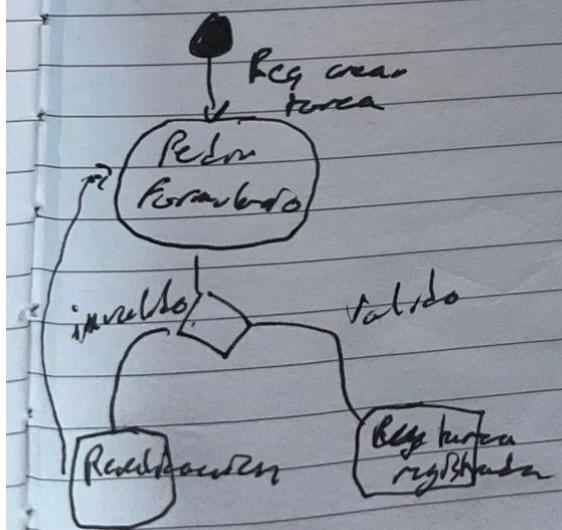
• Desplegar funcionalidad

• Desplegar actores

2. Diagrama de secuencia

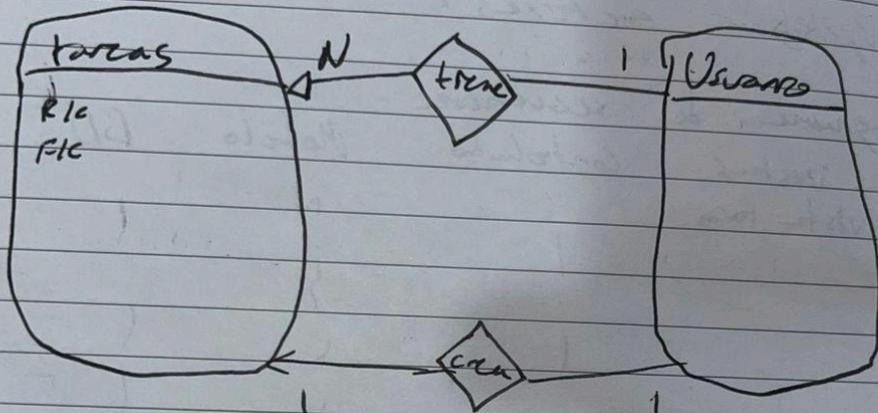


3. Diagrama de estados



## 4. Diagrama de Interacciones

### 5. MER



### 6. MR

\* Con valores int, F1c, P1c, vnl, yor, text, date, varchar

A ( $a_0, a_1$ )

- P1c: ( $a_0$ )

B ( $b_0^*, b_1, b_2, b_3^*$ )

- P1c: ( $b_0$ )

- P1c: ( $b_0$ )  $\rightarrow A$

- F1c: ( $b_3$ )  $\Rightarrow C$

7. Pseud  
Regist  
INIC  
: De  
: "

FIN

8. P

ay.

## 7. Pseudo Código

### Registrar Tarea

INICIO

» GET desc, título

- Desplegar vista de formulario
- "Llena todos los campos"

WHILE desc, título → INVALIDOS  
GET desc, título

IF descripción, título → validos

- Cambiar a vista de tarea creada
- Desplegar "tarea creada"

FIN POST DEFINIR tarea (desc, título)

## 8. Pseudo Código SQL

9. Creación de: nombres, estilos, carpetas  
etc.

## STCD205 - Pruebas de Software

Verificar que el sistema hace lo que tiene que hacer

Tipos de Pruebas:

- Caja Blanca → código por dentro
- Caja Negra → entrada y salida
- Unitaria → Funciones individuales
- Manual → Usuario prueba y observa
- Estática → Revisión de código

Caso de prueba:

Nombre: ~

Requisitos Relacionados: ~ ~

Pasos:

1. ~

2. ~

3. ~

4. ~

5. ~

Resultado Esperado: ~

~~~~~

~~~~~

~~~~~

Resultado Real: ~

~~~~~

~~~~~

Variaciones: ~

~~~~~

~~~~~

Evaluación Heurística  
• Revisar si el sistema es usable

- Principios de Nielsen:
  1. Estado Visible del sistema
  2. Control y libertad del usuario
  3. Coherencia
  4. Prevención de Errores
  5. Reconocer, no recordar
  6. Eficiencia
  7. Estética y diseño minimalista
  8. Ayuda al Usuario
  9. Diagnosticos de errores
  10. Recuperación ante fallos

Pensamiento en Voz alta:

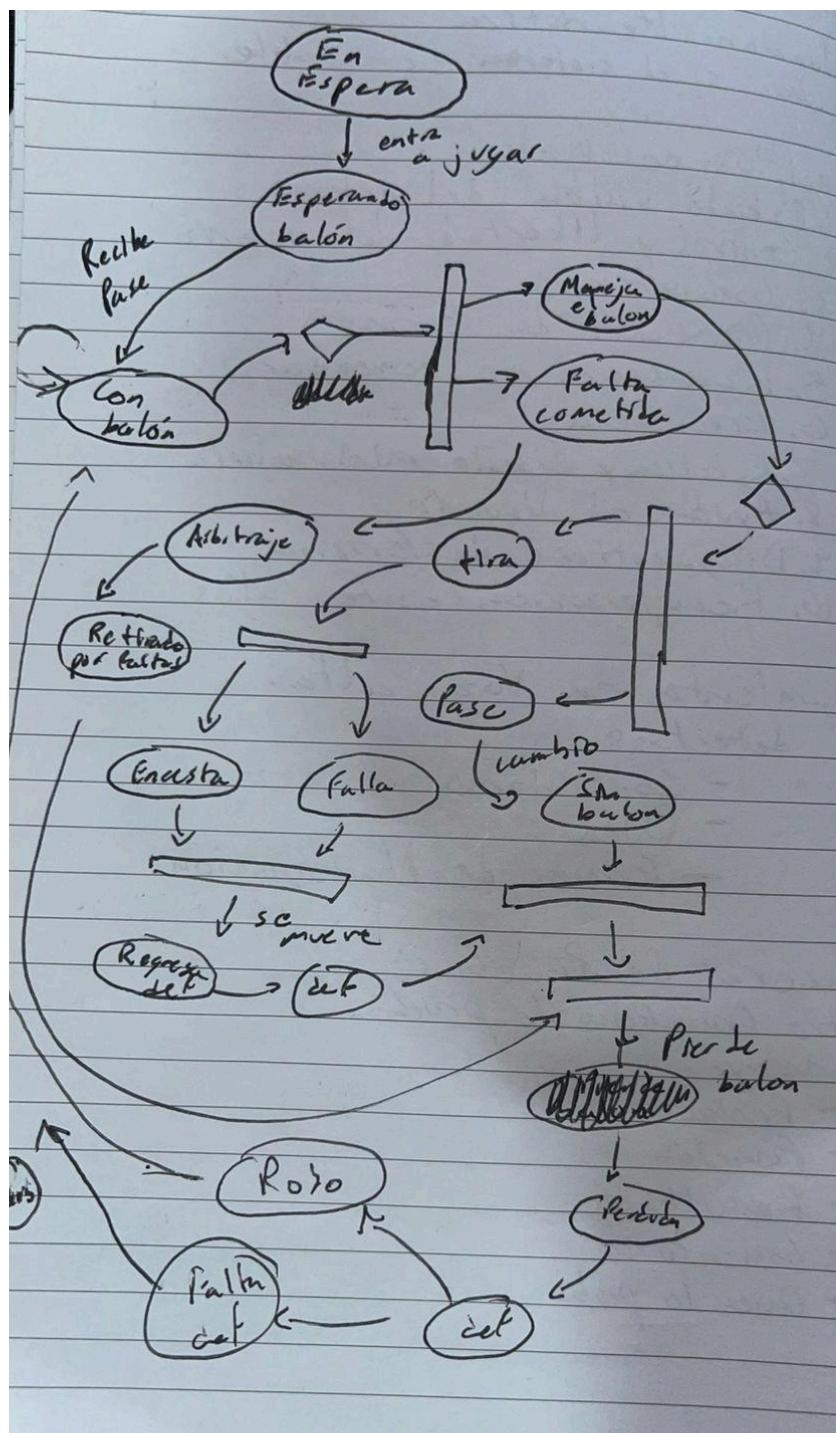
- Se detectan:
  - Confusiones
  - Dudas
  - Errores de Navegación

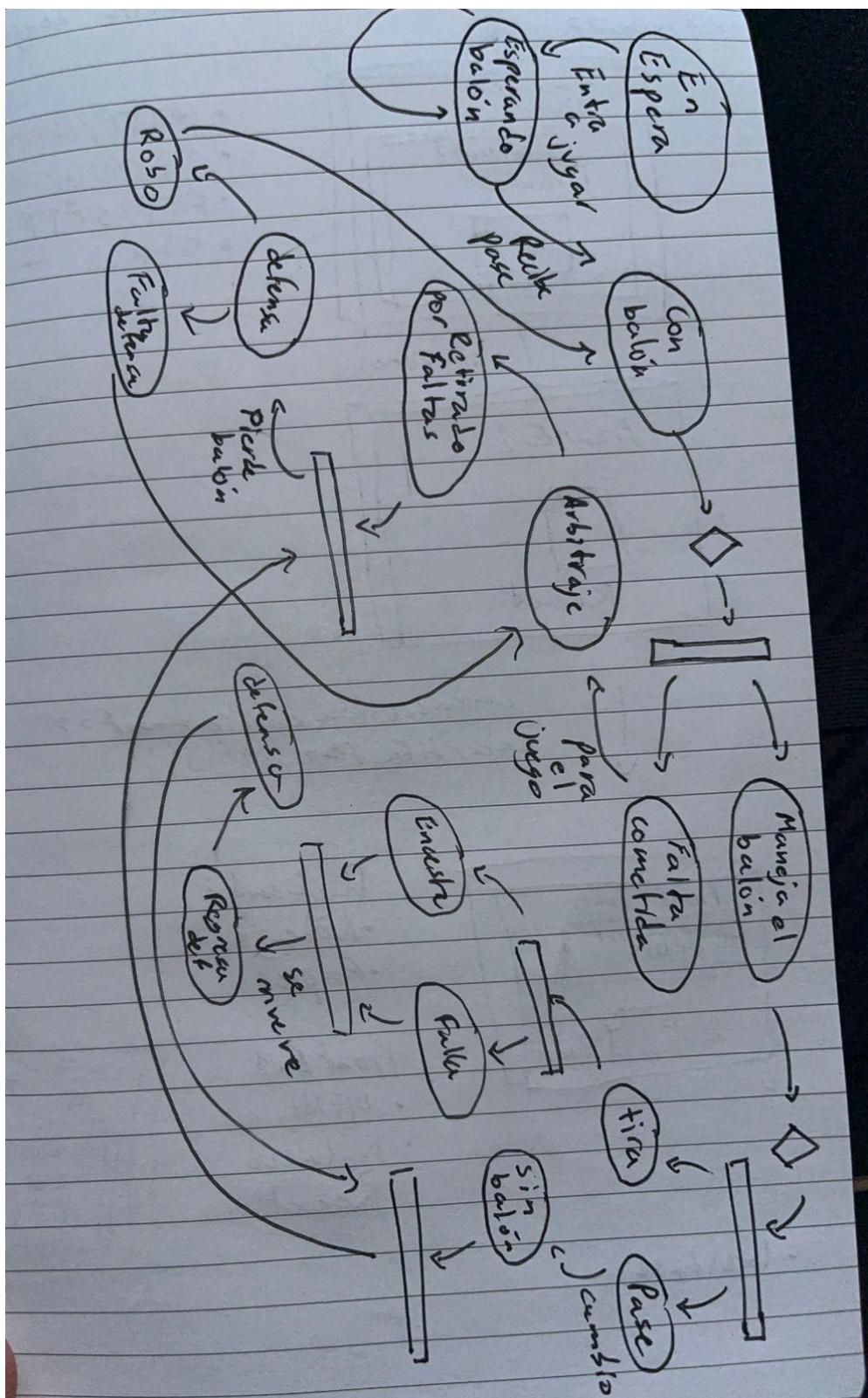
Bitácora de Pruebas:

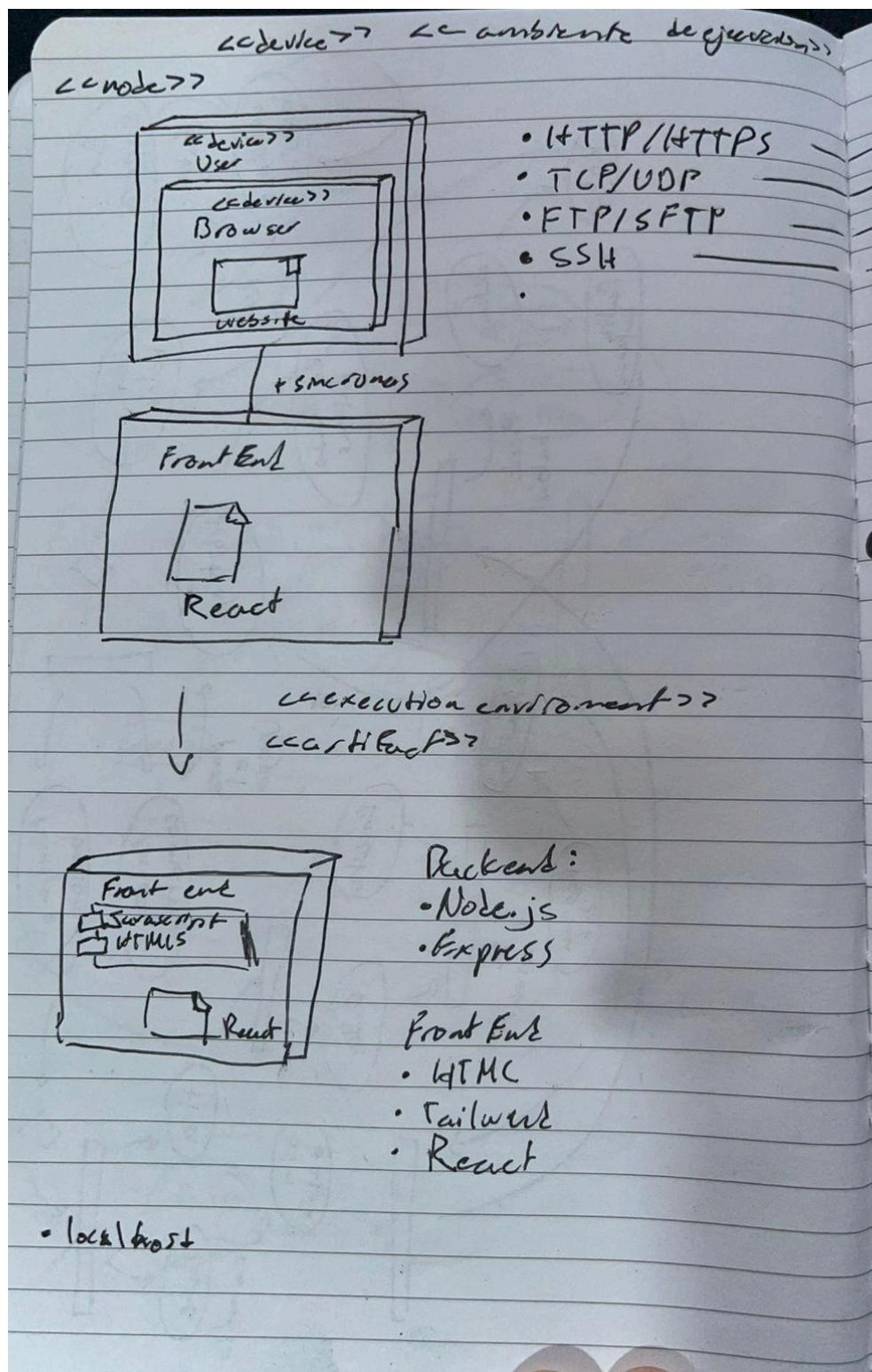
Registro cronológico de pruebas

incluye:

- Fecha
- Función
- Resultado
- Comentario
- Quién lo probó

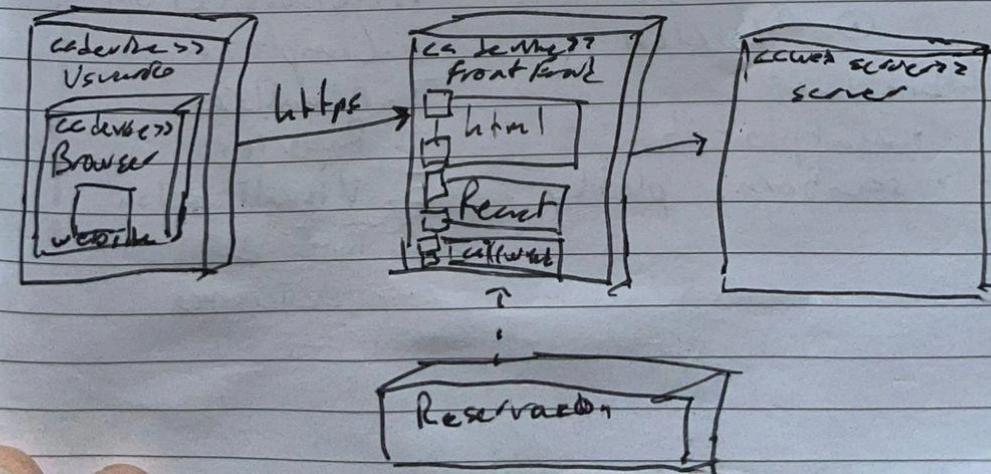
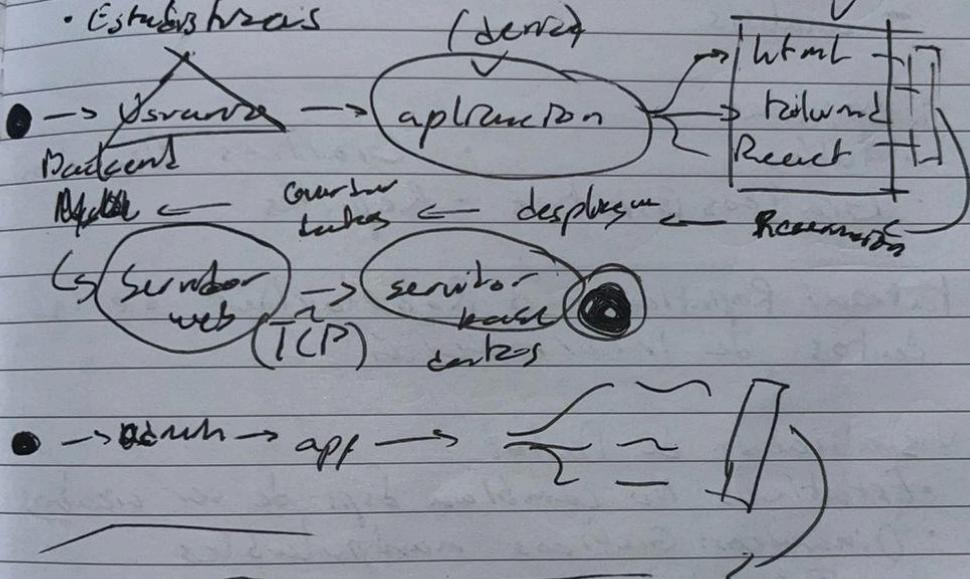






- ej navegador → servidor web
- cliente base de datos → servidor MySQL (TCP)
- Transferencia de archivos
- Para administrar servidores

- Reservaciones en linea
- administrar reservaciones/meses (http)
- Estadísticas (dadas)



## Determinación de Patrones

Dato: Elemento de información

Información: Datos procesados  
~~~~~  
~~~~~  
~~~~~

La transformación de datos  
requiere procesos como:

- La limpieza
- Organización
- análisis

### Herramientas

- Tablas
- Graficos Dinámicos
- Graficos Estáticos
- Reportes

Patrón: Repetición o Regularidad de los  
datos de información

### Visualización de Datos

- Estáticos: No cambian después de ser creados
- Dinámicos: Graficos manipulables

① Claridad

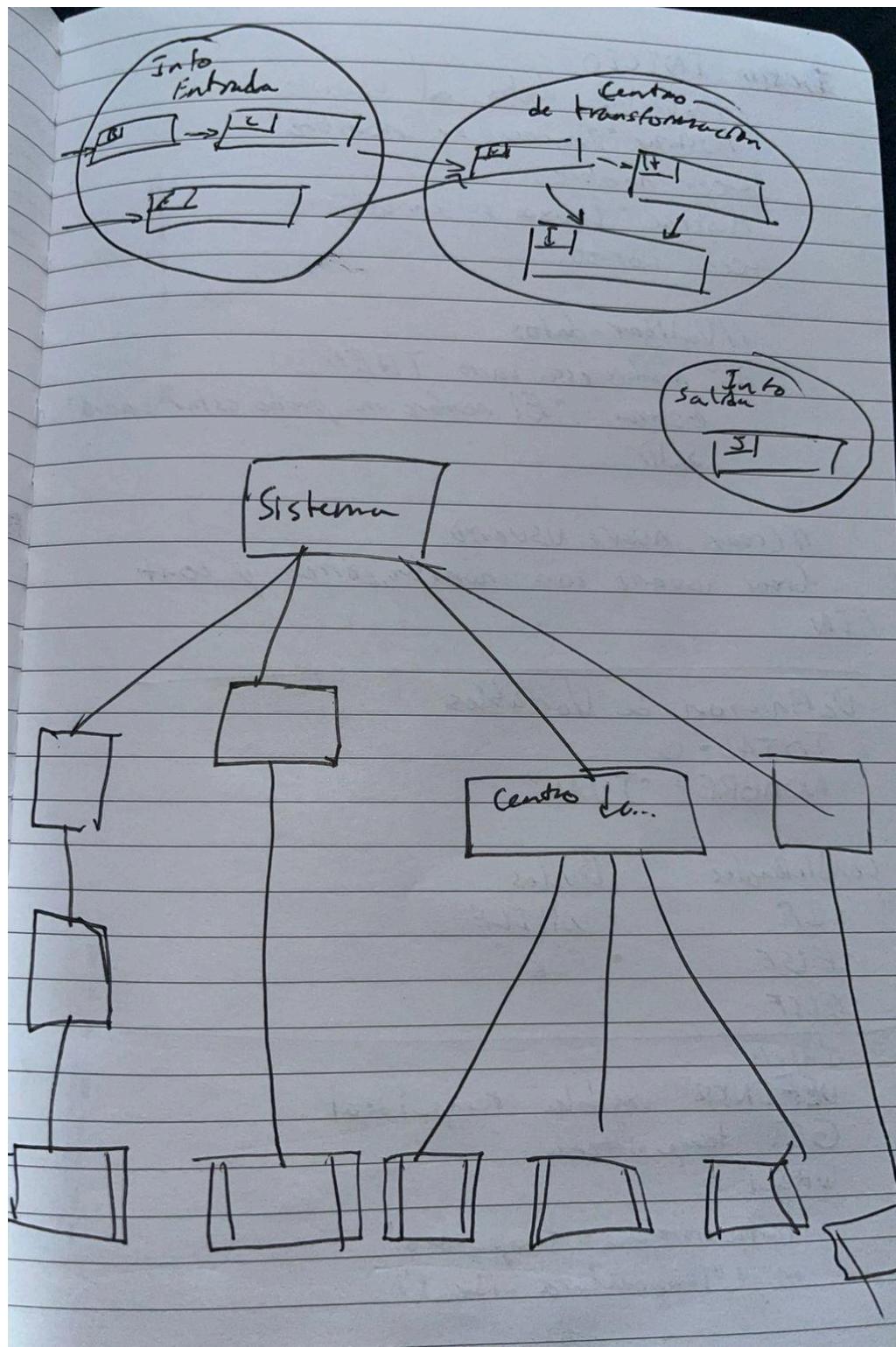
② Rapidez

③ Precisión

### Librerías:

- matplotlib
- seaborn, plotly

1. Recolección
2. Limpieza
3. Análisis
4. Reportes
5. Visualización
6. Toma de decisiones



## ~~INICIO~~ INICIO

//Solicitar datos al usuario

Mostrar "Ingrese su nombre"

Ler nombre

Mostrar "Ingrese su correo"

Ler correo

//Validar datos

IF nombre está vacío THEN

Mostrar "El nombre no puede estar vacío"

Salir

//Crear nuevo usuario

Crear usuario con nombre,correo y cont

FIN

## Definición de Variables

TOTAL = 0

NOMBRE = "JUAN"

## Condicionales

- IF

- ELSE

- ECIF

## Bucles

- WHILE

- FOR

## Inicio

DEFINIR variable temp-ideal

GET temp-sistema

WHILE

temp-sistema < temp-ideal

msg("Temperatura ideal")

Tareas sql

Select: SELECCIONAR # DESDE clientes  
SELECCIONAR nombre, edad DESDE clientes  
where: WHERE edad > 18  
Insert: INSERTAR EN clientes (nombre, edad, com)  
VALORES ('JUAN', '25', 'joven')

SELECT UPDATE CREATE TABLE  
INSERT DELETE ORDER BY