Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso:  Ingeniería de Software 1



# Integrantes


- Daniel Alejandro Duitama Correa
- Edwin Felipe Pinilla Peralta
- Miguel Angel Martinez Fernandez (miamartinezfe@unal.edu.co)
- Juan Sebastián Umaña Camacho (juumanac@unal.edu.co)

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso:  Ingeniería de Software 1

# Levantamiento de Requerimientos

## 1. ¿De dónde surge la idea?

La idea del sistema de mensajería instantánea "UNión" surge de la necesidad de mejorar la comunicación académica entre estudiantes y profesores dentro de los cursos universitarios. Durante el desarrollo del proyecto, se identificó que las plataformas actuales carecen de una integración enfocada en la facilidad para comunicarse y organización específica por curso. La propuesta se basó en entrevistas con estudiantes, así como observación directa de los problemas que enfrentan en el ámbito académico, como la dispersión de información en múltiples canales y la falta de orden.

## 2. ¿Cómo se pusieron de acuerdo para elegir la idea?

El equipo realizó una lluvia de ideas en la que cada miembro propuso un problema académico relevante (transporte, comunicación). Tras discutir las propuestas, se optó por desarrollar un sistema de mensajería por ser una solución práctica y de impacto inmediato donde se pueden utilizar los conocimientos adquiridos durante el curso.

## 3. ¿Cuáles son las problemáticas que buscan resolver?

- Falta de un canal sencillo y centralizado: Actualmente, la comunicación académica suele dispersarse entre plataformas como correos electrónicos, redes sociales y classroom, lo que dificulta la organización y privacidad, por eso se busca una aplicación web centralizada donde se trabajen estas problemáticas.
- Dificultades en la colaboración académica: Los estudiantes tienen problemas con canales de comunicación poco efectivos para interactuar y resolver dudas directamente con sus compañeros o profesores en un entorno formal, por lo qué una plataforma virtual donde cada estudiante esté inscrito automáticamente facilita esta y se promueva la participación con reviews.

## 4. ¿Qué expectativas tienen los usuarios potenciales del sistema?
## Los usuarios esperan:

- Profesores:
    - Un sistema fácil de usar para gestionar cursos y estudiantes.
    - Garantía de que solo los estudiantes autorizados accedan a la comunicación del curso.
    - Herramientas para mantener la privacidad y la organización de la mensajería.
- Estudiantes:
    - Acceso rápido a los cursos y sus comunicaciones asociadas.
    - Un entorno seguro para interactuar con compañeros y profesores.
    - Funcionalidades como notificaciones en tiempo real y búsqueda de mensajes.

### 5. ¿Qué beneficios esperan ustedes al desarrollar este proyecto? El equipo espera adquirir experiencia práctica en:

- Poner en práctica el diseño y desarrollo de sistemas seguros y escalables.
- Aplicación de conceptos avanzados como websockets, clean code y patrones de diseño.
- Resolución de problemáticas reales en entornos académicos, fortaleciendo nuestro perfil profesional.

## Funcionalidades identificadas

### Gestión de cursos (para profesores)

1. Crear un curso para organizar la comunicación de sus estudiantes.
2. Editar la información del curso, como nombre y descripción.
3. Eliminar cursos obsoletos o que ya no son necesarios.
4. Agregar estudiantes al curso mediante invitaciones o códigos de acceso.
5. Un módulo para subir archivos pertinentes al curso.

### Acceso y registro (para estudiantes)

6. Unirse a un curso al que fueron invitados utilizando un código único.
7. Acceder sólo a los cursos en los que están inscritos.
8. Se debe poder decidir qué tipo de notificaciones se quiere recibir.

### Mensajería en tiempo real

9. Los usuarios pueden buscar a otros estudiantes o profesores a través de un buscador que permite filtrar por nombre o correo institucional. Si desean iniciar un chat con una persona específica, deben enviarle una solicitud de amistad. La otra persona puede aceptar o rechazar la solicitud.
10. Enviar mensajes entre estudiantes y profesores dentro del curso.
11. Recibir notificaciones de mensajes nuevos y anuncios importantes.
12. Crear hilos de discusión organizados por temas dentro del curso.
13. Creación de encuestas con fecha límite.
14. Panel en donde se vean todas mis conversaciones, y filtro para encontrar rápidamente conversaciones individuales o grupales

### Roles y permisos

15. Definir roles específicos para profesores y estudiantes, asegurando que cada uno tenga acceso únicamente a las funcionalidades necesarias.
    - Profesores: gestión de cursos y moderación de mensajes.
    - Estudiantes: acceso a cursos y comunicación.

### Seguridad y privacidad

16. Garantizar que solo usuarios autorizados puedan acceder a la información del curso.

17. Proteger los datos de los usuarios mediante protocolos de autenticación segura.

## Adicionales

18. Módulo para coordinar transporte compartido entre usuarios del sistema.

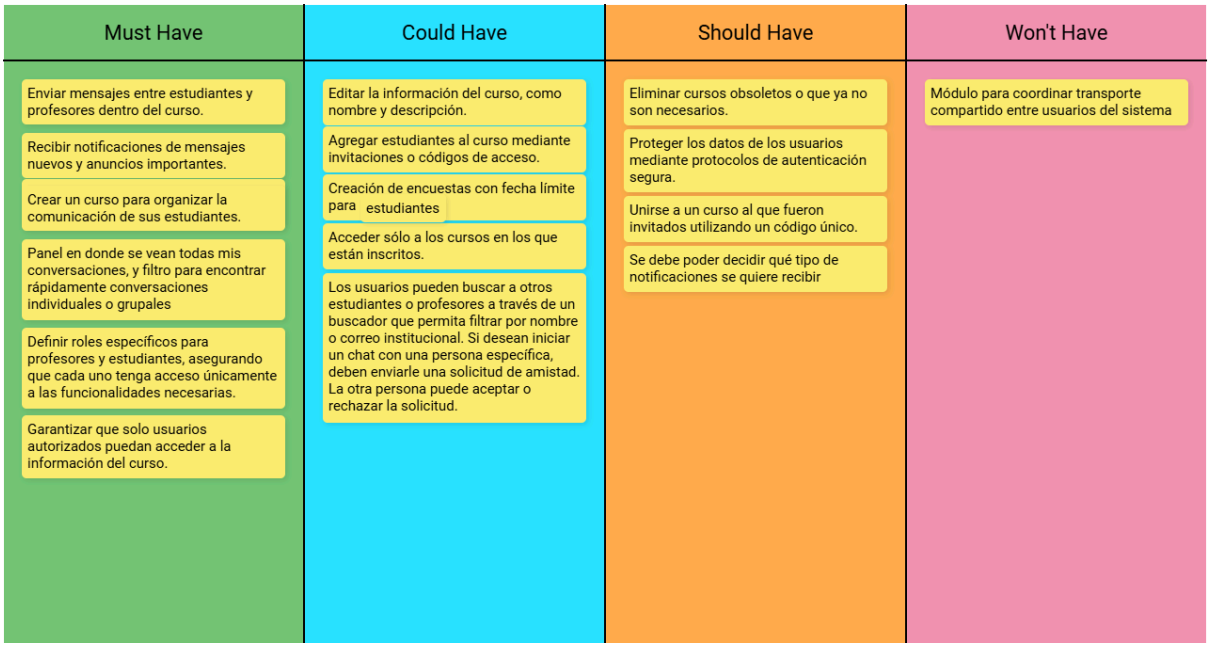
Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso:  Ingeniería de Software 1

# Análisis de requerimientos

## MoSCoW

Para el análisis de requerimientos se utilizó el análisis MoSCoW, para interpretar la siguiente tabla se siguieron los siguientes criterios:

- **Must Have**: requisitos críticos para la versión dada para que se considere un éxito.
- **Should Have**: requisitos de alta prioridad que deben incluirse en la versión determinada si es posible.
- **Could Have:** requisitos deseables, pero no necesarios para el éxito del lanzamiento determinado.
- **Won't Have:** los requisitos que las partes interesadas hayan acordado no se implementarán en la versión, pero pueden considerarse en el futuro.




## Estimación Fibonacci

| Tarea | Complejidad Técnica | Recursos Disponibles (1-13) | Impacto en la experiencia del Usuario | Relación con los Objetivos del Proyecto | Esfuerzo Total |
|---|---|---|---|---|---|

| | (1-13) | | (1-13) | (1-13) | |
|---|---|---|---|---|---|
| Enviar mensajes entre estudiantes y profesores dentro del curso. | 5 | 3 | 8 | 8 | 5 |
| Recibir notificaciones de mensajes nuevos y anuncios importantes. | 3 | 2 | 5 | 5 | 3 |
| Crear un curso para organizar la comunicación de sus estudiantes. | 8 | 5 | 8 | 8 | 8 |
| -Panel en donde se vean todas mis conversaciones, y filtro para encontrar rápidamente conversaciones individuales o grupales | 5 | 3 | 8 | 8 | 5 |
| Definir roles específicos para profesores y estudiantes, asegurando que cada uno tenga acceso únicamente a las funcionalidades necesarias. | 8 | 5 | 8 | 8 | 8 |
| -Garantizar que solo usuarios autorizados puedan acceder a la información del curso. | 5 | 3 | 8 | 8 | 5 |
| Editar la información del curso, como nombre y descripción. | 3 | 2 | 5 | 5 | 3 |

| | | | | | |
|---|---|---|---|---|---|
| Agregar estudiantes al curso mediante invitaciones o códigos de acceso. | 5 | 3 | 8 | 8 | 5 |
| Creación de encuestas con fecha límite dentro de una conversación o curso. | 3 | 2 | 5 | 5 | 3 |
| Acceder sólo a los cursos en los que están inscritos. | 3 | 2 | 5 | 5 | 3 |
| Los usuarios pueden buscar a otros a través de un buscador. Si desean iniciar un chat con una persona específica, deben enviarle una solicitud de amistad. | 5 | 3 | 8 | 8 | 5 |
| Proteger los datos de los usuarios mediante protocolos de autenticación segura. | 2 | 5 | 3 | 3 | 3 |

El cálculo del esfuerzo total se hizo promediando todas las columnas y luego redondeando al número de la secuencia fibonacci más cercano.

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1

# Gestión del software

| Tarea | Tiempo (Semanas) | Costo | Alcance |
|---|---|---|---|
| Módulo Gestión de Cursos<br>● Crear un curso para organizar la comunicación de sus estudiantes.<br>● Editar la información del curso, como nombre y descripción<br>● Eliminar cursos obsoletos o que ya no son necesarios<br>● Agregar estudiantes al curso mediante invitaciones o códigos de acceso<br>● Unirse a un curso al que fueron invitados utilizando un código único<br>● Acceder sólo a los cursos en los que están inscritos. | 4 | ● Desarrollador Junior (4 semanas) $450 USD<br>● Licencias de herramientas (IDE, Git, etc.) $100 USD<br>● Servicios en la nube (AWS, Google Cloud) $200 USD<br>● **Total $750 USD** | Incluido<br>● Crear, editar y eliminar cursos.<br>● Invitar estudiantes mediante códigos de acceso.<br>● Unirse a cursos con códigos únicos.<br>● Acceso restringido a cursos inscritos.<br>Excluido:<br>● Integración con sistemas externos (por ejemplo, plataformas de gestión educativa como Moodle).<br>● Funcionalidades avanzadas de personalización de cursos (por ejemplo, temas o diseños personalizados). |
| Módulo Comunicación y Mensajería<br>● Enviar mensajes entre estudiantes y profesores dentro del curso.<br>● Recibir notificaciones de mensajes nuevos y anuncios importantes.<br>● Crear hilos de discusión organizados por temas dentro del curso.<br>● Creación de encuestas con fecha límite. | 3 | ● Desarrollador Junior (3 semanas) $337,5 USD<br>● Servicios en la nube (almacenamiento y mensajería en tiempo real) $150 USD<br>● **Total $487,5 USD** | Incluido<br>● Mensajería en tiempo real entre estudiantes y profesores.<br>● Notificaciones de mensajes y anuncios.<br>● Hilos de discusión organizados por temas.<br>● Creación de encuestas con fecha límite.<br>Excluido:<br>● Mensajería de voz o video. |

| | | | |
|---|---|---|---|
| Módulo Gestión de Usuarios y Seguridad<br>● Definir roles específicos para profesores y estudiantes.<br>● Garantizar que solo usuarios autorizados puedan acceder a la información del curso.<br>● Proteger los datos de los usuarios mediante protocolos de autenticación segura. | 2 | ● Desarrollador Junior (2 semanas) $225 USD<br>● Servicios en la nube (autenticación y almacenamiento seguro) $150 USD<br>● **Total $375 USD** | Incluido<br>● Roles de usuario (profesor y estudiante).<br>● Autenticación segura (por ejemplo, OAuth)<br>● Control de acceso a la información del curso.<br>Excluido:<br>● Autenticación biométrica (huella digital o reconocimiento facial). |
| Módulo Búsqueda y Organización de Conversaciones<br>● Los usuarios pueden buscar a otros a través de un buscador.<br>● Panel donde se vean todas las conversaciones con un filtro para encontrar rápidamente chats individuales o grupales. | 2 | ● Desarrollador Junior (2 semanas) $225 USD<br>● **Total $225 USD** | Incluido<br>● Búsqueda de usuarios y conversaciones.<br>● Panel de conversaciones con filtros.<br>Excluido:<br>● Búsqueda avanzada (por ejemplo, por palabras clave o fechas). |

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso:  Ingeniería de Software 1

# Taller 2

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso:  Ingeniería de Software 1

# Patrones de diseño

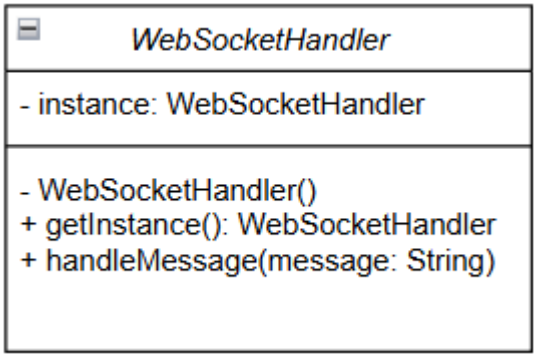
## Patrón Singleton

Para WebSocket Handler y servicios críticos

El patrón **Singleton** asegura que una clase tenga una única instancia en toda la aplicación y proporciona un punto de acceso global a esa instancia. En este caso, evita la creación de múltiples manejadores de WebSockets o instancias duplicadas de servicios críticos, lo que podría generar conflictos o un uso ineficiente de recursos, asegurando que todas las conexiones sean gestionadas de manera centralizada y eficiente.

- Asegura que servicios críticos, como la **gestión de sesiones**, mantengan un único punto de acceso y no haya inconsistencias entre diferentes instancias.
- Mejora la escalabilidad, ya que una sola instancia puede gestionar múltiples conexiones sin sobrecargar el sistema.

**Implementación:**




El constructor es privado para evitar la creación de múltiples instancias.

El método getInstance() garantiza que solo exista una instancia del manejador de WebSockets.

El método handleMessage() gestiona los mensajes recibidos a través de WebSocket.

Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso:  Ingeniería de Software 1

# Patrón Observer

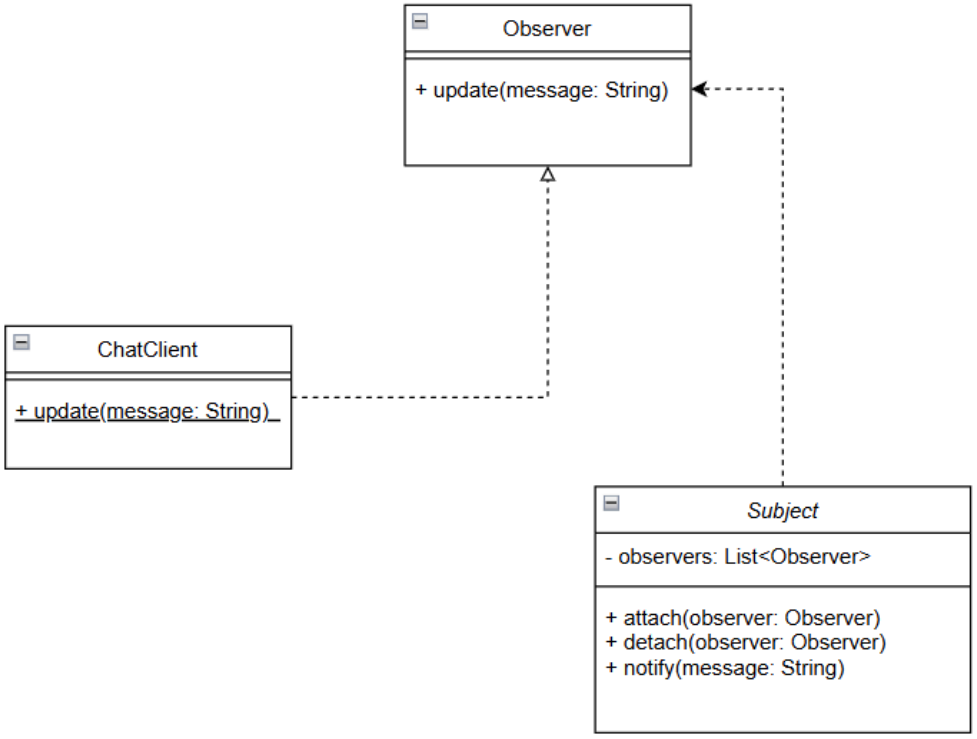
Para notificar eventos en tiempo real, para que los clientes (estudiantes/profesores) sean notificados automáticamente cuando hay nuevos mensajes o cambios en el estado de los chats.

Permite que múltiples partes del sistema sean notificadas automáticamente cuando ocurre un evento sin necesidad de consulta constante. Es ideal para la actualización en tiempo real de los chats cuando un usuario recibe un nuevo mensaje o cuando hay un cambio en el estado del chat (por ejemplo, que un usuario esté escribiendo).

- Queremos que los estudiantes y profesores sean notificados en tiempo real cuando reciben un nuevo mensaje o cuando hay actualizaciones en los chats.
- Evita que los usuarios tengan que hacer peticiones constantes al servidor para verificar cambios, lo que optimiza el rendimiento.
- Facilita la integración con tecnologías como **WebSockets**..

**Implementación:**

“Subject” es la clase que mantiene una lista de observadores (observers) y notifica cambios.
“Observer” es una interfaz que define el método update().
“ChatClient” es una implementación concreta de Observer que recibe notificaciones cuando hay nuevos mensajes.



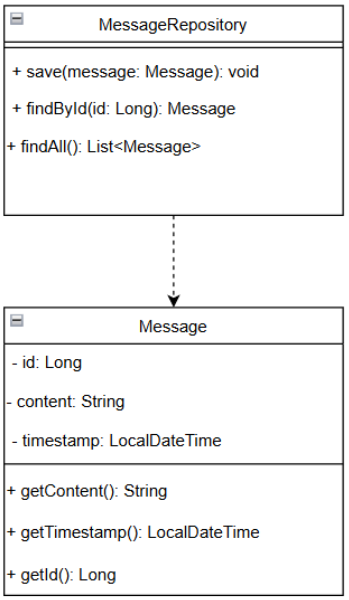
# Patrón Repository

Para la capa de persistencia.

Separa la lógica de acceso a la base de datos del resto de la aplicación, lo que facilita el mantenimiento y la prueba del código. Evita que los controladores y servicios accedan directamente a la base de datos.

- Nos permite manejar la persistencia de los mensajes y usuarios sin acoplar el código a una base de datos específica.
- Facilita el cambio de la base de datos en el futuro sin modificar la lógica de negocio.
- Mejora la organización del código y permite realizar pruebas unitarias sin necesidad de acceder a la base de datos real.

**Implementación:**




“MessageRepository” es una interfaz que define los métodos para acceder a la base de datos.
“Message” es una entidad que representa un mensaje en el sistema.