UNIVERSIDAD DE CASTILLA-LA MANCHA

ESCUELA SUPERIOR DE INFORMÁTICA

Trabajo teórico 1

Adrián Díaz-Plaza Remesal
Victor Cerrillo Arévalo
Imane El Harradji Auory
Chorouk Chemmari
Arturo González Rojas
Sergey Ghukasyan Poghosyan

Asignatura: Ingeniería del Software II

Grupo de Titulación (25/26): Grupo 3b1

Titulación: Grado en Ingeniería Informática

Fecha: 14/11/2025

ÍNDICE

Proceso Unificado de Desarrollo (PUD):

En esta parte se definen los requisitos, los casos de uso, su clasificación entre cliente y servidor, los grupos funcionales resultantes, las iteraciones del PUD con su coste y esfuerzo estimado, y la planificación temporal del proyecto.

El sistema sigue una arquitectura cliente-servidor distribuida, por lo que cada caso de uso puede pertenecer al cliente, al servidor o a ambos.

1. Tipos de usuarios:

1. **Usuarios Estudiantes**
2. **Usuarios Docentes e Investigadores**
3. **Personal Bibliotecario**
4. **Administradores del Sistema**

2. Requisitos Funcionales:

Los siguientes requisitos funcionales aplican al sistema completo.

Dado que el software está diseñado bajo una arquitectura cliente-servidor, la implementación de cada requisito se distribuye entre el cliente (interfaz y comunicación) y el servidor (procesamiento y almacenamiento).

En la siguiente sección se indicará para cada caso de uso si pertenece al cliente, servidor o ambos.

- **RF01:** El sistema debe permitir el inicio de sesión mediante el sistema de autenticación única (SSO) de la Universidad de Castilla-La Mancha.
- **RF02:** El sistema debe integrarse con los sistemas de gestión académica de alumnos y profesorado de la universidad.
- **RF03:** El sistema debe permitir la gestión de roles y permisos para diferentes tipos de usuarios.
- **RF04:** El sistema debe permitir la búsqueda y consulta del catálogo bibliográfico (libros, revistas, tesis, recursos digitales, etc.).
- **RF05:** El sistema debe permitir realizar búsquedas filtradas por asignatura, autor, título, tema o tipo de material.
- **RF06:** El sistema debe permitir la reserva y renovación de préstamos de materiales bibliográficos.
- **RF07:** El sistema debe permitir el acceso a recursos electrónicos (bases de datos, e-books).
- **RF08:** El sistema debe gestionar el historial de préstamos de cada usuario.
- **RF09:** El sistema debe permitir la solicitud de adquisición de materiales bibliográficos.
- **RF10:** El sistema debe enviar notificaciones automáticas sobre vencimiento de préstamos, disponibilidad de reservas y otras alertas relevantes.

- **RF11:** El sistema debe permitir la gestión de uso y reserva de espacios físicos (salas de reuniones, puestos de estudio, etc.).
- **RF12:** El sistema debe permitir la gestión del préstamo de equipos o hardware (portátiles, tablets, etc.).
- **RF13:** El sistema debe permitir la creación y gestión de grupos de estudio o clubs de lectura.
- **RF14:** El sistema debe permitir la gestión de bibliografía recomendada para asignaturas y proyectos de investigación.
- **RF15:** El sistema debe permitir la gestión de suscripciones y recursos asociados a proyectos de investigación.
- **RF16:** El sistema debe permitir la consulta de estadísticas de uso de recursos de los alumnos, o por parte de los investigadores del grupo.
- **RF17:** El sistema debe permitir la generación de informes bibliométricos y de actividad investigadora.
- **RF18:** El sistema debe permitir el alta, baja y modificación de usuarios de la biblioteca.
- **RF19:** El sistema debe permitir el alta, baja y modificación de registros bibliográficos.
- **RF20:** El sistema debe permitir la gestión de préstamos, devoluciones, reservas y renovaciones por parte del personal bibliotecario.
- **RF21:** El sistema debe permitir el control e inventario de materiales físicos y electrónicos.
- **RF22:** El sistema debe generar informes operativos sobre el uso de los recursos hardware y de recursos electrónicos.
- **RF23:** El sistema debe permitir la configuración de parámetros generales (políticas de préstamo, horarios, etc.).
- **RF24:** El sistema debe permitir la auditoría de actividades.
- **RF25:** El sistema debe permitir la generación de estadísticas globales y reportes de uso del sistema.
- **RF26:** El sistema debe permitir la integración con otros sistemas universitarios (p. ej., académicos).
- **RF27:** El sistema debe permitir la gestión de actualizaciones.

3. Requisitos No Funcionales

Los requisitos no funcionales definen las propiedades de calidad y las restricciones del sistema.

Estos requisitos garantizan la seguridad, disponibilidad, rendimiento y usabilidad del sistema bibliotecario, así como su compatibilidad con diferentes entornos tecnológicos.

- **RNF01:** El sistema debe garantizar una disponibilidad mínima durante su horario operativo.

- **RNF02:** El sistema debe ser accesible desde distintos sistemas operativos (Windows, Linux, macOS, Android, iOS).
- **RNF03:** El sistema debe estar desarrollado bajo una arquitectura cliente-servidor mediante una arquitectura distribuida.
- **RNF04:** El sistema debe soportar un número de usuarios concurrentes sin degradación significativa del rendimiento.
- **RNF05:** El sistema debe cumplir con la normativa de protección de datos **(RGPD)**.
- **RNF06:** El sistema debe implementar un control de acceso basado en roles y permisos definidos.
- **RNF07:** El sistema debe registrar logs de auditoría para operaciones críticas y accesos de usuarios.
- **RNF08:** La interfaz del sistema debe ser intuitiva y accesible.
- **RNF09:** El sistema debe ser responsive, adaptándose a diferentes tamaños de pantalla.
- **RNF10:** El sistema debe permitir actualizaciones sin afectar a la disponibilidad del servicio.
- **RNF11:** El sistema debe realizar copias de seguridad automáticas diarias.
- **RNF12:** El sistema debe poder **recuperarse de los fallos** de manera automática o asistida.
- **RNF13:** El sistema debe permitir la integración mediante APIs con servicios externos.
- **RNF14:** El diseño del sistema debe facilitar la escalabilidad para atender mayor demanda futura.
- **RNF15:** El sistema debe ofrecer mensajes de error claros y comprensibles para el usuario final.
- **RNF16:** El sistema debe permitir el acceso desde diferentes plataformas (aplicaciones web, móviles y de escritorio).

4. Casos de uso

Req	CDU	Prioridad	Requisitos	Análisis	Diseño	Implementación	Pruebas	Tipo(clinte /servidor)
RF1	CDU1	1	3	4	5	6	3	ambos
RF2	CDU2	1	3	5	6	7	4	ambos
RF3	CDU3	1	2	3	5	5	3	servidor
RF4	CDU4	1	2	4	4	6	3	cliente
RF5	CDU5	1	1	2	3	5	2	cliente
RF6	CDU6	1	2	3	5	5	3	cliente
RF7	CDU7	2	2	4	5	6	3	cliente
RF8	CDU8	2	1	3	3	4	2	cliente
RF9	CDU9	4	1	2	2	3	2	servidor
RF10	CDU10	2	1	2	3	5	2	cliente

RF11	CDU11	4	1	2	2	4	1	servidor
RF12	CDU12	4	1	2	4	4	2	servidor
RF13	CDU13	4	1	1	2	3	1	cliente
RF14	CDU14	3	2	3	3	4	3	servidor
RF15	CDU15	4	1	2	2	3	2	servidor
RF16	CDU16	3	1	3	3	4	2	servidor
RF17	CDU17	4	1	2	3	3	1	servidor
RF18	CDU18	2	2	3	5	6	3	servidor
RF19	CDU19	2	2	3	4	5	2	servidor
RF20	CDU20	2	2	3	4	5	2	servidor
RF21	CDU21	2	1	2	3	4	2	servidor
RF22	CDU22	4	1	1	2	3	1	servidor
RF23	CDU23	3	1	2	3	4	2	ambos
RF24	CDU24	2	2	3	4	5	3	servidor
RF25	CDU25	4	1	1	2	3	1	servidor
RF26	CDU26	2	2	3	5	6	3	servidor
RF27	CDU27	4	1	1	2	3	1	servidor

Justificación:

En la columna “Tipo (Cliente/Servidor)” se indica el componente del sistema responsable de la implementación de cada caso de uso.

- Los casos de uso de tipo **Cliente** corresponden a la interacción directa con el usuario, como búsquedas, consultas o reservas.
- Los de tipo **Servidor** están relacionados con la gestión de datos, seguridad, mantenimiento o generación de informes.
- Algunos casos, como la **autenticación** o la **integración de servicios**, implican participación de ambos componentes y se etiquetan como **Ambos**.

Esta clasificación permite diferenciar las responsabilidades entre cliente y servidor sin duplicar los casos de uso.

Prioridades:

- *Prioridad 1:* Casos de uso que son críticos para el funcionamiento básico del sistema.
- *Prioridad 2:* Casos de uso necesarios para el funcionamiento correcto y completo del sistema, son menos críticos que los de prioridad 1.
- *Prioridad 3:* Son casos de uso que contienen funcionalidades importantes, pero no son imprescindibles para el funcionamiento básico del sistema.
- Prioridad 4: Casos de uso opcionales, que agregan mejoras o funcionalidades extra.

5. Grupos Funcionales

Se agrupan los casos de uso del cliente y del servidor en grupos funcionales que comparten objetivos o trabajan sobre los mismos datos.

Cada grupo funcional contendrá casos de ambos lados del sistema, lo que permite desarrollar iteraciones conjuntas de cliente y servidor.

Grupo Funcional	Casos de Uso Cliente	Casos de Uso Servidor	RF Asociados
GF1 – Inicio de sesión SSO	CDU1C	CDU1S	RF01
GF2 – Integración con sistemas de gestión	CDU2C	CDU2S	RF02
GF3 – Configuración de parámetros generales	CDU23C	CDU23S	RF23

Los grupos funcionales se han definido únicamente para los requisitos que tienen casos de uso tanto en el cliente como en el servidor, ya que estos requieren coordinación entre ambos componentes para su correcto desarrollo.

El resto de los casos de usos de uso que se implementan solo en cliente o en el servidor se documentan de forma individual, dado que no requieren coordinación.

6. Proceso unificado de desarrollo

SUPUESTOS:

- **Equipo:**
  - 2 especialistas en requisitos (requisitos y análisis): Imane y Chorouk
  - 2 programadores (diseño e implementación): Victor y Sergio
  - 2 testers : Adrián y Arturo
- **Costos por hora:**
  - Gestión de requisitos: 50 €/hora
  - Analista: 70 €/hora
  - Diseñador: 60 €/hora
  - Implementador: 30 €/hora
  - Tester: 20 €/hora

- **Tareas realizables por dos o más personas a la vez.**
  Permite paralelizar las tareas complejas para acortar la duración del proyecto lo máximo posible.

- **Tareas no solapables.**
  Evita conflictos de dependencias y asegura que los resultados de una tarea están disponibles antes de empezar con la siguiente.
- **Una iteración puede empezar el mismo día que termina la anterior.**

Debido a que el proyecto cuenta con numerosas iteraciones, no aplicar esta medida generaría tiempos muertos entre ellas, incrementando innecesariamente la duración total del proyecto y reduciendo la eficiencia del equipo.

- **Una fase no puede comenzar el mismo día que termina la anterior.**
  De esta manera se garantiza una transición formal entre fases, garantizando que todo se haya hecho de manera correcta y reduciendo riesgos antes de pasar a la siguiente fase.
- **Jornada laboral de 8 horas diarias.**
  La jornada laboral de 8 horas se establece como equilibrio entre productividad y bienestar del equipo, garantizando un rendimiento óptimo durante el tiempo de trabajo.
- **Los findes de semana y festivos no se trabaja.**
  No se trabaja fines de semana ni festivos para reflejar la disponibilidad real del equipo, asegurar descansos adecuados y planificar de manera realista la duración del proyecto.

ITERACIONES:

IT1	IT2	IT3	IT4	IT5	IT6	IT7	IT8	IT9	IT10	IT11	IT12	IT13	IT14
GF1	GF2	CDU 3	CDU 4	CDU 5	CDU 6	CDU 7	CDU 8	CDU10	CDU18	CDU19	CDU20	CDU21	CDU24

IT15	IT16	IT17	IT18	IT19	IT20	IT21	IT22	IT23	IT24	IT25	IT26	IT27
CDU 26	CDU 14	CDU16	GF3	CDU 9	CDU 11	CDU 12	CDU 13	CDU 15	CDU 17	CDU 22	CDU 25	CDU 27

Iteración 0					
	R	A	D	I	P
Imane	5	3			
Chorouk	5	2			
Sergey			1	1	
Victor			1		
Adrián					
Arturo					
Horas	18				
Coste	1000				
Agenda	2 dias y 2 horas				

Iteración 1 – GF1					
	R	A	D	I	P
Imane	2	2			
Chorouk	1	2			
Sergey			2	3	
Victor			3	3	
Adrián					2
Arturo					1
Horas	21				
Coste	970				
Agenda	1 dia y 3 horas				

Iteración 2 – GF2					
	R	A	D	I	P
Imane	1	3			
Chorouk	2	2			
Sergey			3	4	
Victor			3	3	
Adrián					2
Arturo					2
Horas	25				
Coste	1150				
Agenda	1 dia y 6 horas				

Iteración 3 – CDU3					
	R	A	D	I	P
Imane	1	2			
Chorouk	1	3			
Sergey			2	3	
Victor			3	2	
Adrián					1
Arturo					2
Horas	18				
Coste	820				
Agenda	1 dia y 4 horas				

Iteración 4 – CDU4					
	R	A	D	I	P
Imane	1	2			
Chorouk	1	2			
Sergey			2	3	

Victor			2	3	
Adrián					2
Arturo					1
Horas	19				
Coste	860				
Agenda	1 dia y 2 horas				

Iteración 5 – CDU5					
	R	A	D	I	P
Imane		1			
Chorouk	1	1			
Sergey			1	2	
Victor			2	3	
Adrián					1
Arturo					1
Horas	13				
Coste	560				
Agenda	1 dia				

Iteración 6 – CDU6					
	R	A	D	I	P
Imane	1	1			
Chorouk	1	2			
Sergey			3	2	
Victor			2	3	
Adrián					1
Arturo					2
Horas	18				
Coste	820				
Agenda	1 dia y 3 horas				

Iteración 7 – CDU7					
	R	A	D	I	P
Imane	1	2			
Chorouk	1	2			
Sergey			2	3	
Victor			3	3	
Adrián					2
Arturo					1
Horas	18				
Coste	880				

Agenda	1 dia y 3 horas

Iteración 8 – CDU8					
	R	A	D	I	P
Imane		2			
Chorouk	1	1			
Sergey			2	2	
Victor			1	2	
Adrián					1
Arturo					1
Horas	12				
Coste	570				
Agenda	1 día				

Iteración 9– CDU10					
	R	A	D	I	P
Imane		1			
Chorouk	1	1			
Sergey			1	2	
Victor			2	3	
Adrián					1
Arturo					1
Horas	12				
Coste	540				
Agenda	1 día				

Iteración 10 - CDU18					
	R	A	D	I	P
Imane	1	1			
Chorouk	1	2			
Sergey			3	3	
Victor			2	3	
Adrián					1
Arturo					2
Horas	19				
Coste	850				
Agenda	1 dia y 3 horas				

Iteración 11 – CDU19					
	R	A	D	I	P
Imane	1	1			
Chorouk	1	2			

Sergey			2	2	
Victor			2	3	
Adrián					1
Arturo					1
Horas	16				
Coste	740				
Agenda	1 día y 1 hora				

Iteración 12 – CDU20					
	R	A	D	I	P
Imane	1	2			
Chorouk	1	1			
Sergey			2	3	
Victor			2	2	
Adrián					1
Arturo					1
Horas	16				
Coste	740				
Agenda	1 día y 1 hora				

Iteración 13 – CDU21					
	R	A	D	I	P
Imane	1	1			
Chorouk		1			
Sergey			1	2	
Victor			2	2	
Adrián					1
Arturo					1
Horas	11				
Coste	530				
Agenda	7 horas				

Iteración 14 – CDU24					
	R	A	D	I	P
Imane	1	1			
Chorouk	1	2			
Sergey			2	2	
Victor			2	3	
Adrián					1
Arturo					2

Horas	17
Coste	760
Agenda	1 dia y 2 horas

Iteración 15 – CDU26					
	R	A	D	I	P
Imane	1	1			
Chorouk	1	2			
Sergey			3	3	
Victor			2	3	
Adrián					1
Arturo					2
Horas	19				
Coste	850				
Agenda	1 dia y 3 horas				

Iteración 16 – CDU14					
	R	A	D	I	P
Imane	1	2			
Chorouk	1	1			
Sergey			1	2	
Victor			2	2	
Adrián					1
Arturo					2
Horas	15				
Coste	670				
Agenda	1 dia y 1 hora				

Iteración 17 - CDU16					
	R	A	D	I	P
Imane		2			
Chorouk	1	1			
Sergey			2	2	
Victor			1	2	
Adrián					1
Arturo					1
Horas	13				
Coste	600				
Agenda	1 día				

Iteración 18 – GF3					
	R	A	D	I	P
Imane		1			
Chorouk	1	1			
Sergey			1	2	
Victor			2	2	
Adrián					1
Arturo					1
Horas	12				
Coste	530				
Agenda	7 horas				

Iteración 19 – CDU9					
	R	A	D	I	P
Imane	1	1			
Chorouk		1			
Sergey			1	1	
Victor			1	2	
Adrián					1
Arturo					1
Horas	9				
Coste	420				
Agenda	6 horas				

Iteración 20 – CDU11					
	R	A	D	I	P
Imane	1	1			
Chorouk		1			
Sergey			1	2	
Victor			1	2	
Adrián					1
Arturo					
Horas	10				
Coste	450				
Agenda	6 horas				

Iteración 21 – CDU12					
	R	A	D	I	P
Imane	1	1			
Chorouk		1			

Sergey			2	2	
Victor			2	2	
Adrián					1
Arturo					1
Horas	13				
Coste	590				
Agenda	7 horas				

Iteración 22 – CDU13					
	R	A	D	I	P
Imane	1				
Chorouk		1			
Sergey			1	1	
Victor			1	2	
Adrián					1
Arturo					
Horas	8				
Coste	350				
Agenda	6 horas				

Iteración 23 – CDU15					
	R	A	D	I	P
Imane		1			
Chorouk	1	1			
Sergey			1	2	
Victor			1	1	
Adrián					1
Arturo					1
Horas	10				
Coste	420				
Agenda	6 horas				

Iteración 24 - CDU17					
	R	A	D	I	P
Imane	1	1			
Chorouk		1			
Sergey			1	2	
Victor			2	1	
Adrián					
Arturo					1
Horas	10				

Coste	480
Agenda	7 horas

Iteración 25 – CDU22					
	R	A	D	I	P
Imane	1				
Chorouk		1			
Sergey			1	2	
Victor			1	1	
Adrian					1
Arturo					
Horas	18				
Coste	350				
Agenda	6 horas				

Iteración 26 – CDU25					
	R	A	D	I	P
Imane	1				
Chorouk		1			
Sergey			1	2	
Victor			1	1	
Adrián					1
Arturo					
Horas	8				
Coste	350				
Agenda	6 horas				

Iteración 27 – CDU27					
	R	A	D	I	P
Imane	1				
Chorouk		1			
Sergey			1	2	
Victor			1	1	
Adrián					
Arturo					1
Horas	8				
Coste	350				
Agenda	6 horas				

Iteración 28					
	R	A	D	I	P
Imane		1			
Chorouk		1			
Sergey			1	5	
Victor				5	
Adrián					37
Arturo					38
Horas	88				
Coste	2000				
Agenda	11 días				

Fases

En este apartado se encuentra la división de las diferentes fases del proyecto con sus respectivos costes y horas.

FASES	INICIO	ELABORACION	CONSTRUCCION	TRANSICIÓN
Iteraciones	It0	It1-it6	It7-it27	It28
Costes	1000	6000	12020	2000
Horas	18	66	153	88
Días	3	9	20	11

Calendario

Aquí se muestra la división del tiempo total en las diferentes iteraciones teniendo en cuenta los días festivos y los fines de semana.

Inicio : Elaboración : Construcción : Transición :

MES	L	M	X	J	V
NOV	10	11	12 (it0)	13 (it0)	14 (it0)
NOV	17 (it1)	18 (it1 - it2)	19 (it2)	20 (it2 – it3)	21 (it3 – it4)
NOV	24 (it4 – it5)	25 (it5 – it6)	26 (it6)	27 (it6 – it7)	28 (it7-it8)
DIC	1 (it8-it9)	2 (it9 - it10)	3 (it10)	4 (it10-it11)	5(it11-it12)
DIC	8 (festivo)	9 (it12-it13)	10 (it13-it14)	11 (it14-it15)	12 (it15)
DIC	15 (it15-it16)	16 (it16-it17)	17 (it17-it18)	18 (it18-it19-it20)	19 (it20 – it21)
DIC	22 (it21-it22)	23 (it22-it23)	24 (festivo)	25 (festivo)	26 (it24-it25)
DIC/EN	29(it26)	30 (it27)	31 (festivo)	1 (festivo)	2 (it28)
EN	5 (it28)	6 (festivo)	7 (it28)	8 (it28)	9 (it28)
EN	12 (it28)	13 (it28)	14 (it28)	15 (it28)	16 (it28)
EN	19 (it28)	20	21	22	23

Plan de Gestión de Configuración

1. Alcance

- **Propósito del plan:** Nuestro objetivo con este plan es llevar un seguimiento de versiones que esperamos obtener del desarrollo del software, así como las limitaciones del plan y responsabilidades que tenemos. El alcance del plan coincide con el del proyecto: Desarrollo del software que nos ha pedido el cliente.
- **Elementos de configuración:** Son los componentes que desarrollar del software. Cada componente debe coincidir con el software a realizar en cada iteración.
- **Limitaciones y suposiciones:**
    a. Sábados y domingos no se trabaja.
    b. Fecha máxima de entrega: 19 de enero de 2026.
    c. Se dispone de 6 recursos humanos, cada uno con distintos costes.
    d. Jornada laboral de 8 horas.
    e. También pueden afectar todas aquellas suposiciones tomadas con anterioridad al planificar el PUD

2. Organización

- **Autoridad:** El coordinador es la máxima autoridad y responsable del proyecto, su trabajo consiste en verificar que las actividades de la gestión de configuración sean planificadas y ejecutadas correctamente.
- **Responsabilidades:** Todos los RRHH se comprometen con el cumplimiento del plan a lo largo del desarrollo del software en cada una de sus áreas asignadas.
- **Agenda:** Debe coincidir con lo planeado anteriormente en el PUD. Las fechas fijadas son su versión estimada.
- **Versionado:** Semantic versioning.

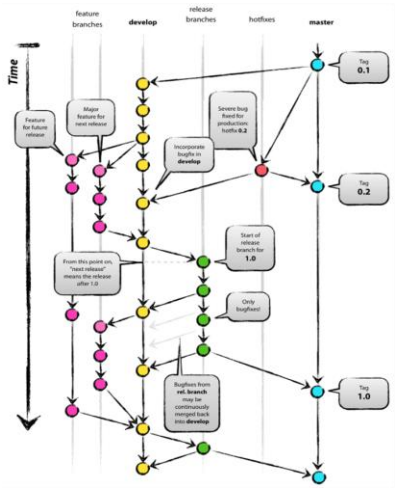
3. Ramificación:

Hemos decidido usar la ramificación usada por la propia comunidad de GitHub: GitFlow. La cual se basa en tener dos ramas para el desarrollo principal de la aplicación, una para el cliente y otra para el servidor. Las ramas de desarrollo se ramifican en otras ramas encargadas de cosas concretas, las ramas de Características, mientras la de desarrollo sigue con lo principal y esencial. Llegado un punto dado, las ramas de características, se transfieren a la de desarrollo y esta a su vez es transportada su contenido a la de Lanzamiento, ahí se harán las pruebas pertinentes sobre su funcionamiento, en caso de fallar, se envía el código de nuevo a Desarrollo, si esta todo correcto, se envía a la rama main. En caso de que el código en la rama main tenga fallos o no cumplen con los requisitos del cliente, en la rama Hotfix se elaborará un informe de cambios que hay que hacer con Prioridad 1, se llevara el proyecto de nuevo a desarrollo y ahí se aplicaran las correcciones. Nótese que hay una rama extra, llamada Documentación, esa rama funciona en paralelo al proyecto en general y es para ir guardado toda la documentación sobre el desarrollo,

se publicará luego el contenido de la rama en una carpeta llamada “doc” en la rama main. En total son:

a. 1 rama main
b. 2 rama DesarrolloPrincipal
c. 1 rama Hotfix
d. 1 rama de Lanzamiento
e. N ramas de DesarrolloCaracteristicas
f. 1 rama de Documentación

Imagen de referencia del modelo GitFlow:



4. Gestión de peticiones de cambio

Los cambios relacionados por cualquier miembro del equipo de desarrollo se aceptarán. Si alguna de esas peticiones puede afectar requisitos o funcionalidades críticas serán revisados por el coordinador y, si se consideran viables, aprobados por el equipo de desarrollo. Todas las solicitudes se documentarán para mantener trazabilidad.

5. Gestión de versiones

El cumplimiento del versionado será supervisado por los responsables y el coordinador.

6. Construcción del sistema

Ambos sistemas ejecutables (cliente y servidor) partirán de la V0.0.0.
Por cada componente desarrollado en cada sistema, se aumentará el número minor, siguiendo las reglas de Semantic Versioning.
De este modo:

- El cliente, compuesto por 9 componentes, alcanzará la V0.9.0
- El servidor, compuesto por 19 componentes, alcanzará la V0.19.0.

En caso de que se detecte un error en algún componente tras su desarrollo, se desarrollará su correspondiente hotfix y se incrementará el patch, sin alterar ni el minor ni el major.

Una vez terminada la fase de transición y ambos sistemas estén preparados para el despliegue, se realizará el lanzamiento oficial. De acuerdo con Semantic Versioning, se incrementará el número major, fijando la versión de lanzamiento para ambos sistemas en la V1.0.0

7. Gestión de Liberaciones/Entregas

El proyecto está planeado para ser finalizado el día 19/1/2026.
La etapa de transición comienza el 2/1/2025 y durante esa etapa se planifican las siguientes liberaciones previas:

- Versión Alpha: del 2/1/2025 al 10/1/2025
- Versión Beta: del 11/1/2025 al 17/01/2026

Tras las pruebas y correcciones de errores encontrados en ambas versiones, se realizará el lanzamiento oficial el día 19/01/2026

Gestión de Calidad

En este documento se detallan las características de calidad que se consideran esenciales para el desarrollo del sistema integral de gestión bibliotecaria de la Universidad de Castilla-La Mancha, justificando su selección, su impacto en los distintos tipos de usuarios y cómo se evaluarán.

1. Características de Calidad Seleccionadas

1.1. Seguridad

Porque: El sistema gestionará información académica y personal de estudiantes, docentes e investigadores, así como el control de préstamos, reservas y acceso a recursos electrónicos. Por tanto, la protección de los datos y el control de accesos son fundamentales.

Implementación:

- Control de acceso basado en roles (estudiantes, docentes, bibliotecarios, administradores).
- Autenticación integrada con el SSO de la universidad.
- Cifrado de datos tanto en tránsito como en reposo.
- Auditorías automáticas para registrar actividades del sistema.

1.2. Fiabilidad (subcaracterística de Disponibilidad y Tolerancia a Fallos)

Porque: El sistema debe estar disponible continuamente, especialmente durante el horario académico, y debe evitar pérdidas de información. Si el sistema falla, afectaría directamente a la gestión diaria de la biblioteca.

Implementación:

- Arquitectura cliente-servidor distribuida que permita redundancia.
- Backups automáticos de bases de datos y restauración programada.
- Manejo de excepciones y recuperación automática ante fallos.

- Monitorización de disponibilidad y alertas automáticas.

1.3. Usabilidad

Porque: Los usuarios del sistema son muy diversos (estudiantes, docentes, bibliotecarios y administradores), y la eficiencia del sistema dependerá de que todos puedan utilizarlo de manera intuitiva sin formación técnica avanzada.

Implementación:

- Interfaces personalizadas según el rol del usuario.
- Diseño accesible conforme a estándares WCAG.
- Navegación intuitiva, con menús adaptados y buscadores inteligentes.
- Notificaciones claras y mensajes de error comprensibles.
- Manuales y tutoriales integrados.

1.4. Compatibilidad (subcaracterística de Interoperabilidad)

Porque: El sistema debe integrarse con los servicios universitarios existentes (SSO, gestión académica, etc.) y además funcionar correctamente en diferentes plataformas (Windows, Linux, macOS, Android, iOS, web).

Implementación:

- APIs RESTful para interoperabilidad con sistemas de la universidad.
- Diseño multiplataforma usando tecnologías compatibles (React Native o Flutter).
- Gestión de dependencias para coexistir con otros sistemas institucionales sin conflicto.

2. Especificar Evaluación

A continuación, se definen los tipos de usuario identificados y el peso porcentual que cada uno otorga a las características de calidad seleccionadas.

Tipo de Usuario	Seguridad	Fiabilidad	Usabilidad	Compatibilidad
Estudiante	20%	25%	40%	15%
Docente/Investigador	25%	30%	25%	20%
Personal Bibliotecario	30%	35%	20%	15%
Administrador del Sistema	40%	30%	10%	20%

3. Diseñar Evaluación

Puntuaciones asignadas (0-100) para cada característica por tipo de usuario:

Tipo de Usuario	Seguridad	Fiabilidad	Usabilidad	Compatibilidad
Estudiante	80	85	90	80

Docente/Investigador	85	90	85	85
Personal Bibliotecario	90	90	80	75
Administrador del Sistema	95	85	75	90

4. Llevar a cabo la Evaluación

Cálculo de la media ponderada combinando el peso y la puntuación de cada característica:

Tipo de Usuario	Seguridad	Fiabilidad	Usabilidad	Compatibilidad	Media Ponderada
Estudiante	0.20×80=16	0.25×85=21.25	0.40×90=36	0.15×80=12	85.25
Docente/Investigador	0.25×85=21.25	0.30×90=27	0.25×85=21.25	0.20×85=17	86.5
Personal Bibliotecario	0.30×90=27	0.35×90=31.5	0.20×80=16	0.15×75=11.25	85.75
Administrador del Sistema	0.40×95=38	0.30×85=25.5	0.10×75=7.5	0.20×90=18	89.0

Conclusión: El usuario Administrador del Sistema es el más beneficiado con las características de calidad seleccionadas.

5. Finalizar la Evaluación (Informe de Calidad)

Se derivan los siguientes requisitos no funcionales de calidad que serán incorporados en la fase de desarrollo:

Seguridad

- Autenticación mediante SSO y control de accesos basado en roles.
- Cifrado de datos en tránsito y almacenamiento.
- Auditorías automáticas de actividades.

Fiabilidad

- Disponibilidad mínima del 100% durante el horario académico.
- Copias de seguridad automáticas diarias y restauración en menos de 15 minutos.
- Manejo de errores y recuperación automática ante fallos del servidor.

Usabilidad

- Interfaces adaptadas por tipo de usuario.
- Cumplimiento de estándares WCAG 2.1.
- Sistema de ayuda y tutoriales integrados.

Compatibilidad

- Arquitectura cliente-servidor multiplataforma.
- Interoperabilidad mediante APIs REST.
- Coexistencia con otros sistemas del entorno universitario.