Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Curso Inteligencia Artificial aplicada a la Ingeniería de Software

Curso 2025-26

Proyecto Final

Título

ENTREGA: **1 - FINAL**

EQUIPO: **XXX**

ALUMNOS: **Nombre apellido / Nombre Apellido /Nombre apellido / Nombre apellido**

FECHA: **xx/xx/xx** –

PÁGINA DE CONTROL DE DOCUMENTO

Registro de Cambios

|  |  |  |  |
| --- | --- | --- | --- |
| Versión | Fecha | Autor | Cambios en la versión |
| 0 | 04/08/2025 | Ana García | Plantilla inicial |
| 1 |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Horas invertidas en el proyecto

|  |  |  |  |
| --- | --- | --- | --- |
| NOMBRE | INDIVIDUAL | EQUIPO | TOTAL |
| XX |  |  |  |
| XX |  |  |  |
| Xx |  |  |  |
| XX |  |  |  |
| TOTAL |  |  |  |

Contribución a la redacción del documento

|  |  |
| --- | --- |
| NOMBRE | Descripción de sus principales contribuciones a la redacción del documento |
| XX |  |
| XX |  |
| XX |  |
| XX |  |

TABLA DE CONTENIDO

1 INTRODUCCIÓN Y OBJETIVOS GENERALES 5

1.1 Objetivos generales del proyecto 5

1.2 Visión general del documento 5

2 DESCRIPCIÓN DEL PRODUCTO 6

2.1 Descripción del producto a desarrollar 6

2.2 Situación actual y estado del arte 6

2.3 Alcance y Funcionalidades generales 6

2.4 Restricciones y elementos fuera del alcance 6

2.5 Diagrama de Contexto 6

3 REQUISTOS 9

3.1 Clasificación de los requisitos 9

3.2 Plantilla para la redacción de requisitos 9

3.3 Requisitos funcionales 10

3.4 Requisitos no funcionales 10

3.5 Vocabulario del dominio 10

4 ALGORITMO GENÉTICO 11

4.1 Población / individuo / cromosoma / gen 11

4.2 Creación de población 11

4.3 Algoritmo de selección 11

4.4 Algoritmos de cruzamiento 11

4.5 Algoritmo de mutación 11

4.6 Algoritmo de reemplazo 11

5 MODELADO CONCEPTUAL 12

5.1 Diagrama de Casos de Uso 12

5.2 Diagramas de Clases 13

6 ARQUITECTURA 15

6.1 Modelo de implementación 15

6.2 Diagrama de componentes 16

7 LISTAS Y TABLAS 18

7.1 Consistencia entre requisitos: conflictos, redundancias, acoplamientos 18

7.2 Trazabilidad requisitos – modelo conceptual (clases) 18

7.3 Trazabilidad requisitos – modelo de implementación (componentes) 18

8 ANEXOS 19

8.1 Anexo I – Bibliografía 19

8.2 Anexo II – Uso de IA en el proyecto 19

CONSIDERACIONES SOBRE LAS ENTREGAS

* + La **primera entrega** incluye los capítulos 1, 2, 3 y 4
  + La **entrega final** incluye el resto de capítulos y revisión de capítulos previos

Eliminar esta página del trabajo final

# INTRODUCCIÓN Y OBJETIVOS GENERALES

Introducción al documento y al proyecto

## Objetivos generales del proyecto

## Visión general del documento

Esta sección debe:

1. Describir lo que contiene el resto del documento.
2. Explicar cómo está organizado el documento.

# DESCRIPCIÓN DEL PRODUCTO

## Descripción del producto a desarrollar

Descripción global del producto a desarrollar

* Problemas que aborda
* Objetivos
* Motivación
* Beneficios esperados

Descripción de alto nivel

## Situación actual y estado del arte

Investigación inicial sobre trabajos (académicos, profesionales…) ya realizados alrededor del problema que se aborda con el proyecto.

Productos existentes (comerciales, etc.,) análisis, ventajas, inconvenientes.

## Alcance y Funcionalidades generales

Alcance en términos de grandes funcionalidades / objetivos del producto a desarrollar

## Restricciones y elementos fuera del alcance

## Diagrama de Contexto

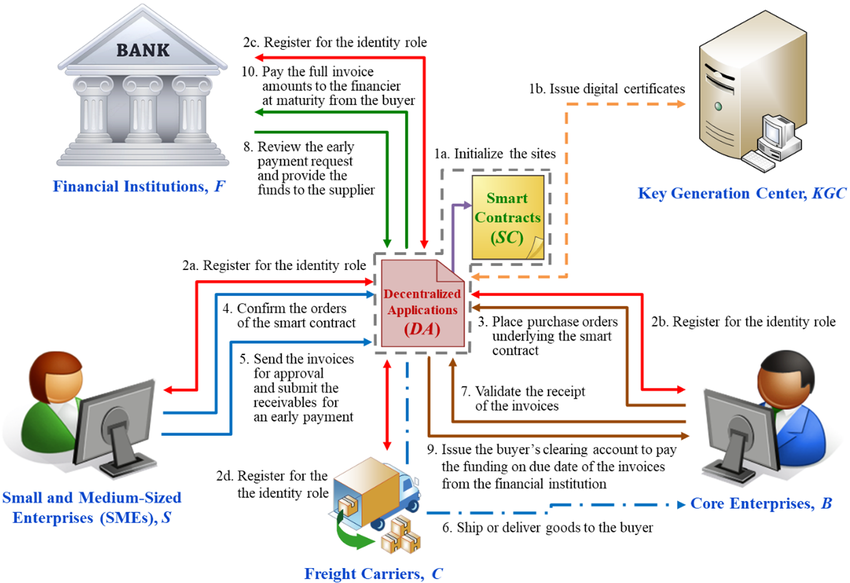
Diagrama posicionando al producto en su contexto de uso, interacciones con usuarios y otros stakeholders, otros productos, instituciones, etc.

Ejemplos:

Diagrama

El contenido generado por IA puede ser incorrecto.

Ejemplo de diagrama de contexto., fuente: <https://grupo13sistemaacademicoescolar.wordpress.com/wp-content/uploads/2012/04/diagrama-de-contexto1.jpg>



Ejemplo de diagrama de contexto., fuente: <https://www.researchgate.net/figure/Operational-context-diagram-of-the-proposed-blockchain-driven-model_fig4_365156154>

# REQUISTOS

Este capítulo debe contener la especificación completa de los requisitos del sistema software que se va a desarrollar

Los requisitos deben expresarse de forma clara, precisa, completa, correcta, consistente y verificable (ver sesión en clase dedicada a la calidad de los requisitos).

Este capítulo también incluye el vocabulario del dominio, de modo que en los requisitos se utilice en todo momento un vocabulario controlado. Los requisitos y vocabulario del dominio se elaboran en paralelo.

## Clasificación de los requisitos

Hacer una clasificación práctica (por bloques funcionales, por grandes apartados del producto a desarrollar…) de los requisitos.

* + Requisitos Funcionales
    - Control de acceso
    - Consultas e Informes
    - Parametrización Algoritmo Genético
    - …
  + Requisitos No Funcionales
    - Seguridad
    - …

## Plantilla para la redacción de requisitos

Descripción de la plantilla utilizada para la elaboración de requisitos. Usar una plantilla básica, con pocos atributos.

Ejemplo:

|  |  |  |  |
| --- | --- | --- | --- |
| ID |  | Requisito | **TITULO /DESCRIPCIÓN CORTA** |
| Descripción | Descripción larga del requisito | | |
| Pruebas de verificación | Pruebas a realizar para verificar / validar el requisito | | |
| Dependencias | Lista de otros requisitos dependientes o de los que depende | | |

## Requisitos funcionales

Esta sección debe contener los requisitos funcionales. Los requisitos deben ser redactados en lenguaje natural. Esto los hace más comprensibles para los no especialistas, aunque abre la puerta a que surjan inconsistencias, ambigüedades e imprecisiones. Estas propiedades no deseables pueden ser evitadas con el uso de un lenguaje técnico adecuado para la especificación de requisitos.

## Requisitos no funcionales

Esta sección debe contener los requisitos no funcionales, igualmente redactados en lenguaje natural. Pueden afectar a todo el conjunto de requisitos funcionales, o bien sólo a una parte de los mismos, en cuyo caso debe especificarse de qué parte se trata (por ejemplo, requisitos de seguridad relacionados con determinado grupo de funcionalidades del software).

## Vocabulario del dominio

El vocabulario del dominio (también denominado a veces “glosario de términos”) se subdivide en dos apartados, que es muy importante distinguir bien.

### Vocabulario del modelo conceptual

Define todos los términos significativos y específicos del problema que aparecen en los requisitos. Los términos pueden ser simples o compuestos, nominales o verbales (ejemplos: “cuenta”, “estación de control”, “facturar”, “reasignar zona”).

Su elaboración comienza con la captura de requisitos, y evoluciona con el análisis de requisitos y el modelado conceptual.

Es una herramienta muy útil para elaborar los requisitos y el modelo conceptual, ya que éste debe recoger el vocabulario relevante utilizado en los requisitos.

|  |  |
| --- | --- |
| TÉRMINO | DESCRIPCIÓN |
| Término 1 |  |
| Término 2 |  |

Tabla de vocabulario del dominio. Modelo Conceptual

### Vocabulario técnico

Incluye otros términos (típicamente referidos al diseño o a las tecnologías empleadas en el sistema) que se ha juzgado conveniente definir, relativos a artefactos de implementación que no tienen relación directa con el modelo conceptual, aunque aparezcan en los requisitos.

No es necesario definir todos los términos técnicos que ya se conocen por el contexto de la asignatura (por ejemplo, ‘interfaz’), sino sólo aquellos que son específicos del proyecto (por ejemplo, un sistema especial de transmisión de información). También se deben definir aquí los acrónimos y abreviaturas empleados en el documento.

|  |  |
| --- | --- |
| TÉRMINO | DESCRIPCIÓN |
| Término 1 |  |
| Término 2 |  |

Tabla de vocabulario del dominio. Vocabulario técnico

# ALGORITMO GENÉTICO

Definición de los principales artefactos del algoritmo genético a utilizar. Modelado de poblaciones e individuos, algoritmos principales (fitness, cruzamiento, mutación, reemplazo…).

Se puede programar con más de una función y parametrizar, o cambiar de función según circunstancias. En este caso se debe explicar aquí, en cada apartado del documento.

Utilizar diagramas visuales que faciliten el entendimiento del AG.

## Población / individuo / cromosoma / gen

Para comprender la codificación y las estructuras del algoritmo genético aplicado al juego *Lights Out*, primero es necesario entender sus reglas básicas. El juego se desarrolla en un **tablero de 5x5**, representado mediante una matriz. Cada posición de la matriz contiene una **luz**, cuyo estado se modela como un valor binario:

* **1** corresponde a una luz encendida.
* **0** corresponde a una luz apagada.

Calendario

El contenido generado por IA puede ser incorrecto.

Cuando el jugador hace *click* sobre una casilla del tablero, se produce un cambio de estado en dicha casilla y en sus vecinas adyacentes (arriba, abajo, izquierda y derecha). Por ejemplo, en la figura XX se muestra el caso en que el jugador presiona la casilla (2,2). A esta acción la llamaremos simplemente **click**.

Imagen que contiene Calendario

El contenido generado por IA puede ser incorrecto.

El objetivo del juego es **apagar todas las luces**, es decir, lograr que todos los valores de la matriz sean 0.

Para modelar este juego mediante un algoritmo genético, definimos los siguientes elementos:

**Gen**: Representa un *click* en una posición específica del tablero. Cada gen corresponde a unas coordenadas (x,y) del tablero (donde x,y∈{0,1,2,3,4}).

**Texto

El contenido generado por IA puede ser incorrecto.**

**Cromosoma**: Se define como una **secuencia de genes**. En nuestro caso, un cromosoma está compuesto por 5 genes, es decir, una lista de 5 movimientos dentro del tablero.

Texto

El contenido generado por IA puede ser incorrecto.

**Individuo**: Es la concatenación de varios cromosomas. Cada individuo representa una **partida completa del juego**, conformada por 10 cromosomas (en total 50 genes).

Texto

El contenido generado por IA puede ser incorrecto.

**Población**: Conjunto de individuos que participan en el proceso evolutivo del algoritmo genético. En este trabajo, la población está compuesta por **500 individuos**.

Imagen que contiene Diagrama

El contenido generado por IA puede ser incorrecto.

**Codificación**:

La **codificación** en nuestro algoritmo genético se implementa de la siguiente manera:

**Gen**: Se representa como un par de coordenadas (x,y) que indican la posición del tablero en la que se realiza un click. En código, esto se puede modelar como una tupla (x, y).

**Cromosoma**: Es una lista de 5 genes, es decir, una secuencia de 5 clicks en distintas posiciones del tablero. En código corresponde a una lista de tuplas:  
  
 cromosoma = [(x1, y1), (x2, y2), ..., (x5, y5)]

**Individuo**: Está compuesto por 10 cromosomas, lo que equivale a 50 genes (50 movimientos). En código se puede representar como una lista de listas de tuplas:  
  
 individuo = [

    [(x1, y1), (x2, y2), ..., (x5, y5)],   # Cromosoma 1

    ...

    [(x46, y46), ..., (x50, y50)]          # Cromosoma 10

]

**Población**: Es un conjunto de 500 individuos. En código corresponde a una lista de individuos:  
  
 poblacion = [individuo1, individuo2, ..., individuo500]

Con esta codificación, el algoritmo puede manipular genes, cruzar cromosomas entre individuos y aplicar mutaciones en posiciones aleatorias del tablero para buscar una solución que apague todas las luces.

## Ejemplo de pseudocódigo (python)

## import random

# Definir un gen como un click en coordenadas (x, y) dentro del tablero 5x5

def crear\_gen():

    x = random.randint(0, 4)  # fila

    y = random.randint(0, 4)  # columna

    return (x, y)

# Un cromosoma es una lista de 5 genes

def crear\_cromosoma():

    return [crear\_gen() for \_ in range(5)]

# Un individuo es una lista de 10 cromosomas (total 50 genes)

def crear\_individuo():

    return [crear\_cromosoma() for \_ in range(10)]

# La población está compuesta por 500 individuos

def crear\_poblacion(tamano=500):

    return [crear\_individuo() for \_ in range(tamano)]

# Ejemplo de generación de población inicial

poblacion = crear\_poblacion()

print("Ejemplo de un individuo:", poblacion[0])

En este ejemplo, la función crear\_poblacion() genera automáticamente una población de 500 individuos, cada uno con 50 movimientos representados como coordenadas dentro del tablero.

## Creación de población

## Algoritmo de selección – Función fitness

## Algoritmo de creación de pool

## Algoritmos de cruzamiento

## Algoritmo de mutación

## Algoritmo de reemplazo

# MODELADO CONCEPTUAL

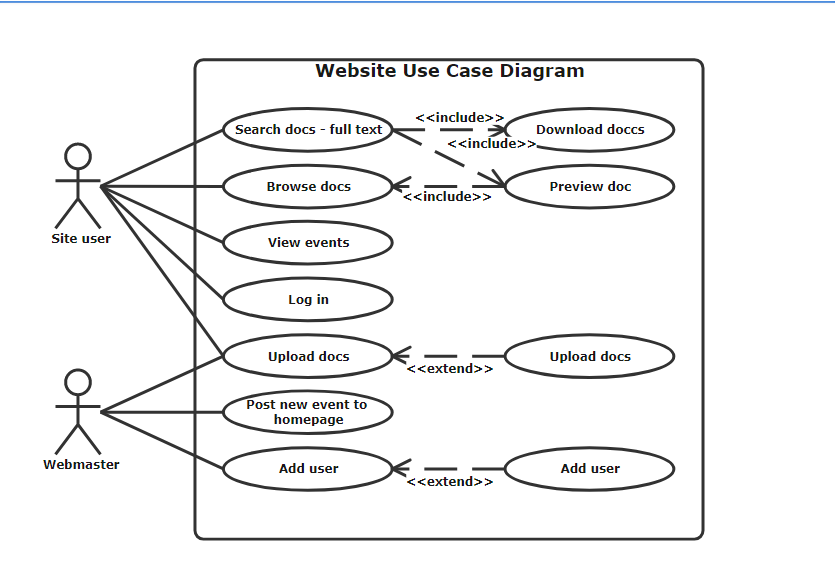
Esta sección debe contener un modelo cuyo propósito es especificar la *modelo de información* del sistema que se desea construir, mediante un conjunto de diagramas de clases adecuadamente explicados

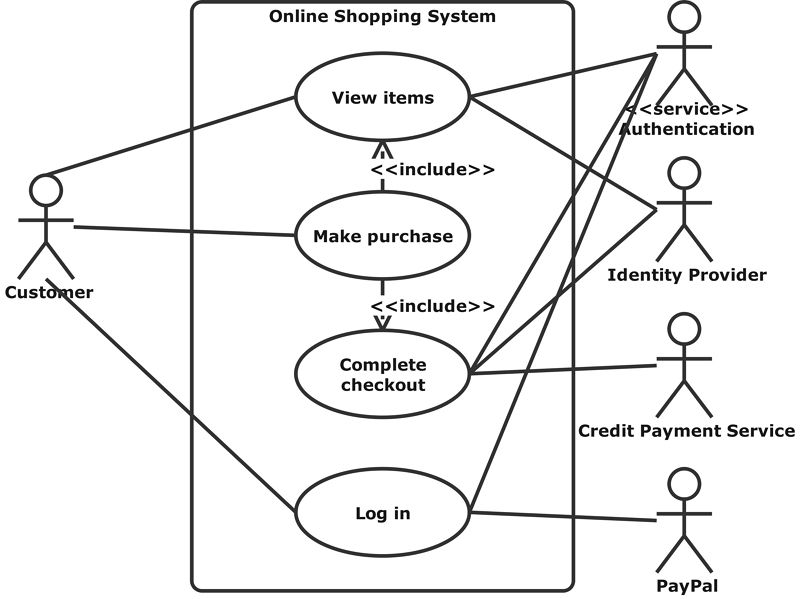
El modelo de información, o modelo conceptual, debe estar adecuadamente *justificado* a partir de los requisitos. No tiene sentido que en él aparezcan clases, atributos, operaciones y otros elementos que no hayan aparecido anteriormente en los requisitos. Igualmente, no tiene sentido que en los requisitos se mencionen conceptos importantes que no aparezcan reflejados de ninguna manera en el modelo conceptual. La referencia al *vocabulario del dominio* es nuevamente un puente importante que vincula los requisitos con el modelo conceptual.

## Diagrama de Casos de Uso

Seguir el estándar UML. Identificar todos los actores y representar sus funciones. Hay una buena explicación en: <https://www.edrawsoft.com/es/uml-use-case.html>

Mencionar la herramienta de modelado que se utiliza en la prácica para la elaboración de los distintos diagramas exigidos. Ejemplo de herramienta: edraw permite elaboración de diagramas on-line UML. <https://www.edrawsoft.com/es/edraw-uml.html>





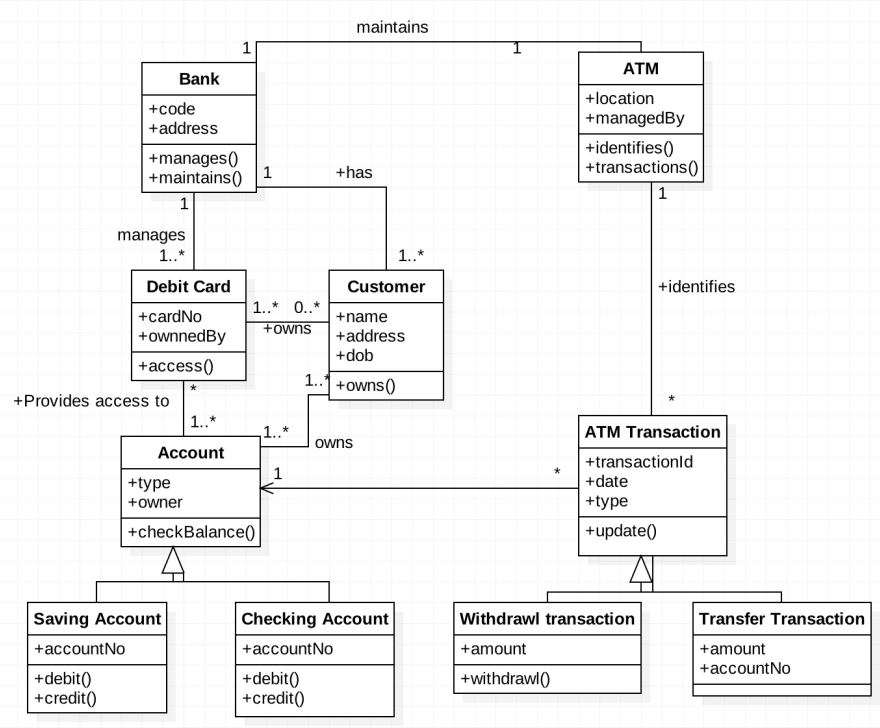
Ejempls de diagrama de Casos de uso. Fuente: <https://www.edrawsoft.com/es/uml-use-case.html>

## Diagramas de Clases

Los diagramas de clases deben ayudar a entender el sistema. En consecuencia, a estos diagramas no se les debe añadir tanta información que sean difíciles de comprender, ni tan poca información que no añadan nada a lo que ya se dice en los requisitos. Estos diagramas deben tener el grado de detalle suficiente para conservar su carácter comunicativo, en tanto que constituyen una vista gráfica de los requisitos. Por tanto, deberán contener todas las clases conceptuales (es decir, las que aparecen mencionadas en los requisitos), pero no las que sean exigencia exclusivamente del diseño posterior. Así mismo, estas clases aparecerán con los atributos que sean relevantes desde el punto de vista conceptual. Para aumentar la claridad y potencia expresiva, puede ser conveniente mostrar un diagrama global en el que sólo aparezcan las clases, y varios diagramas parciales en los que aparezcan clases con atributos.

No incluir las operaciones en las clases, para simplificar y disminuir el trabajo. Se trata de hacer un modelo de información.

Mencionar la herramienta de modelado que se utiliza en la prácica para la elaboración de los distintos diagramas exigidos. Ejemplo de herramienta: edraw permite elaboración de diagramas on-line UML. <https://www.edrawsoft.com/es/edraw-uml.html>



Ejemplo diagrama de clases (nota.- no incluir métodos en este proyecto) Fuente: <https://www.edrawsoft.com/es/example-uml-class-diagram.html>

# ARQUITECTURA

## Modelo de implementación

En esta sección se presenta y justifica la *arquitectura de desarrollo* elegida (descomposición del sistema en subsistemas y componentes), y se especifican las dependencias entre los distintos componentes que hayan resultado de la descomposición. La descomposición se hace habitualmente a varios niveles (componentes que a su vez se descomponen en subcomponentes, etc.), pero en esta práctica sólo se trata la descomposición de primer nivel, centrada en la vista externa (interfaces) de los componentes, y las relaciones entre ellos.

Se deben utilizar los diagramas y vistas que sean necesarios y las oportunas explicaciones textuales. Vale lo dicho en la sección 4.1 sobre la adecuada combinación de imágenes y palabras

Mencionar la herramienta de modelado que se utiliza en la prácica para la elaboración de los distintos diagramas exigidos. Ejemplo de herramienta: edraw permite elaboración de diagramas on-line UML. <https://www.edrawsoft.com/es/edraw-uml.html>

Interfaz de usuario gráfica

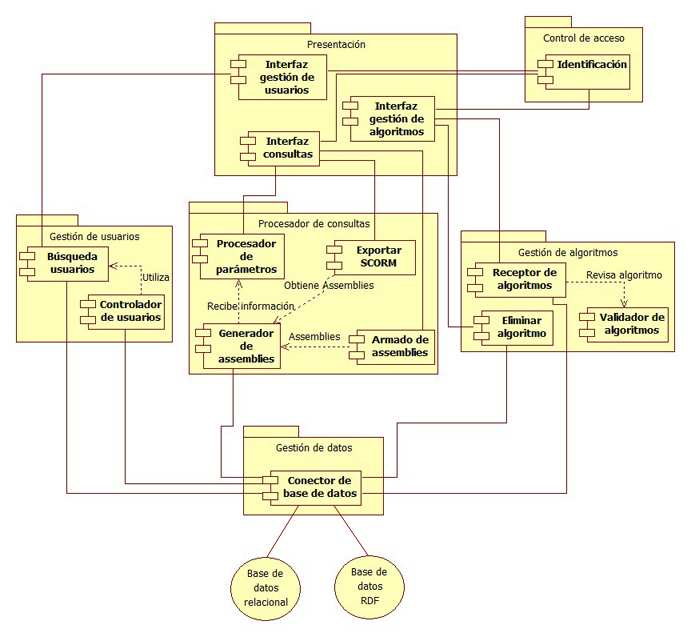
El contenido generado por IA puede ser incorrecto.

Ejemplo de modelo arquitectural en capas. Fuente: https://www.gluo.mx/blog/arquitectura-de-software-que-es-y-que-tipos-hay

## Diagrama de componentes

Diagrama

El contenido generado por IA puede ser incorrecto.



Ejhemplos diagramas de componentes. Fuente: <https://www.researchgate.net/figure/Figura-2-Diagramas-de-componentes-Con-el-proposito-de-ampliar-lo-descrito-por-el_fig2_255751413>

### Componente A (repetir para cada componente)

Por cada componente se repite la misma estructura con los siguientes apartados:

Tipo

Debe indicarse esencialmente si es ejecutable o no ejecutable. Componentes no ejecutables serían archivos de configuración, plantillas de páginas web, etc. El contenido de los demás apartados de la descripción del componente depende del tipo de componente: si es ejecutable, habrá que hacer énfasis en el Procesamiento (por ejemplo, métodos ofrecidos por las clases de implementación); si no es ejecutable, en los Datos (por ejemplo, información contenida en una página web). Los apartados de la descripción no son fijos, sino que deben adaptarse al tipo de componente.

Propósito [Función, Procesamiento, Datos, Recursos]

El propósito del componente se define esencialmente trazándolo contra los requisitos que el componente implementa. Esta trazabilidad hacia atrás justifica la existencia del componente y explica su propósito. Este apartado contiene una breve descripción textual del propósito del componente, basada en esos requisitos. El detalle de la trazabilidad se especifica en la sección 5.3.

Dependencias (interfaces requeridas)

Se definen las dependencias respecto a interfaces proporcionadas por otros componentes del sistema, o por sistemas externos. En el primer caso, dado que toda *interfaz requerida* tiene su contrapartida en una *interfaz proporcionada*, no es necesario repetir la especificación de la interfaz en ambos lugares; basta con especificarla como interfaz proporcionada y referenciarla adecuadamente en el otro lugar donde es requerida. En el segundo caso, sí es necesario especificar completamente la interfaz requerida, puesto que la interfaz proporcionada queda fuera del sistema.

Interfaces (Interfaces proporcionadas)

De modo general una interfaz es un conjunto de operaciones que ofrecen un servicio coherente. En diseño orientado a objetos se va más allá: una interfaz *define un tipo*, que proporciona un conjunto coherente de operaciones sobre las instancias compatibles con ese tipo.

La definición de interfaces permite aislar componentes y lograr un diseño modular y mantenible mediante interacciones bien definidas. Para definir correctamente el tipo especificado por la interfaz, se enumeran las operaciones pertenecientes a la interfaz, así como los contratos que deben satisfacer cada una de las operaciones. Para especificar los contratos debe utilizarse la técnica de *diseño por contratos* vista en el curso.

# LISTAS Y TABLAS

## Consistencia entre requisitos: conflictos, redundancias, acoplamientos

## Trazabilidad requisitos – modelo conceptual (clases)

## Trazabilidad requisitos – modelo de implementación (componentes)

En estas tres secciones pueden utilizarse tablas de doble entrada o tablas de 2-3 columnas. La tabla de doble entrada tiene el peligro de ser excesivamente dispersa, por lo que a menudo las tablas simplificadas de pocas columnas son una solución más conveniente.

# ANEXOS

## Anexo I – Bibliografía

Lista completa de *todos los documentos citados* (estándares, manuales, páginas web, herramientas utilizadas, etc.), identificados por su autor, título y fecha, u otros datos análogos que permitan su localización unívoca (título de la revista u organización que lo ha publicado, URL y fecha de descarga, etc.).

Nota.- Usar algún estándar. Ver información sobre estándares bibliográficos en: <https://uc3m.libguides.com/guias_tematicas/citas_bibliograficas/inicio>. MS-Word permite el uso de estos estándares (ver pestaña Referencias – Citas y bibliografía)

## Anexo II – Uso de IA en el proyecto

Resumen de herramientas IA usadas en el proyecto y para qué se han usado. Incorporar también principales prompts y comentarios a los resultados

### Herramientas y uso

### Prompts relevantes utilizados

### Comentarios y valoración de las herramientas