



Benemérita Universidad Autónoma de Puebla
Facultad de Ciencias de la Computación



Ingeniería de software I
Docente: Juan Manuel Gonzales Calleros

Documento del proyecto



Mejía Ramírez José Carlos
Moreno Gaytán Alhelí
Núñez Grajales Erick
Oropeza López Milcha
Paniagua Velázquez Martín Antonio
Vidal Lumbreras Ramiro

Fecha
02/12/2020

Índice

Hoja de control de cambios	5
Historia de la ingeniería de software	10
Mitos del software	11
La importancia de la ingeniería de software	12
Clasificación del software	14
Software de aplicación	14
Software de sistema	14
Software de programación	14
Problemas generales que afectan al software	15
Heterogeneidad	15
Cambio empresarial y social	15
Seguridad y confianza	15
Escala	15
La ética del software	16
Código de ética del equipo de trabajo	18
Ciclo de vida de un software	20
Roles en el desarrollo de software	22
Los roles del equipo	23
Roles en el desarrollo de Software	23
Gestión del Proyecto	24
Introducción	24
Objetivos del proyecto	24
Restricciones	25
Análisis de riesgos	26
Estimaciones	27
Estimación del esfuerzo	28
Planificación	28
Ejecución	31
Seguimiento y Control	32
Cierre	32
Modelos de desarrollo de software	33
Modelo en cascada	33

Análisis y definición de requerimientos	34
Diseño del sistema y de software	34
Implementación y prueba de unidad	34
Integración y prueba de sistema	34
Operación y mantenimiento	34
Problemas del modelo de cascada	35
Modelo de desarrollo incremental	35
Modelo V	37
Objetivos del Modelo V	37
Modelo por prototipos o Modelo de desarrollo evolutivo	38
Modelo de reutilización de software	39
Proceso de desarrollo:	40
Modelo en Espiral de Boehm	42
Establecimiento de objetivos:	42
Valoración y reducción del riesgo:	43
Desarrollo y validación:	43
Planeación:	43
Modelo Iterativo-Incremental	44
Rational Unified Process (RUP)	44
Principios del RUP	44
Fases de desarrollo de RUP	45
Modelo de desarrollo Ágil	46
Metodologías ágiles	46
Extreme programming (XP)	48
PRÁCTICAS XP AMPLIAMENTE ADOPTADAS	48
SCRUM	50
Actividades	51
Beneficios	52
¿Cuándo emplear un modelo de desarrollo de software específico?	53
Ingeniería de requerimientos	55
Datos de entrada y datos de salida	56
Datos de entrada	57
Datos de salida	57
Requerimientos de usuario (DoRCU)	58
Pasos a seguir para el desarrollo de requerimientos del proyecto	59
Técnicas de recolección de requerimientos	61
Entrevistas	61
Desarrollo de prototipos	62

Ventajas	62
Observación	63
Reutilización de requerimientos	63
Etnografía	64
Escenarios	65
Metodologías de sistemas suaves	67
Safari de nuestro proyecto	67
Requerimientos del proyecto	71
Síntesis de requerimientos funcionales	71
Requisitos no funcionales	94
Diagramas de caso de uso	96
Diseño del proyecto	97
Patrones de diseño	100
Verificación y validación	110
Diseño de las pruebas	114
Mantenimiento	122
Conclusiones	125
Bibliografía	130
ANEXO 1. Requerimientos del proyecto	136
ANEXO 2. Diseño del sistema	217
DIAGRAMAS DE ACTIVIDADES	217
DIAGRAMAS DE SECUENCIA	251
DIAGRAMAS DE CLASES	285
ANEXO 3. Patrones de diseño	290
STRATEGY	290
OBSERVER	292
DECORATOR	295
FACTORY	297
FACTORY METHOD	299
COMMAND	302
ADAPTER	303
ANEXO 4. LINK a repositorio en Github del proyecto	305

**BUAP**

Hoja de control de cambios

Tipo de copia ¹	Versión del documento	Elaboradores	Descripción
T	1.0	Milcha Oropeza López Ramiro Vidal Lumbreras Erick Nuñez Grajales	Creación del documento incluyendo índice, hoja de control de cambios, importancia de la ingeniería de software y tipos de software.
T	1.1		Historia de la ingeniería de software Bibliografía
D	1.2	Milcha Oropeza López Ramiro Vidal Lumbreras Erick Nuñez Grajales Alhelí Moreno Gaytán Martín Antonio Paniagua Velázquez José Carlos Mejía Ramírez	Se incluyó información sobre la importancia de la ingeniería de software, su clasificación y mitos que existen sobre el software. Se anexó información a las conclusiones. Redacción y ortografía.
D	1.3	Martín Antonio Paniagua Velázquez Ramiro Vidal Lumbreras Milcha Sarai Oropeza López	Ciclo de vida de Software Ética del software
D	1.4	Martín Antonio Paniagua Velázquez Alhelí Moreno Gaytán Jose Carlos Mejia Ramirez Erick Nuñez Grajales	Código de ética del equipo de trabajo. Roles en el desarrollo de software Roles del equipo Referencias(Actualización) Conclusiones (Actualización)

¹ Tipo de copia

M = Copia maestra,

E = Email,

C = Copia controlada (papel),

D = Copia electrónica en medio digital,

T = TeamSite (Microsoft Teams)

Tipo de copia	Versión del documento	Elaboradores	Descripción
D	1.5	Milcha Sarai Oropeza López Erick Nuñez Grajales Martín Antonio Paniagua Velázquez Alhelí Moreno Gaytán Jose Carlos Mejia Ramirez Ramiro Vidal Lumbreras	Gestión del proyecto Estimación de tiempo Redacción, ortografía y formato. Conclusiones (Actualización) Referencias(Actualización)
D	1.6	Alhelí Moreno Gaytán Erick Nuñez Grajales Ramiro Vidal Lumbrera Martín Antonio Paniagua Velázquez Milcha Sarai Oropeza López	Modelos de desarrollo de software. ¿Cuándo usar un software específico? Conclusiones(Actualización) Referencias(Actualización)
D	1.7	Alhelí Moreno Gaytán Erick Nuñez Grajales Milcha Sarai Oropeza López Martín Antonio Paniagua Velázquez Ramiro Vidal Lumbreras	Modelos ágiles de desarrollo de software Redacción,ortografía y formato Conclusiones(Actualización) Referencias(Actualización)
D	1.8	Milcha Sarai Oropeza López Alhelí Moreno Gaytán Martín Antonio Paniagua Velázquez Erick Nuñez Grajales Jose Carlos Mejia Ramirez Ramiro Vidal Lumbreras	Ingeniería de Requerimientos Pasos a seguir en nuestro proyecto Conclusiones(Actualización) Referencias(Actualización) Redacción,ortografía y formato
D	1.9	Erick Nuñez Grajales Alhelí Moreno Gaytán Ramiro Vidal Lumbreras Martín Antonio Paniagua Velázquez Jose Carlos Mejia Ramirez Milcha Sarai Oropeza López	Técnicas de recolección de requerimientos Conclusiones(Actualización) Referencias(Actualización)
D	1.10	Alhelí Moreno Gaytán Martín Antonio Paniagua Velázquez Erick Nuñez Grajales Milcha Sara Oropeza López Ramiro Vidal Lumbreras Jose Carlos Mejia Ramirez	Síntesis de requerimientos del Proyecto Anexo 1 Conclusiones(Actualización) Referencias(Actualización)

D	1.11	Martín Antonio Paniagua Velázquez Alhelí Moreno Gaytán Jose Carlos Mejia Ramirez Erick Nuñez Grajales Milcha Sara Oropeza López Ramiro Vidal Lumbreras	Síntesis de requerimientos del Proyecto (Actualización) Anexo 1 (Actualización) Conclusiones(Actualización) Referencias(Actualización)
D	1.12	Jose Carlos Mejia Ramirez Erick Nuñez Grajales Milcha Sara Oropeza López Ramiro Vidal Lumbreras	Se añaden los Diagramas de casos de uso como complemento a la entrega anterior Diseño del proyecto: Síntesis Anexo 2.Diseño del sistema Anexo 3.LINK a repositorio en Github del proyecto Conclusiones(Actualización) Referencias(Actualización)
D	1.13	Martín Antonio Paniagua Velázquez Alhelí Moreno Gaytán Jose Carlos Mejia Ramirez Erick Nuñez Grajales Milcha Sara Oropeza López Ramiro Vidal Lumbreras	Diseño de proyecto: Síntesis (Actualización) Anexo 2.Diseño del sistema (Actualización) Conclusiones(Actualización) Referencias(Actualización)
D	1.14	Martín Antonio Paniagua Velázquez Alhelí Moreno Gaytán Jose Carlos Mejia Ramirez Erick Nuñez Grajales Milcha Sara Oropeza López Ramiro Vidal Lumbreras	Diseño de proyecto: Síntesis (Actualización) Patrones de Diseño parte 1 Anexo 3.Patrones de Diseño Anexo 4.Link a repositorio Conclusiones(Actualización) Referencias(Actualización)
D	1.15	Martín Antonio Paniagua Velázquez Alhelí Moreno Gaytán Jose Carlos Mejia Ramirez Erick Nuñez Grajales Milcha Sara Oropeza López Ramiro Vidal Lumbreras	Diseño de proyecto: Síntesis (Actualización) Patrones de Diseño parte 2 Anexo 3.Patrones de Diseño (Actualización) Anexo 4.Link a repositorio Conclusiones(Actualización) Referencias(Actualización)
D	1.16	Martín Antonio Paniagua Velázquez Alhelí Moreno Gaytán Jose Carlos Mejia Ramirez Milcha Sara Oropeza López	Diseño de proyecto: Síntesis (Actualización) Patrones de Diseño parte 3 Anexo 3.Patrones de Diseño (Actualización)

			Anexo 4.Link a repositorio Conclusiones(Actualización) Referencias(Actualización)
D	1.17	Martín Antonio Paniagua Velázquez Alhelí Moreno Gaytán Jose Carlos Mejia Ramirez Erick Nuñez Grajales Milcha Sara Oropeza López Ramiro Vidal Lumbreras	Verificación y validación Mantenimiento Conclusiones(Actualización) Referencias(Actualización)

Historia de la ingeniería de software

En 1968 surgió el concepto de ingeniería de software, tras una conferencia en Alemania patrocinada por la OTAN, de ahí el tema de crisis de software se vuelve el central (esta crisis tuvo lugar a finales de 1960 a mediados de 1980). En estos tiempos se empezó a crear hardware basado en circuitos integrados y gracias a ello la petición de nuevos sistemas y aplicaciones más complejos era aún mayor, tras dichas peticiones muchos de los proyectos no se terminaban a tiempo y con esto se sobrepasaba el presupuesto establecido, además de que carecían de funcionalidades que el cliente realmente requería.

Durante décadas, querer solucionar la crisis de software provocó que compañías e investigadores se vieran en la necesidad de crear nuevas y novedosas herramientas de software. Cada nueva tecnología o método que apareció entre 1970 y 1990 se trató como una “bala de plata” que se pensaba solucionaría esta crisis, hubo muchas empresas desarrolladoras de software que buscaron resolver esta crisis, sin embargo, hubo otras que dedujeron que no había una solución única ni efectiva ante la misma.



Mitos del software

Probablemente uno de los problemas más comunes al momento de realizar un proyecto grande, o con varios colaboradores, son los mitos que existen o que se pueden generar a través del desarrollo del software. Dentro de esta área se pueden identificar tres tipos de mitos:

Mitos de gestión: son los mitos que se generan dentro de la empresa y las decisiones que se toman para la realización del proyecto, como por ejemplo en la capacitación del personal, y es que se piensa que un libro y un curso es suficiente para que el personal esté capacitado y listo para realizar cualquier proceso de la empresa, cuando no es así, ya que empleados de nuestra rama deben de practicar mucho para mejorar y comprender los posibles errores que pueden surgir en el proceso. Otro es el de contratar más personal cuando las cosas salen mal con la finalidad de reparar los errores y terminar a tiempo; probablemente es una idea que no suena mal, el problema radica en que el nuevo personal no conoce el modo de trabajo, ni todos los detalles del proyecto en sí, por lo que el proceso de adaptación es largo, lo cual retrasa ralentiza el desarrollo y, como consecuencia, la entrega final.

Mitos del cliente: éstos generalmente se dan debido a que el usuario no está familiarizado con los procedimientos que se llevan a cabo para el programa solicitado, ni los costos que implica, por lo que algunos mitos que existen son el de cambiar requisitos del software en cualquier etapa del desarrollo porque el programa es flexible, los cual está lejos de la realidad, ya que hacer modificaciones en un programa diseñado para requisitos específicos toma tiempo y dinero, dependiendo de qué tan avanzado está.

Mitos de los desarrolladores: causados principalmente por la falta de experiencia en ese ámbito. Un ejemplo que se relaciona no sólo con el programador, sino también con el cliente es el pensar que cuando el programa está terminado ya no hay nada más por hacer; y es que la realización de guías para el correcto empleo del software y su mantenimiento, son cosas que el programador y el cliente llegan a considerar innecesarias por no entender la importancia de su implementación.

Este tipo de mitos en definitiva interfieren con un buen trabajo colaborativo y una buena relación de comunicación entre la empresa, el empleado y el cliente, lo cual puede tener repercusiones monetarias para la empresa, pérdida de clientes, entre otros. En el caso del desarrollador puede implicar una mala conexión con sus compañeros de equipo, incluso la pérdida de trabajo si no se logra esa sintonía adecuada; por lo tanto, tener en claro estos mitos desde el inicio de un proyecto es importante para tratar de no alimentarlos y restarles credibilidad con un desarrollo adecuado del proyecto.

La importancia de la ingeniería de software

Para entender lo importante que es aplicar ingeniería en el desarrollo del software hay que partir del concepto principal que envuelve el tema: el software.

¿Qué es el software? Se entiende como un conjunto o colección de reglas, programas, instrucciones, datos e información que permiten realizar tareas diversas en una computadora.



En la actualidad la mayoría de los países tienen una gran dependencia en los sistemas informáticos; son muchos los productos que incluyen en su composición una computadora y un software de control. Es por lo anterior que ante mayor sea la demanda del software en los diversos mercados, es indispensable que el desarrollo y mantenimiento de este sea costeable para un correcto funcionamiento de la economía nacional e internacional.

Debido a estas experiencias previas y las problemáticas derivadas de un mal manejo en el desarrollo del software es que se busca atender las nuevas exigencias a través de un correcto manejo de los procesos, incluyendo el diseño, análisis, implementación y mantenimiento, todo lo anterior a través de la ingeniería de software.

Según la definición aportada por Zelkowitz en 1978, la ingeniería de software “*es el estudio de los principios metodologías para el desarrollo y mantenimiento de sistemas software*”, por lo tanto, su importancia es inminente para un programador ya que le permitirá mejorar sus habilidades para la creación de software e incluso aprender a ser más hábil y eficaz, siguiendo reglas establecidas para lograr una mejor implementación y mantenimiento de los programas.

Es parte de nuestra tarea como ingenieros del área el desarrollar técnicas cada vez más completas y confiables para la creación de un software, siempre tomando en cuenta las necesidades y especificaciones que el cliente otorgue, es por ello que la ingeniería de software se vuelve una herramienta indispensable para el desarrollo correcto de nuestros proyectos y para obtener como resultado programas eficientes y que cubran las expectativas del usuario.

Clasificación del software

El software permite la ejecución de distintas tareas en una computadora, creando un puente de comunicación entre el usuario y el hardware. La clasificación del software se divide en tres grupos o tipos principales.

1. Software de aplicación

Son programas que fueron diseñados para facilitar tareas o trabajos específicos en la computadora para los usuarios.

Algunos ejemplos son:

- Word
- Excel
- Publisher

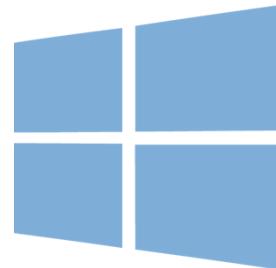


2. Software de sistema

Este permite interactuar con los componentes físicos de la computadora y sirve como base para que el usuario pueda controlarla.

Algunos ejemplos son:

- Linux
- Windows
- iOS



3. Software de programación

Son herramientas que los programadores usan para poder crear nuevas aplicaciones, éstas funcionan con un lenguaje específico de programación.

Algunos ejemplos son:

- C++
- Visual
- Java



Problemas generales que afectan al software

1. Heterogeneidad

Conforme pasa el tiempo los sistemas operan cada vez más como sistemas distribuidos por redes las cuales incluyen diferentes tipos de dispositivos móviles y computadoras.

2. Cambio empresarial y social

La sociedad y los negocios han estado cambiando rápidamente a medida que las economías emergentes avanzan al igual que las nuevas tecnologías necesitan cambiar su software existente y desarrollar un nuevo software rápidamente.

3. Seguridad y confianza

Al estar varios aspectos de nuestra vida entrelazados con el software, es primordial que podamos confiar en ese software.

4. Escala

El software por utilizar debe ser de alta calidad en escalas, desde los sistemas pequeños integrados en dispositivos o portátiles hasta sistemas que están basados en la nube que utiliza una comunidad global.



La ética del software

Los ingenieros de software deben de actuar de manera responsable y coherente con el interés social de manera que produzca el mejor resultado posible para el cliente y los usuarios o las personas que tengan que interactuar con el software.

El desarrollar software no sólo implica realizar una guía al usuario y otras especificaciones del software, sino que se debe garantizar al cliente de que su software es confiable, seguro y funcione a la perfección.

Una de las propuestas dadas es regularnos antes de que se nos regule. Esto quiere decir que dentro de nuestro equipo de trabajo y de manera personal tenemos que buscar la forma de establecer reglas, ver nuestras opciones y tratar a fondo las cuestiones que involucran la ética dentro de un proyecto; analizar si de alguna forma nuestro software puede dañar a alguien o causar mal de manera colectiva, y es que si nosotros regulamos nuestras tareas y estas son transparentes desde un inicio, evitamos que después otro tipo de empresas u organizaciones que no conocen cómo trabaja nuestro software, nos impongan regulaciones que afecten el funcionamiento del mismo.

Según lo expuesto por Graterol (2018), existen dos tipos de impactos:

- Impactos accidentales: Este tipo de impacto está relacionado con errores que, como su nombre dice, no tienen intención alguna de cometerse, vienen de la mano con algoritmos que no están bien desarrollados o no tienen suficiente madurez para ser utilizados; incluso cuando un algoritmo está incompleto. En este caso se puede decir que la ética no entra en juego por completo, pero sí malas prácticas que llevan a errores en el sistema, que, aunque no sean intencionales, pueden perjudicar al usuario final de diferentes maneras.
- Impactos intencionales: Esta clasificación de impactos se relaciona con todas las decisiones que fueron previamente estudiadas y que pese al daño que podrían

generar, se concretaron y desarrollaron. Normalmente se ven implicados los altos mandos que conocen cómo funciona un software y todos sus alcances tanto negativos como positivos, y por motivos diversos permiten que salga a luz; existen diversos ejemplos donde diferentes tipos de software han causado daño a algún sector de una población y tienen que ser evaluados para analizar su funcionamiento. La forma de mejorar en este aspecto es hacer un estudio exhaustivo de las normas éticas que se aplican en la empresa, y en cuáles pueden mejorar.

Cualquier regulación debe venir de nosotros, ya que hay varios problemas que si se permite cambiar por gobiernos más poderosos podría causar daños a futuro y aumentar el problema.

Como propuesta complementaria al tipo de impacto accidental, se proponen algunas prácticas para evitar que se generen desde un principio, como por ejemplo mejorar las prácticas de ingeniería aplicadas en la empresa, esto quiere decir la forma de trabajo utilizada, la mejora en seguridad informática, el tipo de pruebas que se realizan en el software, la calidad del software; e incluso aspectos más relacionados a la persona misma, tales como la disciplina, el trabajo en equipo y el profesionalismo.

A nivel industria, actualización continua de los códigos de ética para que se adapten a las prácticas de la ingeniería de software, tener asociaciones gremiales donde se promueva el desarrollo, protección y resguardo a ingenieros, técnicos, desarrolladores y de las normas establecidas por sus integrantes.

Tomando en cuenta la importancia de la ética dentro de cualquier empresa o equipo de trabajo, se considera de suma importancia analizar de qué manera se están aplicando estos lineamientos de la ética deontológica en el software utilizado, y en la forma de trabajo de cada integrante del proyecto; todo esto con la finalidad de poder

proporcionar un software libre de malas técnicas y que no cause ningún daño a la sociedad.

Código de ética del equipo de trabajo

El siguiente código, basado en el “Código de Ética y Conducta Profesional de ACM” (ACM Code of Ethics and Professional Conduct, s.f), se diseña para guiar la conducta ética de todos los integrantes del equipo, garantizando de esta forma buenas prácticas en el desarrollo de este y futuros proyectos. El Código incluye principios que buscan el bien personal y social, y será un punto de partida para la toma de decisiones éticas en nuestro desempeño colaborativo.

- 1. Contribuir al bien social y humano.** Tenemos la obligación de utilizar nuestras habilidades, individual y colectivamente, en beneficio de la sociedad, de sus integrantes y del entorno que les rodea.

Tenemos como objetivo esencial minimizar las consecuencias negativas en el desarrollo de software, como las amenazas a la seguridad, la salud y la privacidad de datos.

- 2. Evitar el daño.** Tomamos como “daño” todo perjuicio que pueda resultar de nuestro trabajo, incluyendo el daño moral y físico.

Se entiende que hay acciones bienintencionadas que pueden provocar perjuicios, por lo cual los responsables de causar un daño involuntario deberán deshacer o mitigar este daño tanto como sea posible.

Para evitar causar daños, se evaluarán minuciosamente los impactos de nuestras decisiones y se buscarán siempre las buenas prácticas en cualquier tarea que se desenvuelva.

Si algún integrante detecta en cualquier etapa del desarrollo que se puede causar algún daño, tendrá la obligación de informar al equipo después de haber evaluado cuidadosamente la situación.

3. **Ser honesto y confiable.** El valor de la honestidad es básico para crear un ambiente de confianza. Como profesionales debemos brindar información fidedigna y completa sobre nuestras competencias y habilidades individuales; así como detallar los alcances y limitaciones de nuestro proyecto.

Además, tenemos la responsabilidad de cumplir con nuestros compromisos individuales y ser sinceros cuando alguna situación en particular pueda generar conflictos de interés.
4. **Ser justo y tolerante.** Debemos fomentar la participación justa de todos los integrantes, esto en cada una de las etapas del proyecto. No se permitirá la discriminación prejuiciosa, intimidación u ofensas.

Buscaremos que nuestras prácticas sean incluyentes y accesibles, evitando así ocasionar una discriminación injusta.
5. **Ser respetuoso.** El respeto deberá aplicarse dentro y fuera del entorno de desarrollo. Lo anterior implica respetar las tareas que sean asignadas a cada miembro del equipo, sin dejar de fortalecer la comunicación y colaboración continua.
6. **Cuidar la privacidad y confidencialidad.** Tenemos la responsabilidad de respetar la privacidad y confidencialidad de los datos que se nos proporcionen para el desarrollo del proyecto.

Debemos familiarizarnos con los términos de privacidad para entender los derechos y responsabilidades que trae consigo la recopilación y el uso de datos confidenciales.

En caso de tener recopilada información personal, ésta deberá ser tratada según las políticas establecidas y jamás utilizada con otros fines sin un consentimiento anticipado.
7. **Ser responsable.** Debemos ser responsables de las tareas que nos corresponde atender individualmente y en equipo.

Como profesionales tenemos que hacer un esfuerzo constante por conservar una alta calidad en los procesos y productos en los que trabajamos, además de mantener estándares altos en nuestra conducta y práctica ética.

- 8. Defender, promover y respetar los principios del Código.** Todos los integrantes del equipo debemos cumplir los principios expuestos y contribuir para su constante mejora. En caso de reconocer incumplimientos del Código se tomarán medidas para resolver los problemas éticos que se identifiquen.

Ciclo de vida de un software

El ciclo de vida de un software se basa en el desarrollo de este desde el inicio hasta su final cuando ya es reemplazado o retirado, cuyo propósito es que pueda cumplir con los requisitos previamente establecidos asegurándose que los métodos de desarrollo de este sean los adecuados.

Los pasos que siguen son:



- 1. Estudio de factibilidad.**
- 2. Análisis (De requerimientos).**

El analista debe transformar los requisitos del usuario en requisitos de software.

3. Diseño.

Los diseñadores son los encargados de modelar el diseño del sistema, crear prototipos y generar el documento de diseño.

3.1 Creación de prototipos.

3.2 Implementación.

Esta parte del proceso le compete a los programadores quienes deben convertir las especificaciones del sistema en código.

4. Validación y pruebas.

Por su parte a los testers en esta fase les corresponde probar el sistema tratando de encontrar fallos que puedan ser notificados a los programadores para ser corregidos.

5. Operación y mantenimiento.

En la fase final lo que se busca es mantener el software fresco, con capacidad de adaptarse para adoptar nuevas funciones o modificar las existentes.

Este ciclo permite que la detección de errores se pueda verificar, a través de las distintas fases que lo conforman, de una manera más rápida y no generando costos extras, permitiendo a los desarrolladores enfocarse en la calidad del software y cumpliendo con las fechas que se hayan establecido.

Algo con lo que concuerdan todas las maneras de representar el ciclo de vida del software es que es un trabajo interminable, siempre las necesidades del cliente van a cambiar y a medida que cambien se requerirá el software actualizado y mucho más importante con un mantenimiento constante, previniendo futuros problemas y buscando que sea eficiente y aceptable para el usuario.

En caso de que se tuviera que retirar el software y reemplazarse por uno nuevo, podría evaluarse la posibilidad de extender su ciclo de vida reutilizándolo, en este proceso se rescatan partes que se puedan emplear en el nuevo software, consiguiendo optimizar tiempo para las fechas de entrega, aprovechando lo que ya se había trabajado y evitando la redundancia.



Roles en el desarrollo de software

- Analista

El analista se encarga de la especificación y análisis de requerimientos para el software.



- Diseñador

El diseñador se encarga de crear el diseño, prototipos, y documentos que mejor se adecuen al software.

- Programador

El programador se encarga de la creación del código simple cubriendo todas las necesidades del cliente.



- Tester

El tester se encarga de la detección y eliminación de errores que se pudieran encontrar en el software.

- Asegurador de calidad

El asegurador de calidad revisa continuamente los avances del software con el objetivo de que cumpla con la calidad.

- Administrador de configuración

El administrador de configuración ayuda a reducir problemas coordinando a las demás áreas del proyecto.

- Ingeniero de validación y verificación

El ingeniero de validación y verificación evalúa que una vez terminado el software se asegura que esté libre de fallas y cumpla con los requisitos, de todos modos, en determinadas fases checa que cumpla con los requisitos de la fase anterior.

- Documentador

El documentador tiene como objetivo principal mantener la información generada, en otras palabras, mantener la información al día.

- Ingeniero de Manutención

El ingeniero de manutención se encarga de modernizar y modificar el software, mientras informa a el equipo de desarrollo de errores que se puedan encontrar.



Los roles del equipo

Una vez identificados y descritos los roles en el ciclo de vida de software, la asignación en nuestro equipo es la siguiente:

Roles en el desarrollo de Software	
Nombre(s)	Rol a desempeñar
Milcha Sarai Oropeza López	Analista
Ramiro Vidal Lumbreras	Diseñador
Martín Antonio Paniagua Velázquez	Programador
Erick Nuñez Grajales	Tester
Alhelí Moreno Gaytán	Asegurador de calidad
Martín Antonio Paniagua Velázquez	Administrador de configuración
Jose Carlos Mejia Ramirez	Ingeniero de validación y verificación
Erick Nuñez Grajales	Documentador
Alhelí Moreno Gaytán	Ingeniero de manutención

Gestión del Proyecto

Introducción

La BUAP nos ha encomendado la tarea de realizar una plataforma en línea, capaz de integrar comunidades BUAP, en las que los alumnos puedan tener un acceso sencillo a sus clases, se facilite el trabajo en equipo a distancia, comunicación entre docentes y estudiantes, aprovechar la tecnología educativa al máximo y más importante, que los estudiantes al finalizar el semestre/cuatrimestre en línea, puedan reintegrarse sin dificultades a sus clases presenciales en primavera 2021.



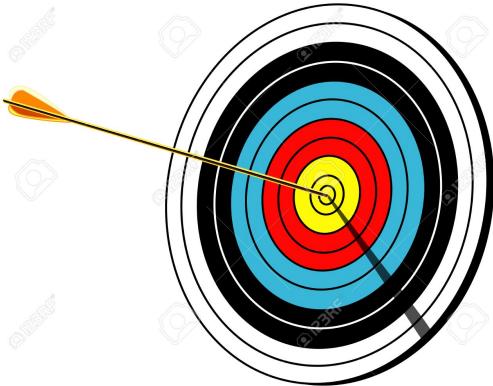
Objetivos del proyecto

El propósito del proyecto del desarrollo de software para la creación de una plataforma educativa en una iniciativa de la Dirección de Innovación y Transferencia de Conocimiento(DITCO) de la comunidad BUAP es la vinculación entre estudiantes y profesores para la mejora y el crecimiento del aprendizaje virtual.

Para lograr adecuadamente el objetivo de nuestro proyecto es importante que se cumplan las siguientes características:

- Facilitar procesos de aprendizaje en épocas de pandemia entre estudiantes y profesores con herramientas para una mejor comunicación como lo son la creación de foros, videoconferencias, chats y correos electrónicos.

- Trabajos en equipo entre estudiantes de cualquier parte de la república y/o extranjeros.



- Flexibilidad y accesibilidad, con los estudiantes a los que se les dificulta tomar sus clases debido a que cuentan con trabajos, por lo que una grabación de calidad los beneficiaría totalmente.
- Facilitar la colaboración entre docentes.
- Herramientas de ayuda para los estudiantes como (autoevaluaciones, zonas de trabajo, contenidos gratuitamente educativos y perfiles que permitan el acceso a plataformas que brindan la información necesaria para su aprendizaje).

Restricciones

Al tratarse de un proyecto de una institución académica debe de contar con ciertas restricciones tales como:

- El proyecto deberá de estar finalizado el día 7 de Diciembre del presente año.
- El proyecto indica que solamente estudiantes de la comunidad BUAP podrán tener acceso a este, con perfiles únicos.
- Para la validez de los estudios dentro de esta comunidad únicamente se dará acceso a las bases de datos que la comunidad BUAP aporte.



Análisis de riesgos

En esta etapa se evalúan los problemas que podemos llegar a encontrar y al mismo tiempo se clasifican de acuerdo a la probabilidad en cuanto ocurra y sea grande, mediana o chica y si puede llegar a tener una consecuencia catastrófica, seria o insignificante y las soluciones que se les puede proveer. En todo caso tenemos que tener ya hechas algunas soluciones por si se nos puede llegar a presentar alguno de los siguientes problemas:

Riesgo	Probabilidad	Consecuencia	Solución
No tener el software a tiempo	Baja	Catastrófica	Pedir más tiempo para la elaboración del software brindando la justificación adecuada.
Problemas entre el equipo	Baja	Seria	Intentar que los afectados en este tema tengan una conversación para poder llegar a un punto en común y no perjudicar el proyecto.
No tener el documento de proyecto a tiempo	Media	Seria	Dialogar con el profesor con tiempo de notificación haciéndole saber que el documento no estará para la fecha esperada y darle una justificación.
Si varios miembros del equipo tienen complicaciones y no pueden cumplir sus partes del proyecto	Media	Seria	El resto de los compañeros del equipo apoyaran a los miembros que no pueden trabajar con sus partes correspondientes.

Estimaciones

Usualmente se suele dejar un apartado con las aproximaciones económicas y de tiempo con un balance más o menos detallado de ingresos y egresos implicados dentro del proyecto, en este caso se va a hacer una estimación de esfuerzo orientado al problema según el modelo COCOMO.

Funciones	optimista (kloc)	promedio (kloc)	pesimista (kloc)	Kloc esperado	costo por línea	líneas por semana	costo función	meses persona

Funciones: las principales actividades que hará el proyecto.

Optimista: El valor mínimo para la creación de código de cada función.

Promedio: El valor medio para la creación de código de cada función.

Optimista: El valor máximo para la creación de código de cada función.

Kloc esperado: Este se calcula en base a la siguiente fórmula, $(\text{optimista} + 4 * \text{promedio} + \text{pesimista}) / 6$

Costo por línea: Es el costo aproximado de una línea de código.

Líneas por semana: Es el valor aproximado de las líneas de código por semana.

Costo función: kloc esperado * costo por línea.

Meses persona: Kloc esperado / líneas por mes.

Kloc = miles de líneas de código.

NOTA: Esta estimación de costos solo se dejó indicada ya que aún se desconoce la cantidad aproximada de líneas de código que se emplearán en cada función del software.

Estimación del esfuerzo

Partiremos de operaciones aritméticas simples para averiguar el tiempo máximo para cada fase, tomando en cuenta que no todas las actividades se deben de cumplir en un determinado tiempo, lo analizaremos de la siguiente manera:

Fecha de trabajo efectivo: Del 17 de agosto de 2020 al 2 de diciembre de 2020 = 105 días

*Días de trabajo efectivo: 16 semanas * 4 días de trabajo por semana = 64 días*

Horas diarias de trabajo efectivo: 2 horas aproximadamente.

Planificación

Esta planificación se basó en la estimación de esfuerzos y según el calendario del módulo para la realización de las actividades, los circulos amarillos son los días de la semana que se han de trabajar, es decir de lunes a viernes.



Planificación por semana del proyecto

TAREA	SEMANA 5	SEMANA 6	SEMANA 7	SEMANA 8
Modelo de desarrollo de software parte 2	● ● ● ● ●	● ● ● ● ● ● ● ● ● ● ● ● ●		
Proceso de ingeniería de requerimientos	● ● ● ● ●	● ● ● ● ●	● ● ● ● ● ● ● ● ● ● ● ●	
Técnicas de ingeniería de requerimientos	● ● ● ● ● ● ● ●	● ● ● ● ● ● ● ●	● ● ● ● ● ● ● ●	
Requerimientos del proyecto	● ● ● ● ● ● ● ●	● ● ● ● ● ● ● ●	● ● ● ● ● ● ● ●	● ● ● ● ● ● ● ●

Planificación por semana del proyecto

TAREA	SEMANA 9	SEMANA 10	SEMANA 11	SEMANA 12
Requerimientos del proyecto	● ● ● ● ●	● ● ● ● ● ● ● ● ● ● ● ●		
Diseño del proyecto parte 1	● ● ● ● ●	● ● ● ● ● ● ● ● ● ● ● ●		
Diseño del proyecto parte 2	● ● ● ● ● ● ● ●	● ● ● ● ● ● ● ●	● ● ● ● ● ● ● ●	
Patrones del diseño parte 1	● ● ● ● ● ● ● ●	● ● ● ● ● ● ● ●	● ● ● ● ● ● ● ●	

Planificación por semana del proyecto

TAREA	SEMANA 13	SEMANA 14	SEMANA 15	SEMANA 16
Patrones del diseño parte 2	● ● ● ● ●	● ● ● ● ● ● ● ● ● ● ● ● ● ●	● ● ● ● ● ● ● ● ● ● ● ● ● ●	● ● ● ● ● ● ● ● ● ● ● ● ● ●
Patrones del diseño parte 3	● ● ● ● ●	● ● ● ● ● ● ● ● ● ● ● ● ● ●	● ● ● ● ● ● ● ● ● ● ● ● ● ●	● ● ● ● ● ● ● ● ● ● ● ● ● ●
Pruebas del proyecto	● ● ● ● ●	● ● ● ● ● ● ● ● ● ● ● ● ● ●	● ● ● ● ● ● ● ● ● ● ● ● ● ●	● ● ● ● ● ● ● ● ● ● ● ● ● ●
Pruebas y mantenimientos del proyecto	● ● ● ● ●	● ● ● ● ● ● ● ● ● ● ● ● ● ●	● ● ● ● ● ● ● ● ● ● ● ● ● ●	● ● ● ● ● ● ● ● ● ● ● ● ● ●

Planificación por semana del proyecto

TAREA	SEMANA 17
Presentación del proyecto	● ● ● ● ●

El desarrollo de este proyecto se realizará en distintas fases, tal y como lo muestra la siguiente tabla, donde los hitos serán las actividades o artefactos que se pueden entregar determinando los objetivos de cada una de estas fases, el fin de estos hitos serán definidos por la última semana de cada fase .

Fase	Semanas	Hitos
Inicio	5	Desarrollo y comprensión de las bases que se necesitan para la elaboración del proyecto.
Elaboración	4	Análisis de los requerimientos de software
Construcción	3	Extensión de los análisis, casos de uso, clases y secuencias
Transición	4	Creación y patrones de diseño del proyecto
Mantenimiento	2	Corrección de errores detectables de programación y lógica, terminando en la última semana con la entrega de este.

Ejecución

En esta fase comienza al cien por ciento la elaboración del software que se va a utilizar con el lenguaje de programación más adecuado, en este caso el lenguaje a tratar será HTML, ya que es la herramienta que mejor se adapta para esta situación y la cual nos permitirá cumplir con los objetivos de la BUAP con mayor facilidad.

En esta etapa ya todas las áreas se coordinan de la mejor manera posible para poder facilitar la integración del proyecto al final y poder cumplir con las fechas establecidas con la institución.



De igual manera, a lo largo de esta etapa se intentará encontrar posibles errores que dificulten el funcionamiento del software y de ser así buscar la más eficiente y eficaz forma de corregirlos para que estos no presenten mayores problemas en el futuro para la universidad, estudiantes o docentes que las utilicen. Por lo que se considera que esta etapa está relacionada con la etapa de seguimiento y control.

Seguimiento y Control



Como se mencionó en la etapa pasada, esta está muy relacionada con ejecución ya que, tan pronto se avanza en la codificación y en la integración de las distintas áreas a trabajar así mismo se realizan pruebas al software checando que este no presente fallas, ya que si dejamos las pruebas al final será mucho más difícil encontrar la raíz del problema y mucho más complicado solucionarlo. Siguiendo esta idea de trabajo, garantizamos que la calidad del software sea elevada y mantenga el proyecto en marcha. Por lo cual se considera que iremos avanzando de acuerdo a la planificación ya hecha, checando constantemente nuestro progreso para verificar que vayamos acorde a él, y evitar que nuestro rendimiento planteado al proyecto baje.

Cierre

Esta etapa es la más corta ya que entregaremos a la BUAP el proyecto terminado pero no la hace menos importante por cerrar el proyecto sino que nos dará retroalimentación nuevas formas de trabajar que podremos aplicar en el siguiente proyecto en el que trabajaremos.



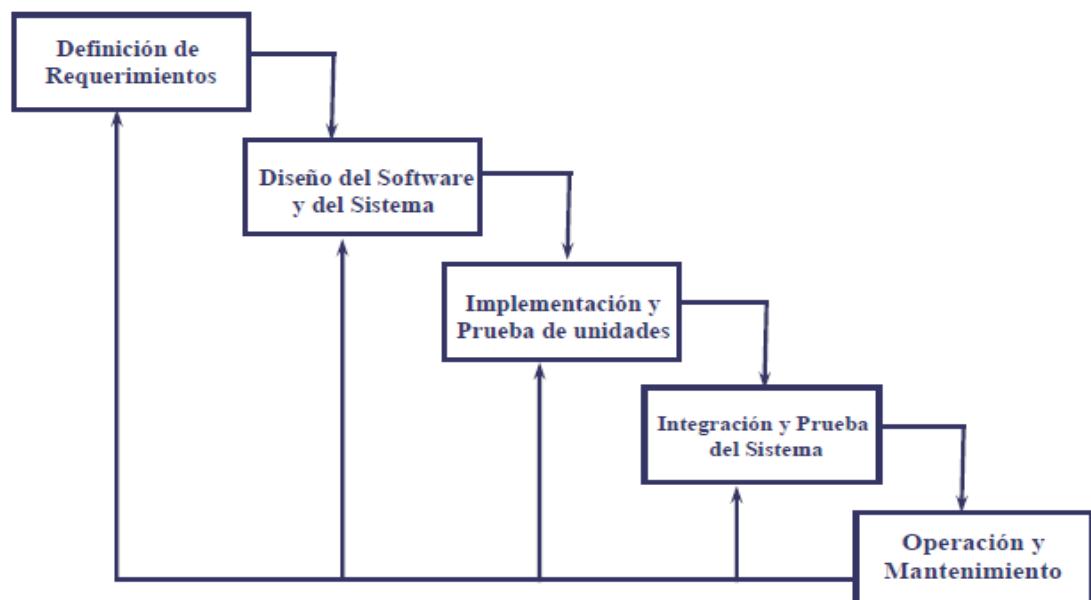
Modelos de desarrollo de software

Los modelos de desarrollo de software ofrecen un marco de trabajo usado para controlar el proceso de desarrollo, puede ser modificado y adaptado de acuerdo con las necesidades del software.

Modelo en cascada

Este fue el primer modelo que surgió para el proceso de desarrollo de software, el cual toma todas las actividades fundamentales como especificación, desarrollo, validación y evolución, y los representa como fases separadas del proceso como especificación de requerimientos, diseño de software, implementación, pruebas, etc...

A este proceso se le conoce como “modelo en cascada” por la transición entre una fase y otra como se puede notar en la siguiente imagen.



Análisis y definición de requerimientos

En esta etapa se consulta con los usuarios/clientes los servicios, restricciones, y metas que van a querer en el software, una vez tomados en cuenta todos los datos del usuario, se definen y utilizan como una especificación del sistema.

Diseño del sistema y de software

Sirve para formular una solución específica con base en los servicios, restricciones, y metas que se establecieron en la etapa anterior. Se establece una arquitectura de sistema global, centrándose en componentes como interfaces, entornos de trabajo, etc...

Implementación y prueba de unidad

A lo largo de esta etapa, usando la arquitectura de software hecha en la etapa pasada, se desarrolla la programación del software, la búsqueda de errores y las pruebas unitarias, las pruebas unitarias consisten en verificar que cada unidad cumpla con su especificación. Esta fase da como resultado un “producto beta”.

Integración y prueba de sistema

El producto beta desarrollado en la etapa pasada se integra y prueba como un sistema completo, para asegurarse que cumplan con los requisitos del usuario. Después de probarlo y si cumple, se libera el sistema de software al cliente.

Operación y mantenimiento

Esta es la etapa más larga del modelo de cascada, donde el sistema se instala y se pone en práctica. El mantenimiento incluye darle seguimiento a los errores que se presenten y que no se habían detectado en otras etapas.

Problemas del modelo de cascada

La división que tiene el proyecto en sus distintas etapas lo hace inflexible, esto hace que se dificulte la respuesta al hacer cambios de los requisitos del cliente

- Por esta razón este modelo solo es apropiado cuando los requisitos están bien comprendidos.
- Los cambios que se hagan estarán limitados mientras se hace el proceso de diseño.
- Los sistemas que tienen requisitos estables son pocos

Este modelo en cascada es utilizado principalmente en grandes proyectos de ingeniería de sistemas en donde este es desarrollado en varios sitios.

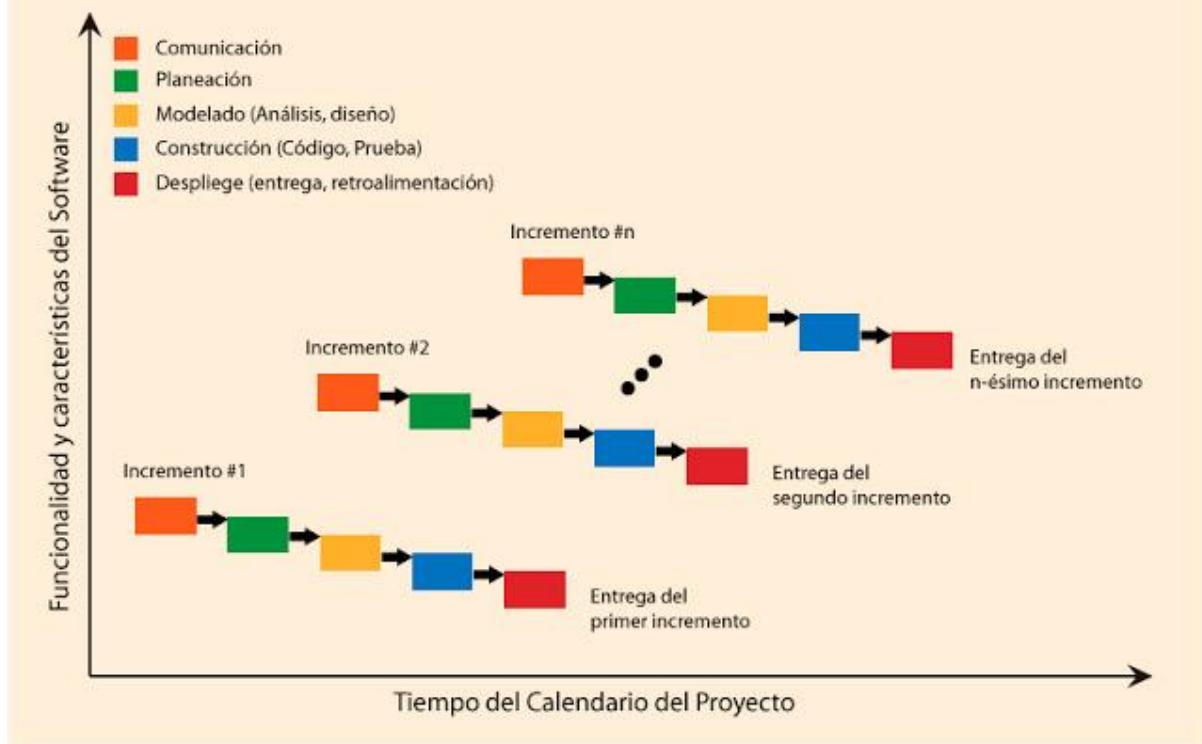
- Debido a esas circunstancias, el proceso hecho naturalmente por el modelo de cascada ayuda en la coordinación del trabajo.

Modelo de desarrollo incremental

Éste combina elementos del modelo en cascada con la implementación de construcción de prototipos, este modelo va incrementando las funcionalidades del programa, marcando sus progresos conforme el calendario. Cada una de estas secuencias lineales produce un incremento en el software.

En este es importante que se definan la cantidad de incrementos que se requieren para crear el software, estos deben de estar organizados tomando como base las funciones más importantes, minimizando la cantidad de cambios.

Modelo Incremental

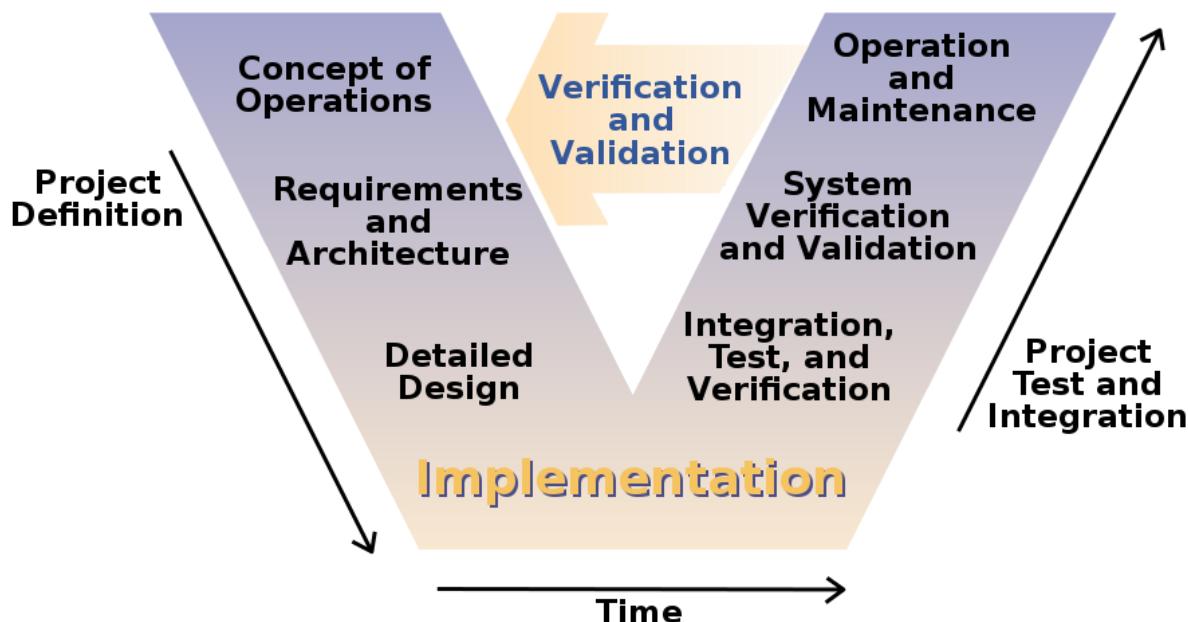


Como todo modelo este cuenta con ventajas y desventajas que se especificarán en la siguiente tabla:

Ventajas	Desventajas
Genera software operativo de forma rápida	Cada fase es inflexible y no se sobreponen sobre otras fases.
Cómo es flexible reduce el coste en el cambio de alcance y requisitos	Problemas en la arquitectura del sistema debido a que no todos los requisitos se llegan a juntar
Es más fácil ir probando y depurando en interacciones pequeñas	Se necesitan entregables regularmente para así poder medir el progreso
La gestión de riesgos es más fácil	Si los sistemas son desarrollados rápidamente, no será rentable producir documentos que reflejen todas las versiones del sistema
Las interacciones se vuelven hitos gestionados más fácilmente	El procesos no es visible

Modelo V

En este modelo se describen las actividades y resultados que desean obtenerse durante el desarrollo del producto. Este modelo toma la figura de V como se muestra a continuación :



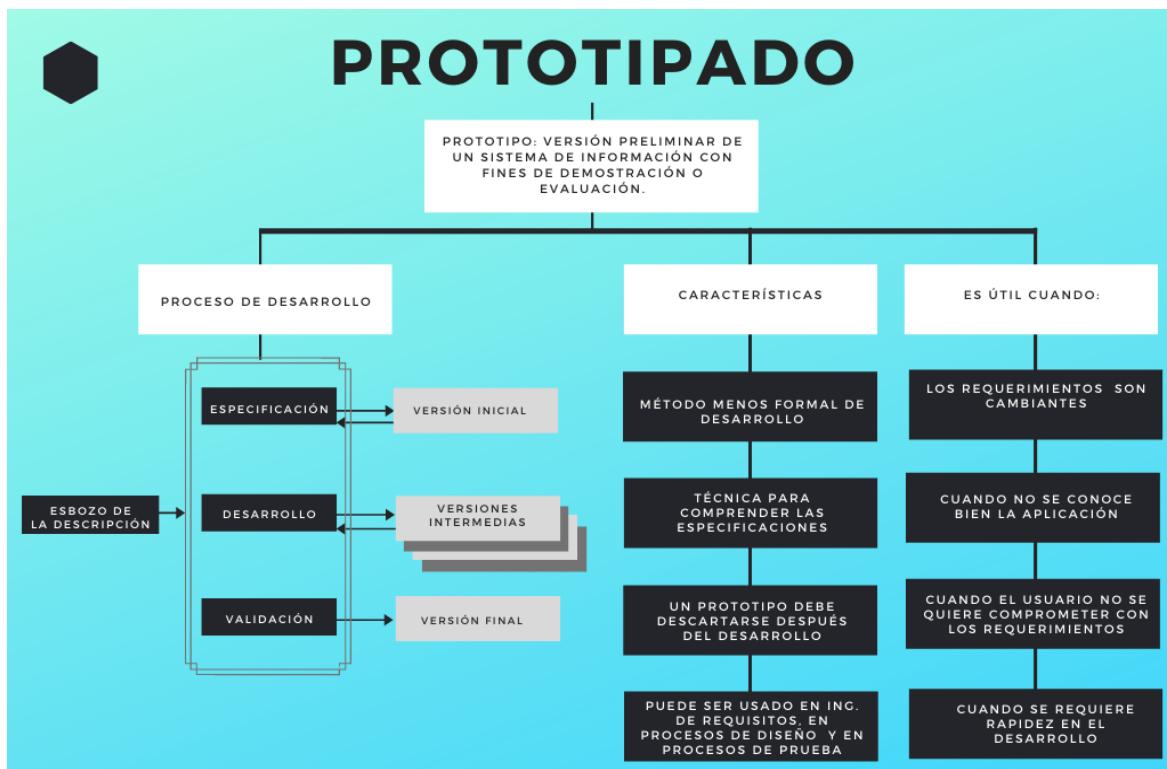
De lado izquierdo de la V se detallan las necesidades, especificaciones que necesitará el sistema, este conjunto de pasos en el diagrama llevan el nombre de “Project Definition” o “Definición del proyecto”. De lado derecho de la V podemos ver que es la integración del software, pruebas, verificación, operación y mantenimiento, este conjunto lleva el nombre de “Project Test and Integration” o “Prueba del proyecto e Integración”. A este modelo se le conoce como V por “Verificación y Validación”.

Objetivos del Modelo V

- **Reducir gastos del proyecto**
 - El desarrollo, producción, operación, y mantenimiento puede ser calculado y controlado para no exceder el presupuesto.

- Mejorar y asegurar la calidad
 - Al ser un proceso estándar, está seguro de estar completo y de proporcionarle al usuario la calidad que deseé. Este objetivo puede ser definido desde un principio.
- Mejora de comunicación entre cliente - desarrollador
 - En la etapa de definición de proyecto es vital la comunicación para confirmar todos los servicios, requisitos que deberá de cumplir el software.
- Minimización de riesgos
 - Este objetivo permite detectar desde un principio los riesgos que se puedan producir a lo largo del proyecto y el cómo evitarlos.

Modelo por prototipos o Modelo de desarrollo evolutivo



Este modelo es útil cuando el cliente conoce los objetivos generales para el software pero no identifica los requisitos detallados de entrada, procesamiento o salida.

VENTAJAS

- Se reduce el costo de adaptarse a los requisitos cambiantes del cliente.
- Retroalimentación constante de los clientes sobre el trabajo de desarrollo que se ha realizado.
- Es posible una entrega e implementación más rápida de software útil para el cliente.
- Reduce la posibilidad de que el producto final no coincida con las necesidades reales de los clientes.

DESVENTAJAS

- Se corren más riesgos al desarrollar antes de entender por completo el problema.
- Puede existir confusión del cliente entre los prototipos y el producto final.
- Los cambios regulares tienden a corromper la estructura del sistema. La incorporación de más cambios de software se vuelve cada vez más difícil y costosa.

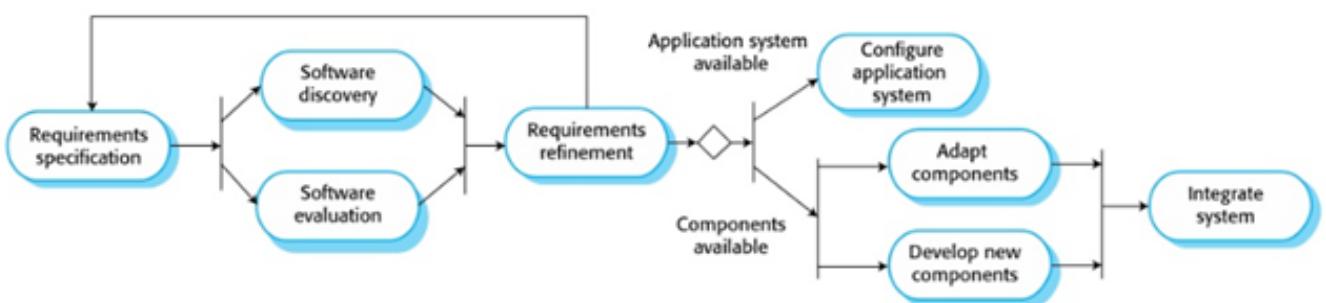
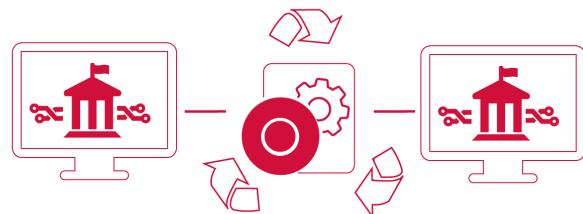
Modelo de reutilización de software

La reutilización de software es crear sistemas a partir de un software existente en lugar de desarrollarlo desde el comienzo. La reutilización se basa en la existencia de componentes reutilizables, que se integran en el sistema.

El equipo del proyecto busca diseños o códigos similares para modificarlos a lo requerido e incorporarlos en el sistema. El enfoque orientado a reutilización se compone de un gran número de componentes de software reutilizable, se estima que entre el 40% y 60% del código fuente de un software es reutilizable en otro similar.

Proceso de desarrollo:

- Definición de requerimientos
- Análisis de componentes
- Modificación de requerimientos
- Diseño de sistemas con reutilización
- Desarrollo e integración
- Validación del sistema



Definición de requerimientos: Parte donde se establece de manera preliminar a la fase de especificación los requerimientos y restricciones del sistema.

Análisis de componentes: Se buscan componentes para implementar la especificación de requerimientos y que funcionen correctamente.

Modificación de requerimientos: Los requerimientos se modifican para reflejar los componentes disponibles; si eso no es posible, se buscan soluciones alternativas

Diseño de sistemas con reutilización: Se diseña o se reutiliza un marco de trabajo para el sistema, tomando en cuenta los componentes disponibles; si no hay componentes adecuados, se diseñan otros nuevos.

Desarrollo e integración: Los componentes disponibles se compran, los componentes no disponibles se desarrollan y todos los componentes y los sistemas se integran.

Validación del sistema: Se verifica que el sistema de software producido cumpla con las especificaciones y que logre su cometido.

Ventajas

- ★ Reduce la cantidad de software a desarrollarse.
- ★ Reduce los costos y los riesgos.
- ★ El proceso es más rápido.
- ★ Incrementar la productividad.
- ★ No tener que reinventar las soluciones.
- ★ Facilitar la compartición de productos del ciclo de vida.



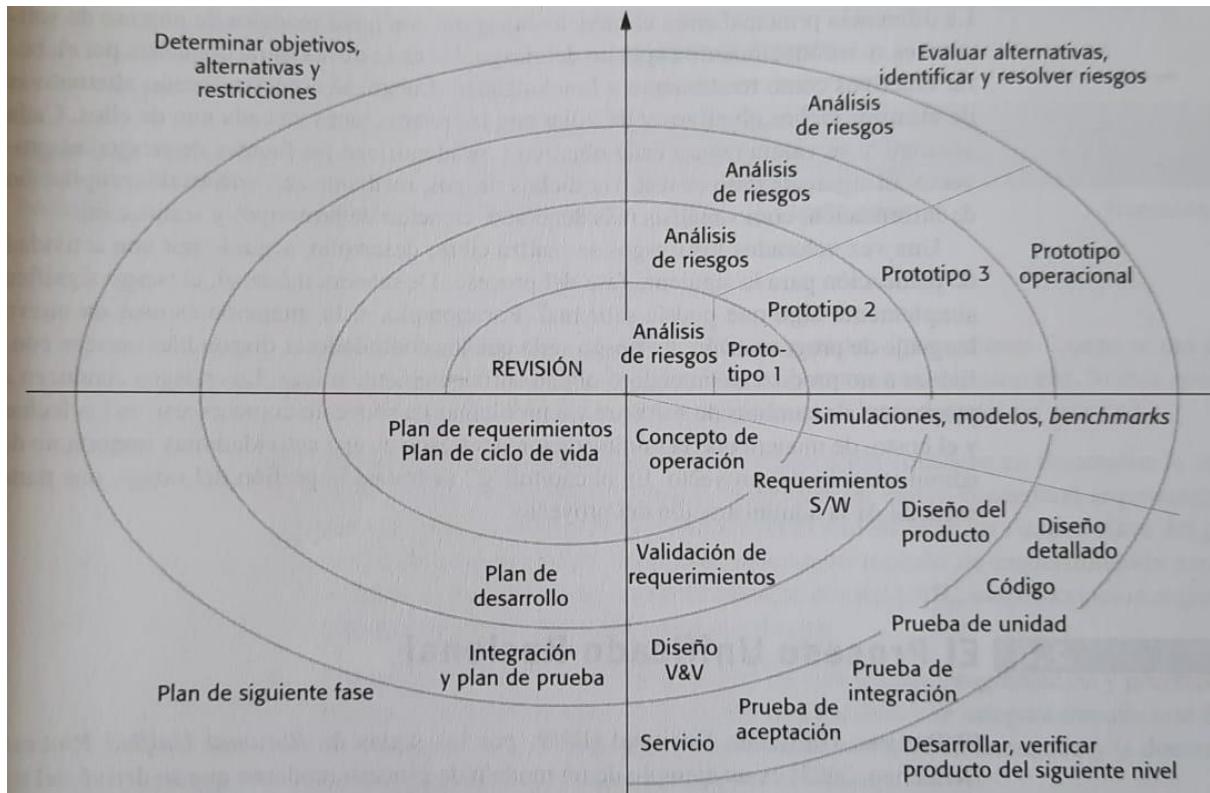
Desventajas

- Peligro de obtener un sistema que no cumple las necesidades reales de los usuarios.
- Necesidad de invertir antes de obtener resultados.
- Carencia de métodos adecuados.
- Convencer a los “mánager”.
- Dificultad para integrar componentes al sistema.



Modelo en Espiral de Boehm

En 1988 Boehm propuso un marco del proceso de software dirigido por el siguiente modelo :

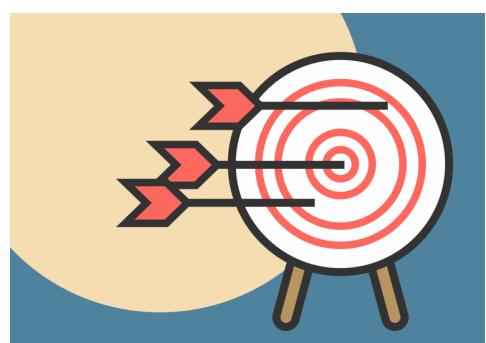


Como se puede ver en la imagen anterior, la representación de este modelo toma forma de un espiral en lugar de una secuencia de actividades con retroceso de una actividad a otra. Cada ciclo en la espiral representa una fase del proceso de software.

El espiral de Boehm se divide en cuatro sectores :

1. Establecimiento de objetivos:

En este sector se identifican los objetivos específicos para el proyecto, las restricciones durante el proceso y el



producto, trazando un plan de gestión detallado, riesgos que puedan ocurrir a lo largo, estrategias alternativas para los problemas que se puedan presentar.

2. Valoración y reducción del riesgo:



Por cada riesgo que se haya encontrado en el sector de “Establecimiento de objetivos” se realiza un análisis minucioso con el objetivo de reducirlos, si no es que de eliminarlos.

3. Desarrollo y validación:

En este sector podemos decidir qué modelo utilizaremos para nuestro proyecto, si lo que domina en el proyecto son los riesgos en la interfaz de usuario, entonces nos convendría utilizar el modelo de “Creación de prototipos desechables”, si el principal riesgo es la integración de subsistemas, el modelo de cascada se adecuará mejor en este caso.



4. Planeación:



El proyecto se revisa y se decide si se avanza con otro ciclo del espiral de Boehm, en caso de que se opte por esta opción, se trazan los planes para la siguiente fase del proyecto.

La ventaja principal entre utilizar el modelo de Boehm y otros modelos es su reconocimiento explícito de riesgo. El ciclo de espiral empieza con desarrollar objetivos de rendimiento y funcionalidad. Luego, se toman en cuenta las diferentes maneras en las que se puede alcanzar dichos objetivos tomando en cuenta las restricciones que nos presenta cada uno de ellos. De ahí, se resuelven dichos riesgos, recopilando información, con análisis más detallados, elaboración de prototipos y simulaciones.

Modelo Iterativo-Incremental

El desarrollo iterativo e incremental es la base de diferentes métodos de desarrollo de software como RUP (Rational Unified Process), Extreme Programming y otros métodos de desarrollo ágiles.

Consiste en realizar tareas agrupadas en etapas que se repiten, también conocidas como iteraciones. En cada iteración se repite un proceso que da un resultado más completo para el producto final, de esta manera quien lo utilice puede recibir beneficios del proyecto de manera creciente.

- Iterativo: se repiten las etapas del modelo en cascada, se rehace, refine y extiende lo que se tiene.
- Incremental: se integran regularmente los avances para crear una versión con sentido para el cliente.

Rational Unified Process (RUP)

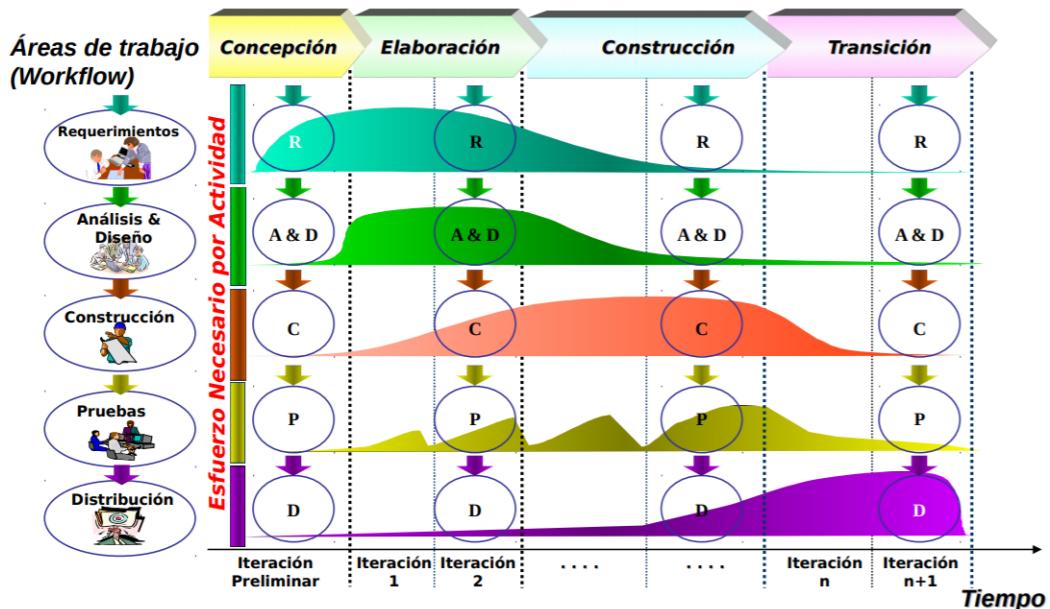
Es un proceso de desarrollo de software que junto con el Lenguaje Unificado de Modelado UML, forma una metodología empleada especialmente para sistemas orientados a objetos.

Principios del RUP

- *Adaptar el proceso:* considerando las características particulares del proyecto, así como el tamaño, tipo, restricciones y alcances del mismo.

- *Equilibrar prioridades*: encontrar un equilibrio entre los requerimientos de los participantes para evitar desacuerdos.
- *Demostrar valor iterativamente*: en cada iteración se analiza la opinión de los involucrados, la calidad del producto y la estabilidad del mismo, también se refinan los objetivos y los riesgos involucrados.
- *Colaboración entre equipos*: comunicación fluida y constante entre todos los involucrados.
- *Elevar el nivel de abstracción*: uso de conceptos reutilizables.
- *Ciclo de vida*: dentro de RUP el ciclo de vida es una implementación del desarrollo en espiral.

Fases de desarrollo de RUP



Cada fase se compone de un número de iteraciones que generan nuevas versiones del sistema. Las cuatro fases son:

- *Inicio o Incepción*: Donde se definen los objetivos y funcionalidades del sistema.
- *Elaboración*: Se define la arquitectura y los recursos con los que se cuenta.
- *Construcción/Desarrollo*: Se desarrolla el producto lo que incluye la programación, pruebas y documentación. Además, se refinan las etapas anteriores iterativamente.
- *Transición o Cierre*: Se entrega el producto, los manuales y las tareas de marketing también de forma iterativa.

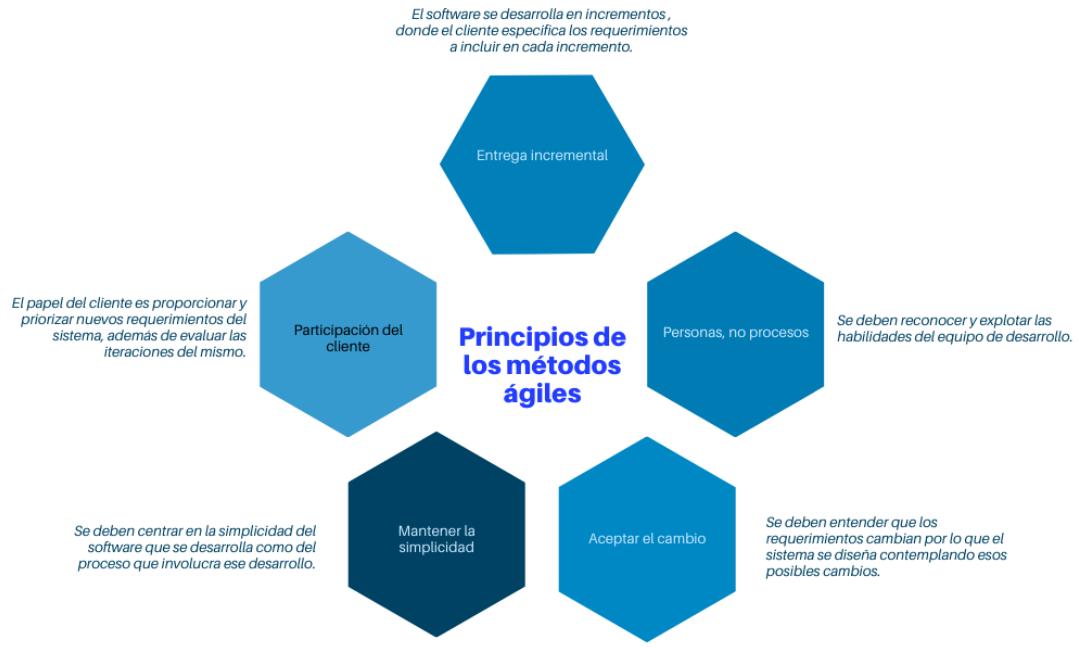
Modelo de desarrollo Ágil

En la década de 1990 se desarrollaron métodos ágiles para enfrentar los problemas como el incremento de gastos generales y documentación, así como entrega de software tardía. Estos métodos se centran en el software más que en su documentación, desarrollan software en una serie de incrementos y tienen como objetivo reducir la burocracia del proceso tanto como sea posible.

El enfoque de estos modelos se basa en el desarrollo iterativo e incremental, de tal forma que existe una evolución constante según lo requiera el proyecto. El trabajo se realiza con equipos altamente inmersos en el proceso en donde se toman decisiones a corto plazo.

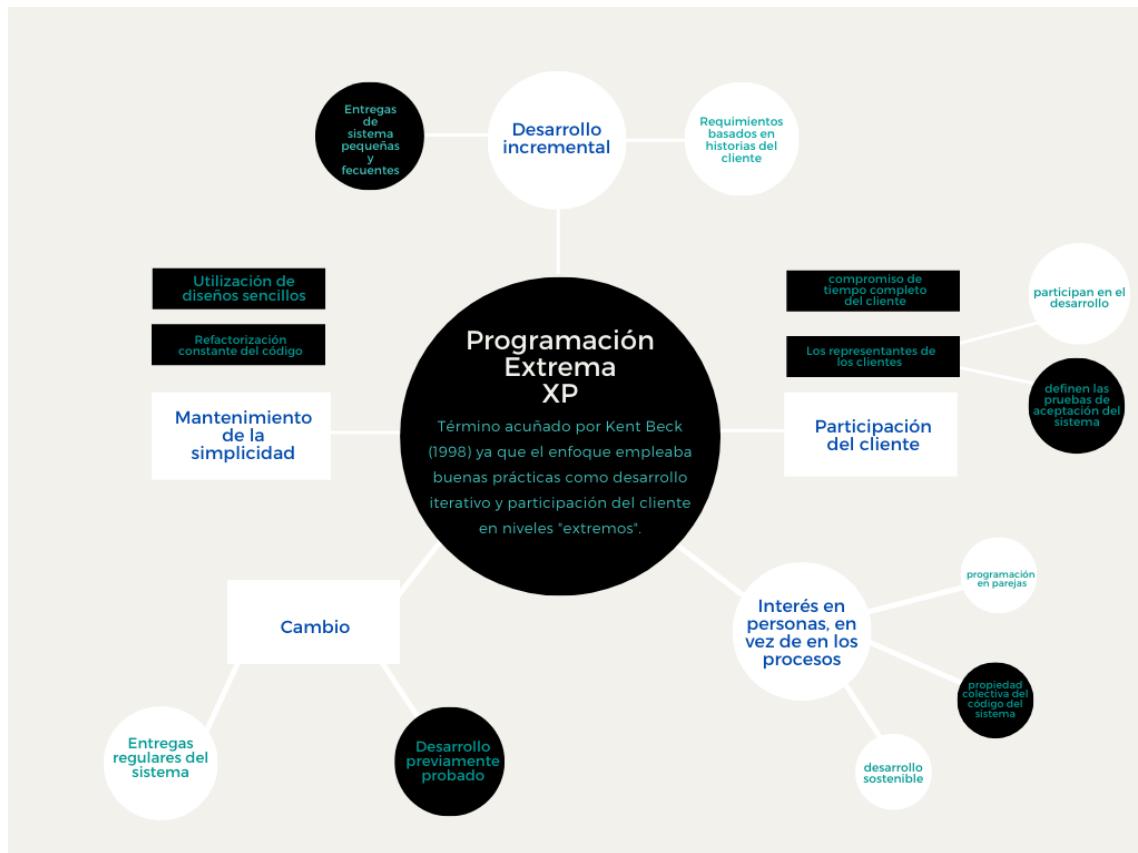
Metodologías ágiles

Existen diversos enfoques ágiles que se diferencian en sus prácticas o procesos, pero que comparten principios específicos para alcanzar sus objetivos.



Son diversas las metodologías ágiles que han surgido como respuesta a la constante necesidad de entregar productos de gran calidad en menor tiempo y costo, entre ellas se encuentran: Extreme Programming (XP), SCRUM, Kanban, Feature Driven Development (FDD), Agile Unified Process (AUP), Agile Modeling, Crystal Clear Methods, etc. A continuación, se describen dos de las más empleadas en la actualidad.

Extreme programming (XP)



PRÁCTICAS XP AMPLIAMENTE ADOPTADAS

Planificación incremental / historias de usuarios

- Se establece lo que se necesita implementar en cada incremento con ayuda del representante del cliente.
- Los requisitos están escritos como historias de usuario. Las historias que se incluirán en un comunicado están determinadas por el tiempo disponible y su prioridad relativa.

Pequeñas entregas

- Primero se desarrolla el conjunto mínimo de funciones útiles que proporcionan valor comercial.
- Las entregas del sistema son frecuentes y añaden funcionalidad de forma incremental a la versión anterior.

Desarrollo impulsado por pruebas

- En lugar de escribir código y luego probar ese código, los desarrolladores escriben primero las pruebas.
- Se utiliza un marco de prueba unitario automatizado para ejecutar las pruebas después de cada cambio.
- El código nuevo no debe "romper" el código que ya se ha implementado

Integración continua

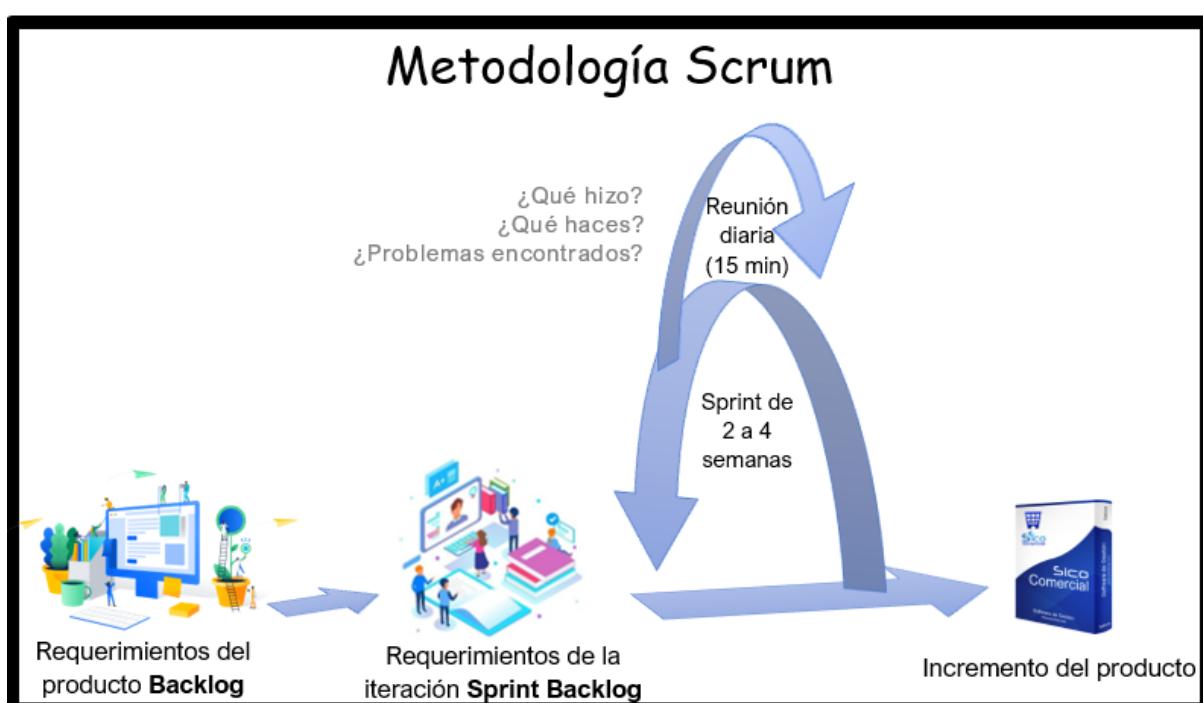
- Tan pronto como se completa el trabajo en una tarea, se integra en todo el sistema y se crea una nueva versión del sistema.
- Todas las pruebas unitarias de todos los desarrolladores se ejecutan automáticamente y deben tener éxito antes de que se acepte la nueva versión del sistema.

Refactorización

- Consiste en mejorar la estructura, legibilidad, eficiencia y seguridad de un programa.
- En cuanto se encuentren posibles mejoras se espera que todos los desarrolladores refactoricen el código para conservarlo simple y fácil de mantener.

SCRUM

Puede tomarse como punto de referencia para definir un proceso de desarrollo ya que va definiendo un conjunto de prácticas y roles dentro de un entorno de trabajo. En éste se hace uso de las entregas parciales y regulares del producto final, manejadas por prioridades que maneja el cliente o receptor del proyecto. Por ello, este método es indicado para proyectos de entorno complejo, donde los requisitos son cambiantes o poco definidos, o lo que se está entregando al cliente no es lo que se espera, este método es muy útil para resolver este tipo de situaciones, entre otras.



Como se muestra en la imagen el proceso de SCRUM está basado en sprints o ciclos temporales que normalmente son de 2 semanas y como máximo 4. Cada uno de estos debe de proporcionar un resultado completo y un incremento del producto final.

Una implementación de sistemas para gestionar el proyecto es utilizar una pizarra con notas autoadhesivas que se divide en tres columnas las cuales son: trabajo pendiente ("**backlog**"), tareas en proceso ("**in progress**") y hecho ("**done**"). Esto lo

hace más sencillo al equipo porque de un solo vistazo, una persona puede ver en qué están trabajando los demás.

Actividades

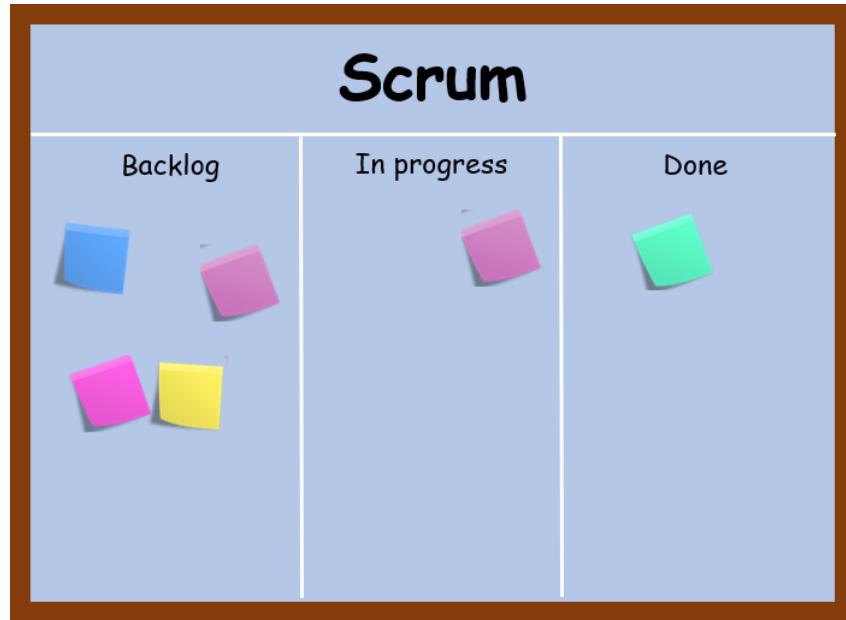
- *Planificación de ciclo.*

Cuenta con dos partes:

- Selección de requisitos. El cliente presenta al equipo los requisitos, se plantean las dudas del momento y se priorizan de forma que si el cliente los necesita estos puedan ser entregados.
- Planificación del ciclo. El equipo planifica la lista de tareas en cada ciclo y se autoasignan cada una de estas, además de que la estimación de esfuerzos y la resolución de objetivos se hace de manera conjunta, a fin de compartir conocimiento.
- *Ejecución del ciclo.* Cada día el equipo debe de realizar una reunión de sincronización a modo de que se vea el avance, es decir que se evalúe de manera conjunta el trabajo que se esté realizando.
- *Inspección y adaptación.*

Cuenta con dos partes:

- Revisión. Se muestra al cliente los requisitos completados en cada ciclo, en forma de incremento.
- Retrospectiva. El equipo analiza su forma de trabajar a modo de que se mejore la productividad del equipo con cada sprint.



Beneficios

Flexibilidad a cambios: El marco de trabajo está diseñado para adecuarse a las nuevas exigencias que implican proyectos complejos.

Mayor calidad del software: Obtener una versión de trabajo funcional después de cada iteración, ayuda a la obtención de un software de alta calidad.

Mayor productividad: El equipo se puede mover de manera autónoma para lo que mejor convenga en el software de acuerdo con lo que pide el cliente.

Predicciones de tiempo: Se conoce la velocidad media del equipo por sprint, con lo que es posible estimar cuándo se podrá hacer uso de una funcionalidad que todavía está en el trabajo pendiente (Backlog).

Reducción de riesgos: Desarrollar las funcionalidades de mayor valor y de saber la velocidad a la que el equipo avanza en el proyecto, permite despejar riesgos de manera anticipada.

¿Cuándo emplear un modelo de desarrollo de software específico?

Es claro que el desarrollo de software se puede lograr a través de diferentes modelos, sin embargo es importante distinguir en qué casos es mejor o más efectivo usar una u otra opción. En la siguiente tabla se enlistan situaciones concretas en las que los modelos pueden ser aprovechados de la mejor manera para lograr los resultados deseados en un proyecto.

MODELO DE DESARROLLO DE SOFTWARE	¿CUÁNDO EMPLEARLO?
MODELO EN CASCADA	Este modelo es ideal cuando los requerimientos del sistema se entienden bien y es improbable que se presente algún cambio durante el desarrollo del software ya que resultará difícil adecuarlo a las necesidades cambiantes del usuario.
MODELO EN V	Al ser una herramienta que facilita la organización de proyectos y su mantenimiento, se utiliza mucho en software para las autoridades públicas y ministerios, incluso es muy utilizado en el sector militar al enfocarse mucho en la disciplina en cada proceso del modelo.
MODELO DE PROTOTIPOS O MODELO DE DESARROLLO EVOLUTIVO	Ideal para proyectos que no disponen de mucho tiempo y donde los requerimientos no están tan claros o cambian constantemente. La interacción con el cliente-desarrollador es alta, por lo que se emplea este modelo cuando el

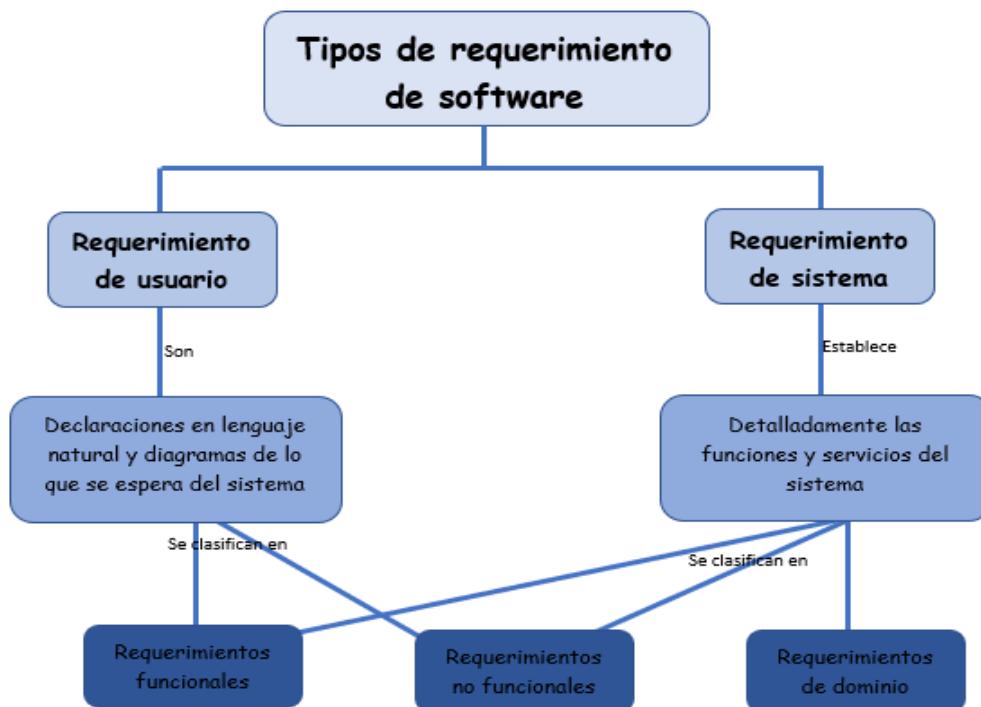
	cliente tiene disponibilidad de evaluar los prototipos presentados.
MODELO DE DESARROLLO INCREMENTAL	Este modelo es idóneo cuando se busca integrar los incrementos, con estrategias y una planeación para su unificación.
MODELO DE REUTILIZACIÓN DE SOFTWARE	Este modelo es conveniente cuando el tiempo es corto, al igual que los costos y además se debe contar con un tipo de software similar al que se va a desarrollar.
MODELO ITERATIVO-INCREMENTAL	Ideal cuando se tiene comunicación directa y constante con el cliente. También es útil cuando hay poco personal de desarrollo y si la fecha de entrega no es estricta.
MODELO EN ESPIRAL	Utilizado normalmente para proyectos en donde los costes en los fallos son muy altos, es decir que el proyecto está sujeto a los riesgos.
MODELOS ÁGILES	Se emplean principalmente en proyectos poco definidos y con requerimientos que cambian constantemente. Además, es necesaria una presencia fuerte y continua del cliente para el desarrollo correcto del proyecto.

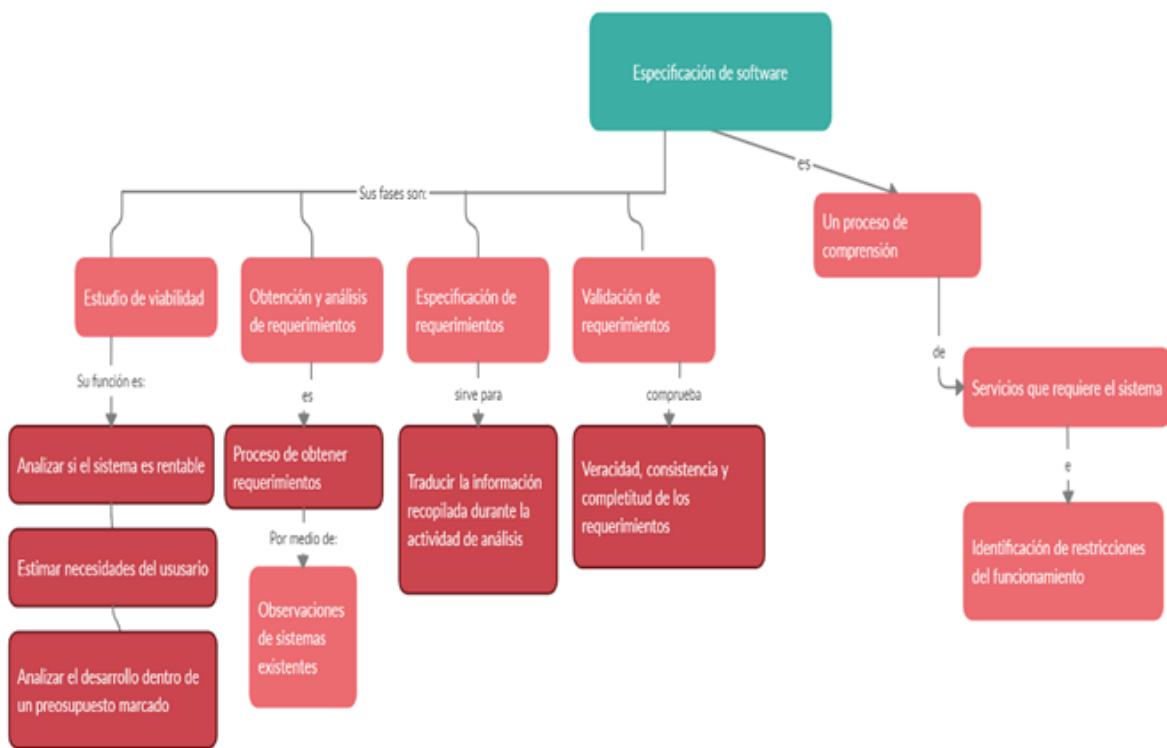
Ingeniería de requerimientos

La ingeniería de requerimientos se encarga de la detección, análisis y documentación de los requerimientos del sistema, así como la definición de las restricciones del mismo.

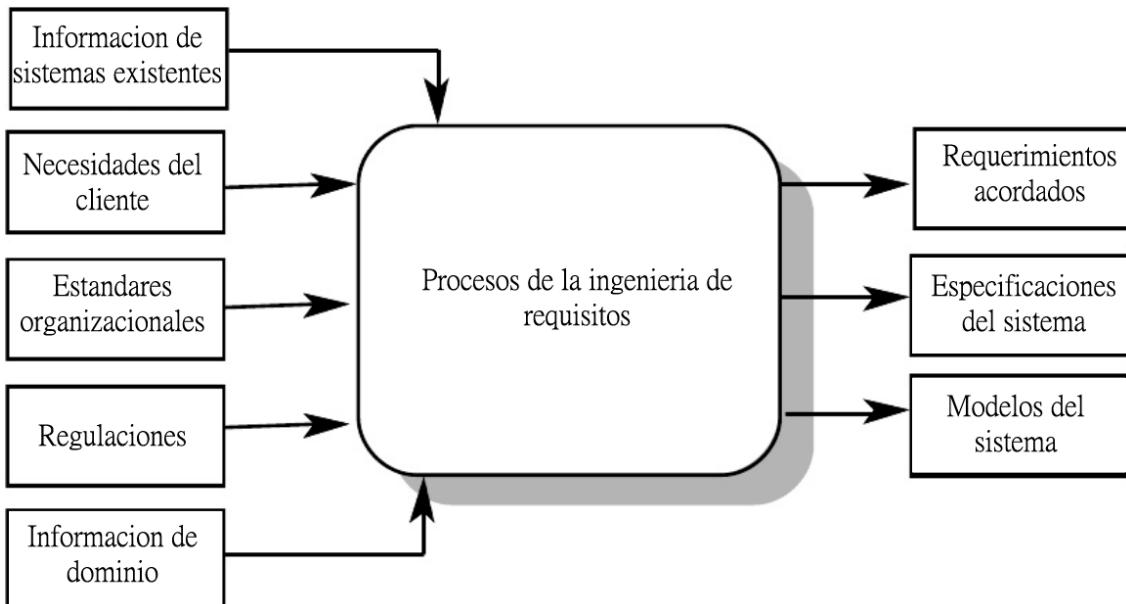
Partimos de la idea de que un requerimiento es la declaración de un servicio o restricción específica que presenta un sistema para satisfacer las necesidades de los usuarios de éste.

A continuación, se presentan dos esquemas para la comprensión general de la ingeniería de requerimientos; el primero describe los tipos de requerimientos de software, y el segundo, los procesos involucrados en la especificación de requisitos.





Datos de entrada y datos de salida



Como en todos los procesos, las entradas son aquellas que se van a transformar para conseguir un resultado específico o una salida. Específicamente en la ingeniería de requerimientos/requisitos podemos encontrar los siguientes datos de entrada y de salida:

Datos de entrada

- Información de sistemas existentes: Información sobre la funcionalidad de los sistemas para poder ser reemplazados por otros sistemas los cuales interactúan con el sistema especificado.
- Necesidades del cliente: Descripciones sobre lo que sistema necesita el cliente para apoyar su trabajo.
- Estándares organizacionales: Los estándares usados en una organización con respecto a la práctica del desarrollo del sistema, la calidad del manejo, etc.
- Regulaciones: Regulaciones externas como salud, seguridad, regulaciones que aplican al sistema.
- Información de dominio: Información general sobre el dominio de la aplicación del sistema.

Datos de salida

- Requerimientos acordados: Una descripción de los requerimientos del sistema la cual sea entendible para el cliente y sean acordadas con el.
- Especificaciones del sistema: Esta es una especificación de la funcionalidad del sistema más específica y detallada para poder ser producida en algunos casos.
- Modelos del sistema: Un grupo de modelos tales como flujo de información, un modelos de proceso, etc. que nos permite analizar el sistema desde diferentes perspectivas.

Requerimientos de usuario (DoRCU)

La metodología consta de las siguientes etapas:

- Elicitación de requerimientos
- Análisis de requerimientos
- Especificación de requerimientos
- Validación y certificación de los requerimientos.



Elicitación de requerimientos: Proceso donde se adquiere conocimiento del trabajo del cliente/usuario, para comprender sus necesidades y detallar restricciones. Por lo que al final se obtendrá un conjunto de requerimientos de todas las partes involucradas.

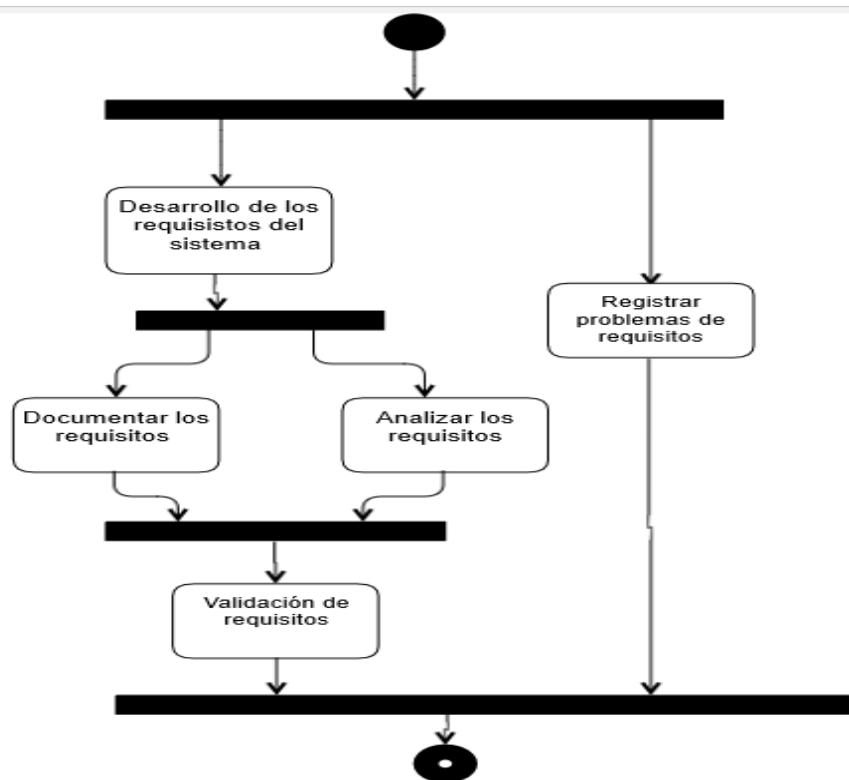
Análisis de requerimientos: Se estudian los requerimientos extraídos en la etapa previa para poder detectar la presencia de áreas no especificadas, requisitos contradictorios y peticiones que son irrelevantes. El resultado es eliminar todas las inconsistencias y equivocaciones que se han detectado. En esta etapa ya se realizan aproximaciones a un lenguaje técnico.

Especificación de requerimientos: Partiendo de lo elaborado en la etapa anterior tales como funciones, datos, requerimientos no funcionales, objetivos, restricciones de diseño, implementación o costos. Si se presentan dificultades para especificar un requerimiento se debe volver a la etapa anterior que se crea conveniente.

Validación y certificación de los requerimientos: Se realiza la integración y validación final de lo obtenido en cada una de las etapas anteriores dando como resultado final el documento de requerimientos.

Uno está destinado para el cliente o usuario, a los efectos de la certificación de los requisitos y el otro técnico, orientado a nutrir las restantes etapas de la ingeniería de software. El resultado puede ser la necesidad de retornar a una etapa anterior.

Pasos a seguir para el desarrollo de requerimientos del proyecto



Una vez comprendidos los procesos involucrados en la ingeniería de requerimientos, enlistamos los pasos a seguir en esta fase para conseguir un correcto desempeño de la misma.

- 1. Recopilación de requerimientos:** Siendo nosotros también stakeholders del proyecto, podemos comprender las necesidades de manera más directa. Debemos determinar cuáles son las funciones que requiere cubrir el programa y sus limitaciones, y convertirlas en requerimientos. Necesitamos conocer si existen restricciones y considerarlas en esta recopilación.
- 2. Análisis y documentación de requerimientos:** Para revisar la consistencia y congruencia de los requisitos para evitar ambigüedades; además, se evaluará si se cubren todas las necesidades de los usuarios y se propondrán los requisitos que hagan falta.

Una vez analizados, se procede a documentar detalladamente cada requerimiento con lenguaje apropiado y de fácil comprensión. Para este paso será importante tomar en cuenta los estándares para tener un documento eficiente y claro.

3. **Validación de requisitos:** Dentro de este apartado existen diversos objetivos, uno de los principales será comprobar que el sistema descrito por la especificación de requisitos, en efecto corresponda a las necesidades del cliente y del usuario; por lo que el proceso para realizar la validación está enfocada en analizar la validez, consistencia, completitud, realismo y verificabilidad del sistema a través de diversas técnicas, tales como prototipos de interfaz de usuario y recorridos de casos de uso.

Adicionalmente y de manera continua, será importante hacer un registro de los inconvenientes que se presenten al desarrollar los requisitos del sistema. Considerando que emplearemos un método ágil, sabemos que la evaluación de riesgos y requisitos será constante y tendrá el objetivo de evitar consecuencias graves y conservar la calidad del software durante todo su desarrollo.



Técnicas de recolección de requerimientos

Nos referimos a la recolección de datos, al uso de herramientas y/o técnicas que puede utilizar un analista para desarrollar un sistema, tal como son las entrevistas, escenarios, observaciones, etnografía, entre otros, esto con el fin de reunir o buscar información que será útil para establecer los requisitos del sistema.

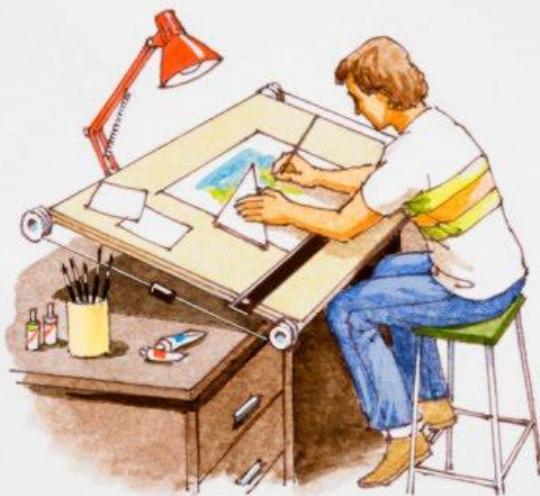
Entrevistas

Es una de las técnicas más empleadas para la recolección de requerimientos cuyo objetivo es la obtención de puntos de vista particulares de los stakeholders del proyecto a través de una conversación estructurada. En el siguiente esquema se pueden distinguir las particularidades de esta técnica.



Desarrollo de prototipos

Desarrollo de prototipos nos es de gran ayuda en caso de que se presentara la situación en que los requisitos del cliente no sean claros o no se entiendan muy bien, ya que se crean múltiples prototipos intentando acercarse a lo deseado.



Esta técnica permite solventar objeciones del cliente tipo “No sé lo que quiero, pero lo sabré cuando lo quiera”.

Ventajas

- Nos permite identificar las discrepancias de forma más sencilla y rápida entre los desarrolladores y los usuarios.
- El prototipo que más se acerque a lo que quiera el cliente se utiliza como base de la producción.
- Rápidamente se puede disponer de un sistema que funcione y demuestre el funcionamiento de la aplicación.
- Permite darle virtualizaciones al usuario para darle una idea de cómo será el proyecto final.

Esta técnica es útil cuando :

- El área no está bien definida.
- El costo de rechazo de parte de los usuarios es alto.
- Será necesario examinar el impacto que tendrá en los usuarios / organización que le darán uso al software.

Esta técnica expone los requisitos de la interfaz del usuario y utiliza también los requisitos obtenidos en las entrevistas, observaciones, escenarios, etc...

Observación



Reutilización de requerimientos

La reutilización de requerimientos es una técnica utilizada en algunos casos para ahorrar tiempo y dinero a lo largo de un proyecto. Como su nombre indica, el proceso consiste en reutilizar requerimientos que han sido previamente ocupados en otros proyectos, pero por diversas circunstancias, se pueden volver a usar en el trabajo actual.

Este proceso tiene diversas ventajas cuando se habla de tiempo y esfuerzo, y es que reutilizar un requerimiento implica que ya ha sido previamente estudiado y por consecuencia, validado; eso quiere decir que existe un equipo de trabajo que estudió los impactos que se pueden lograr, las desventajas y ventajas que otorga

dicho requerimiento, por lo que ese esfuerzo, dinero y sobre todo tiempo, se pueden prácticamente erradicar cuando reutilizamos un requerimiento. Por lo tanto es una técnica importante a tomar en cuenta en cualquier tipo de proyecto y sobre todo cuando las necesidades se ajusten a requerimientos previamente utilizados.

The infographic is divided into two main sections. The left section features a background of various business-related words like 'BUSINESS', 'PLANNING', 'MANAGEMENT', 'MARKETING', 'MODELING', 'DATA', 'ANALYSIS', 'DESIGN', 'QUALITATIVE', 'FRAUD', 'GOODS', 'FACTORY', 'SERVICES', 'MAINTENANCE', 'SYSTEM', 'STRATEGY', and 'CLOUD'. Overlaid on this are the words 'REUTILIZACIÓN DE REQUERIMIENTOS' in large, bold, black letters, with 'REUTILIZACIÓN' in red. Below this, a hand holds a pen over a white area containing the text: 'Consiste en tomar requisitos que se han desarrollado para usarlos en un sistema diferente.' To the right, a dark blue vertical bar contains the text 'Ahorra tiempo y esfuerzo' in large white letters, above a green circular icon of a clock with a green arrow pointing clockwise. Below the icon is a horizontal line, followed by the text 'Posibilidades de reutilizar:' and a bulleted list. At the bottom is a yellow dollar coin with a hand pointing at it against a dark background.

REUTILIZACIÓN DE REQUERIMIENTOS

Consiste en tomar requisitos que se han desarrollado para usarlos en un sistema diferente.

Puede llegar a proporcionar grandes ahorros debido a que los requerimientos ya han sido validados.

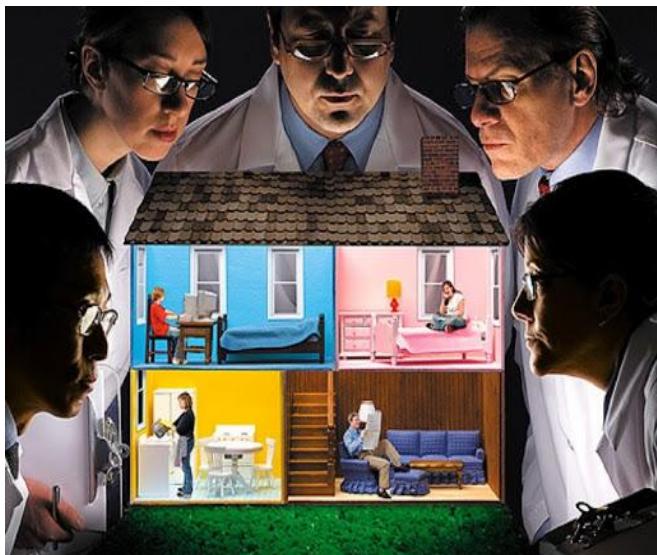
Ahorra tiempo y esfuerzo

Posibilidades de reutilizar:

- Cuando se refiere a proporcionar información del dominio de la aplicación
- Cuando se refiere al estilo de presentación de la información
- Donde el requisito refleja las políticas de la empresa

Etnografía

Esta técnica se utiliza para entender los requerimientos sociales y organizacionales del sistema, donde el analista entra en el ámbito laboral donde este se utilizará. El trabajo que se hace día con día es observado y se hacen las notas de las tareas reales de las que cada integrante del proyecto está involucrado.



Este tipo de estudios ayuda a la revelación de procesos críticos que por lo general no se toman con la importancia que deberían. El problema de este proceso es que al centrarse en el usuario final, no se enfoca en todos los requerimientos del dominio, además este enfoque no está diseñado para que se identifiquen nuevas propiedades del sistema,

es por ello que no es un enfoque completo y debe de ir acompañado de otras técnicas.

Está técnica es efectiva para descubrir dos tipos de requerimientos:

- Requerimientos que proceden del conocimiento y colaboración de las personas.
- Requerimientos que proceden de cómo trabajan realmente las personas.

Escenarios

Documentar el comportamiento del sistema cuando se le presentan eventos específicos. Cada evento de interacción distinto, o la selección de un servicio del sistema, se documenta como un escenario de evento distinto e incluso se puede documentar las excepciones que surgen en el sistema.

Los escenarios de eventos incluyen una descripción del flujo de datos y las acciones del sistema.

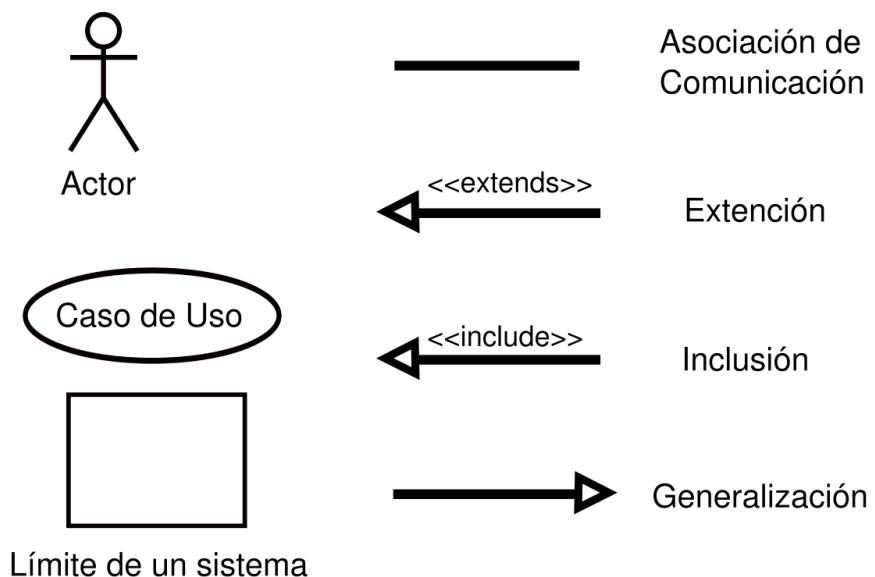
Un escenario empieza con un bosquejo de interacción y se van agregando detalles:

- Una descripción del estado inicial del sistema.

- Una descripción del flujo normal de eventos.
- Una descripción de lo que puede ir mal y cómo manejarlo.
- Información de otras actividades que se pueden llevar a cabo al mismo tiempo.
- Descripción del estado del sistema al terminar.

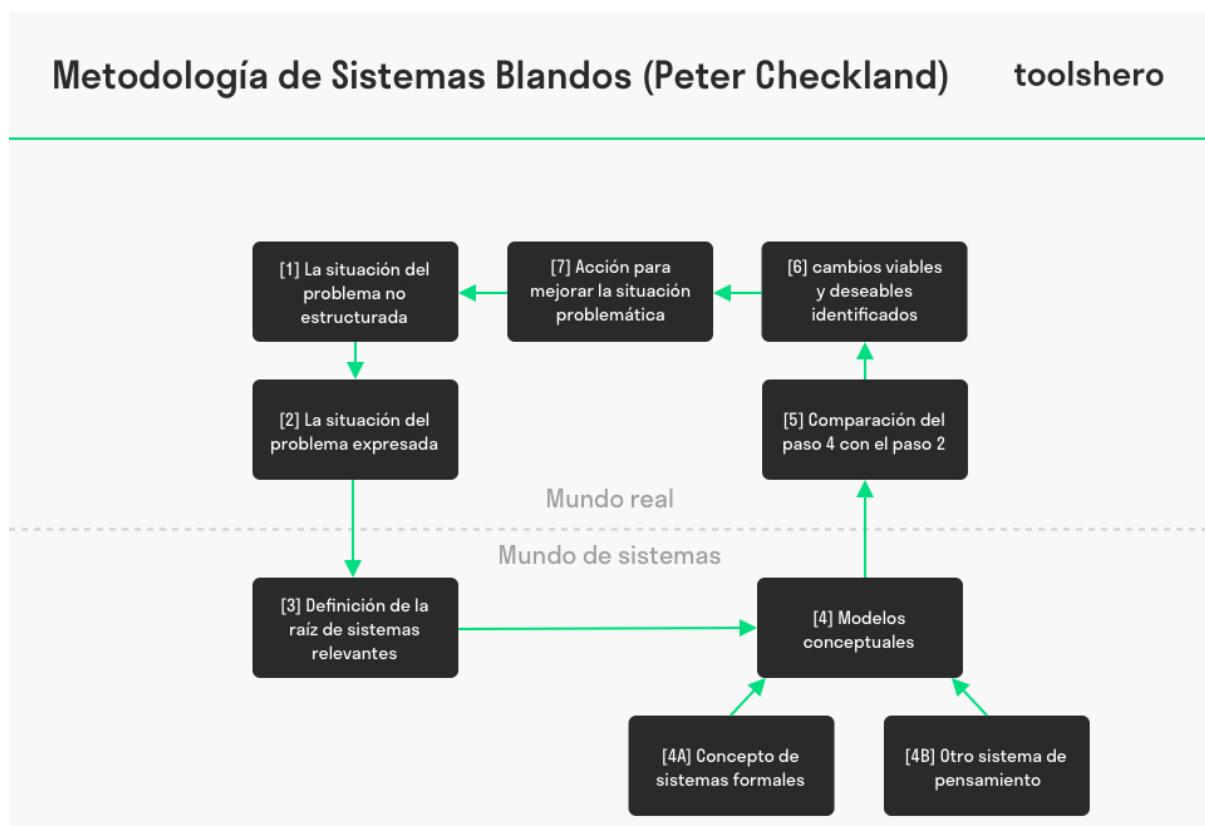
Los Casos de Uso son una técnica que se basa en escenarios para la obtención de requerimientos. Son una técnica fundamental que se utiliza para analizar y describir modelos de sistemas orientados a objetos.

Un caso de uso identifica a los actores involucrados en una interacción y nombra al tipo de ésta.



Metodologías de sistemas suaves

El SSM se originó de la comprensión de los sistemas “duros” estructurados y aunque no son técnicas para la obtención de requisitos detallados, ayudan a entender un problema y su contexto organizacional.



Safari de nuestro proyecto

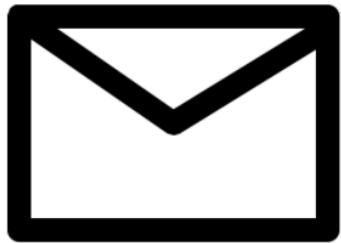
La finalidad de emplear un safari es lograr posicionarnos en el lugar de un usuario que no tiene experiencia en el uso de estas aplicaciones y evaluar las posibles dificultades a las que se enfrentaría en un escenario cotidiano; lo anterior con el objetivo de entender sus necesidades y refinar los requerimientos del sistema.

A continuación se ilustra una de las actividades frecuentes que se realizan en una plataforma escolar y algunas de las complicaciones que podrían presentarse.



Safari: Caso de estudio acceder a una clase en la plataforma escolar

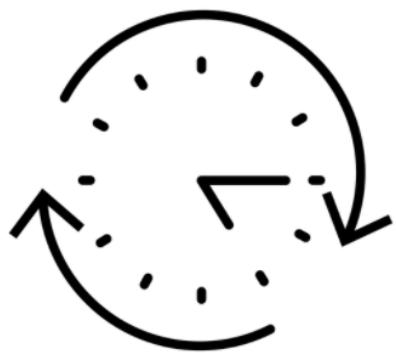
Escenario: Un alumno(a) debe acceder a la plataforma escolar para tomar una clase online. Deberá contar con un correo institucional, su contraseña y estar agregado(a) a dicha clase.



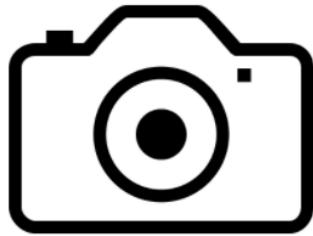
Intenta acceder a la plataforma pero no encuentra fácilmente dónde colocar su correo y contraseña



Encuentra el área de acceso a la plataforma pero al querer ingresar se le indica que su contraseña es errónea



Logra restablecer su contraseña pero le queda poco tiempo para acceder a su clase y no encuentra el enlace para la reunión.



Accede a la clase pero no sabe como desactivar su cámara y su micrófono.

Puntos favorables

- Puede tomar clase sin importar su localización geográfica.
- Se evitan gastos de transporte.
- Al poder tomar la clase sin traslados, se optimizan tiempos.

Aspectos difíciles

- No estar familiarizado con la plataforma.
- No es la misma interacción que se tiene de manera presencial.
- Los exámenes llegan a ser más problemáticos.
- Requiere autodisciplina.



Requerimientos del proyecto

El principal objetivo de este apartado es explicar con mayor detalle los alcances que tendrá el sistema a desarrollar. La plataforma tendrá un diseño atractivo para los estudiantes y maestros ya que serán nuestros usuarios principales.

La descripción de requerimientos se desarrollará con un lenguaje claro y natural para que pueda ser consultada sin inconvenientes por todos los interesados en el funcionamiento de la plataforma.

Síntesis de requerimientos funcionales

Un requisito funcional define lo que esperamos que haga un sistema, aquí se describe la interacción entre el sistema y el entorno en el que estará funcionando, los servicios y funciones que deberá proveer.

Los requisitos funcionales que deberá cubrir la plataforma escolar se describen formalmente en el Anexo 1 de este documento. A continuación mostramos una síntesis de los mismos organizados en tablas:

Identificador de requerimiento	RF01 UC-001
Nombre del requerimiento	Crear cuenta en plataforma
Características	Los usuarios que cuenten con un correo institucional vigente podrán crear una cuenta en la plataforma.
Descripción del requerimiento	El sistema permitirá crear una cuenta a los usuarios que cuenten con su correo institucional vigente.
Entradas	Correo de usuario con el formato institucional (p.ej. xxxxx@xxxx.buap.mx), contraseña(8 caracteres alfanuméricos)
Proceso	El usuario accede al enlace de la plataforma y entrará a la sección de “crear cuenta”. En pantalla se solicitarán los datos personales del usuario, se validarán esos datos y se pedirá confirmación a través de un enlace enviado al correo correspondiente. Una vez confirmado el correo, se solicitará la

	creación de una contraseña con el formato anteriormente especificado, se confirmarán los datos y se creará la cuenta.
Salidas	Mensaje de éxito en caso de cumplir con todos los pasos correctamente. Mensaje de error en caso de no haber llenado un campo. Mensaje de error en caso de fallar la validación de datos.
Prioridad de requerimiento	Alta

Identificador de requerimiento	RF02 UC-002
Nombre del requerimiento	Autentificar y autorizar usuarios
Características	El usuario con los permisos requeridos podrá ser autorizado por el sistema
Descripción del requerimiento	Ingresar a la plataforma. El usuario podrá ingresar a la plataforma en cualquier momento y sin límite de acceso siempre y cuando cuente con sus claves de acceso correspondientes, en caso de que sean incorrectos tiene cinco intentos antes de bloquear el acceso a la cuenta.
Requerimiento no funcional	El sistema debe tener una velocidad de respuesta rápida, ya sea para ingresar a la plataforma o para notificar de clave errónea.
Entradas	Correo de usuario(p.ej.xxxxx@xxxx.buap.mx) contraseña(8 caracteres alfanuméricos).
Proceso	El usuario accederá a la plataforma mediante su navegador, después elegirá la opción de ingresar con sus datos por lo que el sistema solicita su correo y su contraseña. El usuario ingresa sus datos y selecciona 'Ingresar'.
Salidas	Ingreso satisfactorio a la plataforma si los datos son correctos. Mensaje de datos incorrectos en caso de que se hayan colocado mal. Mensaje de inexistencia de usuario en caso de que el usuario no esté registrado.
Prioridad de requerimiento	Alta

Identificador de requerimiento	RF02 UC-003
Nombre del requerimiento	Validar usuario
Características	El administrador hará válida las entradas de los usuarios según un determinado campo
Descripción del requerimiento	Validar usuario. El usuario ingresará sus datos para poder ingresar a la plataforma y el administrador pueda darle acceso.
Requerimiento no funcional	El sistema deberá de ser intuitivo y fácil de manipular. El sistema debe tener una velocidad de respuesta rápida para que al ingresar los datos y el sistema los verifique, este pueda dar una respuesta al usuario.
Entradas	Correo de usuario(p.ej. xxxxx@xxxx.buap.mx) contraseña(8 caracteres alfanuméricos).
Proceso	El usuario abrirá la plataforma, para que posteriormente el sistema muestre los campos a ingresar. El usuario ingresa su correo y contraseña en sus campos correspondientes. El administrador hará un proceso de validación de datos, verificando que los campos estén registrados y llenados.
Salidas	El sistema abrirá la plataforma en caso de no haber errores El sistema enviará un mensaje de error en caso de no haber llenado todos los campos requeridos El sistema enviará mensaje de error en caso de no haber encontrado los datos ingresados.
Prioridad de requerimiento	Alta

Identificador de requerimiento	RF03 UC-004
Nombre del requerimiento	Gestión de aula virtual.
Características	El usuario con los permisos requeridos podrá gestionar las opciones del curso.
Descripción del requerimiento	Agregar curso. El usuario podrá agregar un curso siempre y cuando cuente con la información y clave del curso.

Requerimiento no funcional	El sistema debe ser capaz de tener seguridad para bloquear accesos maliciosos de personas que no cuenten con la clave de curso correcta.
Entradas	Información y clave del curso.
Proceso	El usuario accederá a la plataforma, después seleccionará la opción de ‘Aregar curso’, por lo que el sistema solicitará los datos y la clave del curso; el usuario llenará los datos y le dará click a ‘Aregar’.
Salidas	Mensaje de éxito y acceso al curso en el caso de que todo sea correcto. Mensaje de datos incorrectos en el caso de que los datos sean erróneos. Mensaje de clave incorrecta en el caso de que la clave se encuentre errónea.
Prioridad de requerimiento	Alta

Identificador de requerimiento	RF03 UC-005
Nombre del requerimiento	Gestión de aula virtual.
Características	El usuario con los permisos requeridos podrá gestionar las opciones del curso.
Descripción del requerimiento	Dar de baja curso. El usuario podrá dar de baja un curso siempre y cuando se encuentre inscrito a por lo menos uno y cuente con las características necesarias para dar de baja.
Requerimiento no funcional	El sistema será capaz de evitar accesos de personas que no sean dueñas de la cuenta para así evitar la baja de cursos no deseados.
Entradas	Información del curso a dar de baja.
Proceso	El usuario inicia sesión con su cuenta institucional, posteriormente el sistema mostrará los cursos a los que se encuentra inscrito, el usuario selecciona el curso a dar de baja, dentro del mismo le da click al botón ‘Dar de baja’, por lo que el sistema pedirá una confirmación y la dará de baja.
Salidas	Mensaje de éxito en caso de que todo salga de manera correcta.

	Mensaje de error en caso de que ese usuario no pueda dar de baja cursos.
Prioridad de requerimiento	Alta.

Identificador de requerimiento	RF03 UC-006
Nombre del requerimiento	Gestión de aula virtual.
Características	Crear grupos entre los usuarios registrados en la plataforma
Descripción del requerimiento	Crear grupos entre los usuarios. Creación de grupos entre los usuarios, muy diferente a los grupos de un curso, estos serán de menor tamaño y se podrán compartir archivos, realizar chats o llamadas.
Entradas	Matrícula (los primeros cuatro números deben ser mayores a 2010 y menores a 20020) Correo electrónico (xxxxx@xxxx.buap.mx) Nombre completo (menor a 30 caracteres)
Proceso	El usuario entra al menú lateral de opciones > Grupos > Crear grupo > donde podrá registrar a los miembros y personalizar el grupo > Guardar grupo
Salidas	Mensaje de grupo creado con éxito o error en caso de haber ocurrido. Exceso de miembros. Notificación a los miembros agregados.
Prioridad de requerimiento	Media

Identificador de requerimiento	RF03 UC-007
Nombre del requerimiento	Gestión de aula virtual.
Características	El alumno podrá tomar notas en cualquier momento de la sesión virtual en su computadora.

Descripción del requerimiento	Tomar notas : Poder tomar puntos importantes de la sesión sin perder algo importante por voltear a escribir a otra parte
Requerimiento no funcional	El sistema debe notificar al alumno si hay algún problema relacionado con el uso de las notas antes de empezar
Entradas	Acceso a la sesión, uso de la plataforma
Proceso	El alumno, en la videoconferencia, presiona el símbolo de nota y aparecerán notas en pantalla lo cual le permitirá tomar notas a lo largo de la sesión
Salidas	Mensaje de error en caso de tener problemas con el almacenamiento de las notas. Las notas en pantalla.
Prioridad de requerimiento	Bajo

Identificador de requerimiento	RF04 UC-008
Nombre del requerimiento	Gestión de clases.
Características	El docente podrá programar una, o varias videoconferencias.
Descripción del requerimiento	Programar Videoconferencia: El docente programa una videoconferencia dando a conocer a sus alumnos cuando tomará lugar con anticipación.
Requerimiento no funcional	El sistema notifica al profesor si no choca con alguna clase de otra materia.
Entradas	Acceso al grupo de clase
Proceso	El profesor ingresa al grupo de la materia, dala click en programar sesión, registrará los datos y quedará lista.
Salidas	Mensaje de error en caso de no poder poder programar la clase con su debida descripción. Mensaje de éxito en caso de haber podido programar videoconferencia.
Prioridad de requerimiento	Bajo

Identificador de requerimiento	RF04 UC-009
Nombre del requerimiento	Gestión de clases.
Características	El alumno podrá grabar la clase con el permiso de su docente.
Descripción del requerimiento	Grabar Clase: El alumno podrá grabar la clase para poder repetirla después en caso de dudas o algún otro inconveniente.
Requerimiento no funcional	El sistema deberá de notificar al alumno si no es posible grabar la clase.
Entradas	Acceso a la sesión, autorización del docente para dicha acción.
Proceso	El alumno, en la videoconferencia, presiona el símbolo de grabar, y avisará al profesor con una notificación dándole de opción si acepta que el alumno la grabe o rechazarla.
Salidas	Mensaje de error en caso de no poder grabar la clase. Mensaje de rechazo en caso de que el profesor no autorizará la grabación de la clase. Logotipo de grabación en una de las esquinas de la pantalla, dando a conocer al usuario que ya empezó la grabación.
Prioridad de requerimiento	Bajo

Identificador de requerimiento	RF04 UC-010
Nombre del requerimiento	Gestión de clases
Características	El sistema tomará asistencia de acuerdo a que alumnos ingresaron a la videoconferencia.
Descripción del requerimiento	Revisar asistencia. El sistema registrará que alumnos entraron a la videoconferencia y a la hora que entraron para ver si lo registra con retardos o no.
Requerimiento no funcional	El sistema debe de tomar registro de que alumnos entraron a la videoconferencia.
Entradas	Correo de usuario(p.ej. xxxxx@xxxx.buap.mx) contraseña(8 caracteres alfanuméricos)

Proceso	El docente accedera a la información de la sesión y se desplegará una lista con los miembros que accedieron y también mostrará a los usuarios que ingresaron tarde
Salidas	Lista de estudiantes que ingresaron a la videoconferencia y quienes llegaron tarde.
Prioridad de requerimiento	Alta.

Identificador de requerimiento	RF04 UC-011
Nombre del requerimiento	Gestión de clases
Características	El usuario con los permisos requeridos podrá gestionar las clases en el aula virtual.
Descripción del requerimiento	Iniciar videoconferencia. El usuario podrá iniciar una videoconferencia en el momento que desee, siempre y cuando tenga los permisos correspondientes.
Requerimiento no funcional	El sistema debe tener una velocidad de respuesta rápida para iniciar la videoconferencia, el sistema debe de tener una calidad de imagen nítida y un sonido estable.
Entradas	El usuario accederá a la plataforma, después ingresará al curso deseado y le dará click al botón ‘Iniciar videoconferencia’, por lo que el sistema solicitará la configuración de acceso y el usuario dará click en ‘Iniciar’.
Proceso	Inicio de videoconferencia. Mensaje de error en caso de no contar con internet. Mensaje de error en caso de que el sistema se encuentre saturado.
Salidas	Alta

Identificador de requerimiento	RF05 UC-012
Nombre del requerimiento	Gestión de aula virtual.

Características	El usuario podrá descargar los archivos que suba el profe para que pueda acceder a ellos después de manera offline.
Descripción del requerimiento	Descargar contenido. El alumno podrá descargar los archivos del grupo para acceder a ellos de forma offline.
Requerimiento no funcional	El sistema debe de tener una velocidad de descarga rápida para que el alumno pueda acceder lo más rápido posible a los archivos
Proceso	El alumno ingresará al grupo donde se encuentre lo que quiera descargar, dará clic en opciones y en la parte de descargar
Salidas	Aparecerán las descargas en la sección de descargas
Prioridad de requerimiento	Alta

Identificador de requerimiento	RF05 UC-013
Nombre del requerimiento	Gestión de aula virtual
Características	El usuario con los permisos requeridos podrá gestionar las opciones dentro del aula virtual.
Descripción del requerimiento	Consultar contenido del curso. El usuario podrá consultar el contenido disponible en los cursos.
Entradas	El usuario deberá iniciar sesión con su cuenta y entrar al curso deseado.
Proceso	El usuario accede a la sección “contenido del curso”. En pantalla podrá observar el contenido disponible y seleccionar los enlaces que quiera visualizar. El usuario podrá descargar el contenido en caso de que esté disponible para descarga.
Salidas	Mensaje de error en caso de que no haya contenido disponible en el momento. En pantalla se verá el contenido a consultar disponible.
Prioridad de requerimiento	Alta

Identificador de requerimiento	RF05 UC-014
Nombre del requerimiento	Gestión de aula virtual
Características	El usuario con los permisos requeridos podrá gestionar las opciones dentro del aula virtual.
Descripción del requerimiento	Realizar entrega de una tarea. El usuario podrá realizar la entrega de tareas asignadas.
Entradas	Los documentos que se pueden adjuntar en las tareas deberán cumplir con los siguientes formatos: pdf, xlsx, doc, pptx, docx, jpg, jpeg, png, con un tamaño no mayor a 10MB si se adjuntan directamente.
Proceso	El usuario accederá a la tarea asignada y se le mostrará la opción de adjuntar archivo, procederá a elegir los archivos y confirmar su carga en la plataforma. Posteriormente, el usuario seleccionará “entregar tarea” y confirmará la entrega.
Salidas	Mensaje de éxito si se concretó la entrega. Mensaje de error en caso de que los archivos adjuntos no cumplan con el formato indicado. Mensaje de error en caso de que los archivos rebasen el tamaño permitido.
Prioridad de requerimiento	Alta

Identificador de requerimiento	RF05 UC - 015
Nombre del requerimiento	Gestión de aula virtual.
Características	Crear recordatorios personalizados.
Descripción del requerimiento	Crear recordatorios. Cada usuario podrá crear sus propios recordatorios personalizados.
Entradas	Fecha y hora (deben ser fecha y hora después de la hora y fecha de la creación del recordatorio, la fecha no mayor a 1 año) Texto (Asunto del recordatorio) Archivo de música (.mp3, .mp4)

Proceso	El usuario entra al menú lateral de opciones > Recordatorios > Crear recordatorio > donde podrá crear y personalizar el recordatorio> Guardar recordatorio
Salidas	Mensaje de recordatorio creado con éxito. Error en la fecha y hora. Error en los formatos subidos son incorrectos. Error de que el asunto se quede vacío.
Prioridad de requerimiento	Media

Identificador de requerimiento	RF05 UC-016
Nombre del requerimiento	Gestión de aula virtual
Características	Los alumnos podrán mensajearse con otros compañeros o con docentes.
Descripción del requerimiento	Enviar mensajes. Los alumnos podrán comunicarse con otros compañeros o con docentes en cualquier momento que lo necesiten.
Entradas	El alumno debe de tener su usuario y su contraseña para poder mensajear con otros alumnos.
Proceso	El alumno entra a la sección de chats, da clic en el signo de más y buscará al alumno/docente con el que se querrá escribir.
Salidas	Chat con compañeros o docentes.
Prioridad de requerimiento	Media

Identificador de requerimiento	RF05 Caso de uso: UC- 017
Nombre del requerimiento	Gestión de aula virtual
Características	Los alumnos tendrán la posibilidad de visualizar el archivo que deseen cargar como tarea para poder ver una vista previa del documento que quieran entregar para no caer en

	equivocaciones o ver algún error y arreglarlo antes de ser enviado.
Descripción del requerimiento	Previsualización de archivos. El sistema contará con una previsualización para que el alumno pueda ver que está a punto de enviar antes de enviarlo.
Entradas	Archivo que se desee subir a la plataforma que cuente con los siguientes formatos: (.pdf , .doc , .docx , .pptx , .xlsx , .mp4 , .jpg , .png , .html , .css , .cpp , .cs , .txt , .rar , .zip)
Proceso	El usuario entra al apartado de tareas y selecciona la tarea a entregar, después da click en el apartado de “subir”, examina sus documentos para seleccionar el archivo a subir. Una vez subido, saldrá una previsualización del archivo en la misma página, puede elegir si enviar, cancelar, o subir otro más. Si escoge un archivo cuya extensión no se pueda visualizar saldrá un mensaje que no se pudo mostrar la visualización.
Salidas	Visualización del archivo. Botones para enviar, cancelar, o agregar más archivos.
Prioridad de requerimiento	Alta

Identificador de requerimiento	RF05 UC-018
Nombre del requerimiento	Gestión de aula virtual
Características	El usuario con los permisos requeridos podrá gestionar las opciones dentro del aula virtual.
Descripción del requerimiento	Acceder a videollamada activa. El usuario podrá acceder a una videollamada/sesión activa de los cursos en los que se encuentre inscrito.
Entradas	El usuario deberá estar dado de alta en al menos un curso. Videollamada activa.
Proceso	En pantalla se mostrará la opción de “unirse a videoconferencia”, el usuario accederá y se solicitará que confirme el uso de audio y video, después de configurar las

	opciones, el sistema permitirá el acceso al usuario a esa videoconferencia.
Salidas	Pantalla con videoconferencia activa. Mensaje de error en caso de no poder configurar el audio y/o video
Prioridad de requerimiento	Alta

Identificador de requerimiento	RF05 UC-019
Nombre del requerimiento	Gestión de aula virtual.
Características	Usuarios registrados con su correo institucional tanto docentes como alumnos que creen una cuenta en la plataforma podrán bloquear llamadas y mensajes entrantes de personas específicas.
Descripción del requerimiento	Bloquear mensajes/llamadas de otros usuarios. El sistema permitirá a tanto docentes como alumnos la opción de bloquear mensajes/llamadas de otros usuarios.
Entradas	Correo de usuario con el formato institucional (p.ej. xxxx@xxxx.buap.mx), contraseña(8 caracteres alfanuméricos) Mensaje del usuario que se desea bloquear. Especificar el usuario a bloquear.
Proceso	El usuario recibirá un mensaje de un usuario, podrá decidir si bloquear al usuario, en el caso de que lo quiera bloquear, deberá dar click en el perfil del usuario a ser bloqueado y dar click en el apartado de “Bloquear usuario”, después deberá confirmar la acción. Después de esto, el usuario dejará de tener comunicación directa de ese usuario en particular, y el usuario que ha sido bloqueado se le deshabilitará la opción mantener cualquier comunicación directa con el usuario que lo haya bloqueado.
Salidas	Texto de opciones de perfil del usuario. Opciones del perfil del usuario. Mensaje para pedir confirmación de bloqueo. Mensaje de bloqueo exitoso.

Prioridad de requerimiento	Media
-----------------------------------	--------------

Identificador de requerimiento	RF05 UC-020
Nombre del requerimiento	Uso offline para visualizar tareas
Características	Los usuarios podrán visualizar tareas o actividades cuando no tengan conexión
Descripción del requerimiento	Cada usuario podrá ver sus tareas, solo visualizar cuando no tenga una conexión a internet, esto para si se encuentra desconectado pueda realizar la tarea si no necesita internet.
Entradas	El usuario deberá de contar con una cuenta para acceder al portal académico.
Proceso	El usuario entra a configuración > modo offline > Activar el visualizar tareas Cuando esté en internet, automáticamente la app se abre en modo offline, visualizando las tareas pendientes.
Salidas	Si la descarga de las tareas no se hizo correctamente para el modo offline, Mensaje de error de descarga de tareas.
Prioridad de requerimiento	Media

Identificador de requerimiento	RF05 UC-021
Nombre del requerimiento	Gestión de aula virtual
Características	Los usuarios podrán acceder a los editores externos por medio de enlaces que se vinculen con los editores web de word y excel.
Descripción del requerimiento	Acceder a editores externos.Los usuarios seleccionarán editores externos para editar sus documentos.

Entradas	El usuario deberá de contar con una cuenta para poder acceder al portal académico
Proceso	El usuario seleccionará la opción de más opciones, para poder seleccionar el editor de su preferencia. Cuando el usuario seleccione qué editor desea abrir este se abrirá en una pestaña nueva
Salidas	Si no se pudo abrir el editor, el sistema enviará un mensaje de error Nueva pestaña con el editor abierto.
Prioridad de requerimiento	Baja

Identificador de requerimiento	RF05 UC - 022
Nombre del requerimiento	Gestión de aula virtual.
Características	Los usuarios podrán compartir archivos con otros usuarios.
Descripción del requerimiento	Compartir archivos para trabajar de manera colaborativa. Los usuarios podrán compartir archivos con usuarios ya registrados en la plataforma. Solo los documentos que sean editables y que sean compartidos se podrán trabajar de manera colaborativa.
Entradas	Archivo que se deseé subir a la plataforma para compartir cuente con los siguientes formatos: (.pdf , .doc , .docx , .pptx , .xlsx , .mp4 , .jpg , .png, .css , .cpp , .cs , .txt , .rar , .zip). Añadir usuarios para trabajar de manera colaborativa con: Matrícula (los primeros cuatro números deben ser mayores a 2010 y menores a 20020) Correo electrónico (xxxxx@xxxx.buap.mx) Nombre completo (menor a 30 caracteres)
Proceso	1° El usuario entra a Archivos > Eligen el documento > Compartir > Agregar colaboradores > Listo 2° El usuario entra a Archivos > Crear documento > Elegir el formato > Compartir > Agregar colaboradores > Listo 3° El usuario entra a Archivos > Carga documento > Compartir > Agregar colaboradores > Listo

Salidas	Mensaje de éxito y aviso a los colaboradores para poder trabajar de manera colaborativa. En caso de algún fallo, mensaje de error. En caso de superar los colaboradores un mensaje donde se especifica el exceso de colaboradores.
Prioridad de requerimiento	Media

Identificador de requerimiento	RF06 UC-023
Nombre del requerimiento	Gestión de alumnos
Características	El docente tiene el poder para dar de alta nuevos alumnos al curso en caso de que no estuvieran desde un principio.
Descripción del requerimiento	Registrar alumno: El docente podrá agregar un alumno al curso.
Requerimiento no funcional	El sistema debe notificar al docente lo antes posible si no se encontró la matrícula deseada.
Entradas	Matrícula, correo
Proceso	El docente deberá de ingresar al listado de miembros que se encuentran en dicho curso, dar en el botón agregar, e introducir la matrícula del alumno que desee agregar a su curso.
Salidas	Mensaje de error en caso de no haber encontrado la matrícula. Mensaje de error seguido de la descripción por la cual no se pudo registrar al alumno. Mensaje notificando que el registro fue hecho con éxito.
Prioridad de requerimiento	Medio

Identificador de requerimiento	RF06 UC-024
Nombre del requerimiento	Gestión de alumnos.

Características	El docente tiene el poder para dar de baja al alumno del curso en caso de que así lo requiera.
Descripción del requerimiento	Eliminar alumno: El docente podrá eliminar a un alumno del curso.
Requerimiento no funcional	El sistema debe notificar al docente lo antes posible si no logró eliminar al alumno.
Entradas	Matrícula, acceso al portal
Proceso	El docente , en la sección de miembros del grupo donde se encuentre el alumno a eliminar, presionara en el menú de opciones, presionando eliminar del curso
Salidas	Mensaje de error en caso de no poder eliminar al alumno. Mensaje de éxito en caso de haber podido eliminar al alumno.
Prioridad de requerimiento	Medio

Identificador de requerimiento	RF07 UC-025
Nombre del requerimiento	Gestión de tareas
Características	El usuario con los permisos requeridos podrá gestionar las tareas en el aula virtual.
Descripción del requerimiento	Asignar una tarea.. El usuario podrá asignar una tarea siempre que lo desee siempre que cuente con los permisos de usuario correspondientes.
Requerimiento no funcional	El sistema debe tener una velocidad de respuesta rápida para subir las tareas y debe ser seguro para que nadie sin los permisos correspondientes pueda subir tareas a la materia.
Entradas	Texto con las especificaciones de la tarea, archivo en formato jpeg, pdf, zip, rar o docx.
Proceso	El usuario accederá a la plataforma, después ingresará al curso deseado y le dará click al botón 'Asignar tarea', por lo que el sistema solicita un texto y un archivo, el usuario coloca la información correspondiente y le da click al botón 'Subir tarea'.
Salidas	Mensaje de éxito en el caso de que no exista ningún error.

	<p>Mensaje de archivo en formato incorrecto en el caso de que el archivo no sea compatible.</p> <p>Mensaje de archivo corrupto en el caso de que el archivo esté corrupto o tenga algún virus.</p> <p>Mensaje de error desconocido en caso de que no se identifique el error de primera instancia.</p>
Prioridad de requerimiento	Alta

Identificador de requerimiento	RF07 UC-026
Nombre del requerimiento	Gestión de tareas
Características	El usuario con los permisos requeridos podrá gestionar las tareas.
Descripción del requerimiento	Modificar entrega de una tarea. El usuario podrá modificar una tarea previamente entregada. El usuario podrá modificar la tarea máximo 5 veces, en caso de que haya cumplido ese límite, el sistema deshabilitará la opción de modificar la entrega.
Entradas	Los documentos que se pueden adjuntar en las tareas deberán cumplir con los siguientes formatos: pdf, xlsx, doc, pptx, docx, jpg, jpeg, png, con un tamaño no mayor a 10MB si se adjuntan directamente.
Proceso	El usuario accede a la tarea que desea modificar y seleccionará la opción “modificar tarea”, en pantalla podrá observar las opciones para adjuntar y/o modificar los archivos para posteriormente confirmar la carga de dichos archivos y confirmar la entrega de la tarea.
Salidas	Mensaje de éxito si se concretó la modificación. Mensaje de error en caso de que los archivos adjuntos no cumplan con el formato indicado. Mensaje de error en caso de que los archivos rebasen el tamaño permitido.
Prioridad de requerimiento	Alta

Identificador de requerimiento	RF07 UC-027
Nombre del requerimiento	Gestión de tareas
Características	El usuario con los permisos requeridos podrá gestionar las tareas en el aula virtual.
Descripción del requerimiento	Asignar calificación a tareas.. El usuario podrá ingresar a la plataforma en cualquier momento, y con sus permisos de usuario, será capaz de asignar o modificar el valor de las tareas pendientes a calificar.
Requerimiento no funcional	El sistema debe tener una velocidad de respuesta rápida y debe de tener seguridad suficiente para no permitir el cambio de calificación de un usuario no autorizado.
Entradas	Calificación de la tarea en un rango del 1 al 10, además de una retroalimentación y un archivo opcional en formato pdf, jpeg o docx.
Proceso	El usuario ingresará con sus datos a la plataforma, posteriormente le dará click al curso deseado y a la tarea por calificar; por lo que asignará una calificación y la retroalimentación y va a seleccionar 'Calificar'.
Salidas	Mensaje de éxito al mandar la calificación. Mensaje error desconocido al no poder asignar la calificación.
Prioridad de requerimiento	Alta

Identificador de requerimiento	RF07 UC- 028
Nombre del requerimiento	Gestión de tareas.
Características	Los alumnos tendrán la posibilidad de notificar al docente si algo ha ido mal con la calificación que se le ha dado a una tarea, ya sea un error del profesor, un error administrativo, o una aclaración sobre la nota dada por el docente y un pequeño mensaje para describir la razón.

Descripción del requerimiento	Notificar sobre un error en la calificación. El sistema contará con una opción para que el alumno pueda notificar al docente sobre un error en una calificación de alguna actividad.
Entradas	Una entrega previa. Click en el apartado de “Reportar” Descripción sobre el problema.
Proceso	Una vez hecha una entrega previa, después de que el profesor evalúe la entrega, se habilitará el botón de “Reportar”, el alumno dará click en él, aparecerá un cuadro de texto para añadir la descripción sobre el problema, y el profesor recibirá una notificación sobre un “reporte” junto a la descripción sobre una entrega en específico.
Salidas	Notificación al profesor.
Prioridad de requerimiento	Alta

Identificador de requerimiento	RF07 UC- 029
Nombre del requerimiento	Gestión de tareas.
Características	El usuario con los permisos requeridos podrá gestionar las tareas en el aula virtual.
Descripción del requerimiento	Personalización de actividades. El sistema permitirá a los docentes establecer una actividad para poder manejar los grupos de actividades con un valor igualitario entre cada una dependiendo el tipo de actividad.
Entradas	El usuario crea una actividad y el sistema le da la opción de elegir un perfil para la actividad.
Proceso	El usuario accede al sistema y se dirige a la sección de personalizar tareas, después selecciona las actividades deseadas y les asigna el formato deseado; por último el sistema muestra en pantalla las actividades personalizadas.
Salidas	Mostrar mensaje de éxito en la personalización de la actividad. Mostrar mensaje de error en caso de no concretarse el cambio. El sistema notifica al alumno la actividad asignada.
Prioridad de requerimiento	Media

Identificador de requerimiento	RF07 UC-030
Nombre del requerimiento	Gestión de tareas.
Características	Los alumnos tendrán la posibilidad de notificar al docente si algo ha ido mal con la calificación que se le ha dado a una tarea, ya sea un error del profesor, un error administrativo, o una aclaración sobre la nota dada por el docente y un pequeño mensaje para describir la razón.
Descripción del requerimiento	Modificar calificación de una actividad. El docente podrá modificar la calificación a una actividad siempre y cuando sea reportada por un alumno.
Entradas	Click previo del alumno reportando la actividad. Notificación del sistema al profesor. Descripción del problema. La nueva calificación.
Proceso	El profesor hace click en la notificación lo que lo llevará a la actividad específica que se ha reportado esto marcará a la actividad como “Reportada” para poder verse después sin la notificación. Una vez está en actividad reportada, el profesor podrá ver la descripción. Podrá responder de manera directa al alumno si solo requiere una aclaración, y modificar la calificación. Una vez modificada la calificación, deberá confirmar la acción.
Salidas	Mensaje del profesor al alumno.
Prioridad de requerimiento	Alta

Identificador de requerimiento	RF08 UC-031
Nombre del requerimiento	Gestión de cursos.
Características	Los alumnos tendrán la posibilidad de cambiar la interfaz a unas configuraciones pre-establecidas por el sistema para poder ser más intuitiva para ciertos usuarios.

Descripción del requerimiento	Personalizar perfil de usuario. El sistema permitirá a tanto docentes como alumnos la personalización de su perfil.
Entradas	Click en la configuración. Click en “modificar interfaz” Click en cualquiera de las interfaces disponibles. Confirmación del cambio.
Proceso	El usuario dará click en su perfil, eso desplegará un menú de opciones, después en configuraciones, esto abrirá otro menú, de ahí en interfaz, se dará una visualización de las interfaces disponibles, cuando seleccione una se pedirá una confirmación la cual podrá confirmar o cancelar, si confirma se aplicaran los cambios, si cancela se volverá a la configuración actual.
Salidas	Vista previa de las interfaces. Mensaje de confirmación o cancelación.
Prioridad de requerimiento	Baja.

Identificador de requerimiento	RF08 UC-032
Nombre del requerimiento	Modificar cuenta.
Características	El usuario podrá modificar sus datos con el fin de actualizarlos, con los nuevos datos que el usuario registre en la plataforma
Descripción del requerimiento	Actualizar datos generales del usuario. Los usuarios tendrán la posibilidad de actualizar sus datos, según estos los requieran
Entradas	Correo electrónico, contraseña(alfanumérica de 8 caracteres), nuevos datos por registrar del usuario
Proceso	Los usuarios se colocan en la opción de perfil de usuario, para posteriormente seleccionar configuraciones. El usuario agrega los nuevos datos, para posteriormente seleccionar guardar cambio.
Salidas	Mensaje de éxito en caso de no haber errores en los campos, con los nuevos datos agregados. Mensaje de error en caso de haber errores en algún campo. Mensaje de error en caso de haber dejado algún campo requerido vacío

Prioridad de requerimiento	Baja
----------------------------	------

Identificador de requerimiento	RF08 UC-033
Nombre del requerimiento	Modificar cuenta.
Características	Los usuarios recibirán notificaciones incluso si no tienen abierta la aplicación mientras se encuentre trabajando en segundo plano.
Descripción del requerimiento	Configuración de notificaciones. Cada usuario podrá configurar si quiere que lleguen las notificaciones, aunque tenga la aplicación cerrada y de qué actividades quieren que le lleguen notificaciones.
Entradas	
Proceso	El usuario entra a configuración > notificaciones y en este apartado podrá configurar si desea recibir notificaciones cuando la app se encuentre sin usar (en segundo plano) y elegir cuáles actividades quiere que se le notifique.
Salidas	Mensaje de que se hizo correctamente o si hubo algún error.
Prioridad de requerimiento	Media

Identificador de requerimiento	RF08 UC-034
Nombre del requerimiento	Modificar cuenta
Características	Los usuarios podrán restablecer su contraseña para entrar a la plataforma
Descripción del requerimiento	Restablecer contraseña. Los usuarios podrán restablecer su contraseña por medio del correo electrónico que el usuario proporcionó desde un principio.
Entradas	Correo electrónico(ejemplo alguien@.buap.mx)
Proceso	Los usuarios abrirán el sistema, con el fin de restablecer su contraseña, seleccionarán la opción de restablecer contraseña.

	Una vez seleccionada la opción se le enviará un mensaje al correo electrónico proporcionado con la nueva contraseña.
Salidas	Contraseña nueva.
Prioridad de requerimiento	Baja

Identificador de requerimiento	RF09 UC-035
Nombre del requerimiento	Guía de usuario.
Características	Los usuarios podrán visualizar una guía para el usuario si es que lo desean.
Descripción del requerimiento	Mostrar guía de usuario. Cada uno de los usuario al iniciar por primera vez dentro de la plataforma podrá consultar una guía para cada tipo de usuario
Entradas	Correo electrónico, contraseña
Proceso	EL usuario al ingresar por primera vez, darán click en más opciones, para después seleccionar la opción de MI guía de usuario. Una vez seleccionada la opción se le dará al usuario un pequeña introducción ante las funcionalidades del sistema
Salidas	Guía de usuario
Prioridad de requerimiento	Baja

Requisitos no funcionales

Los requisitos no funcionales definen atributos relacionados con la calidad como rendimiento, escalabilidad, fiabilidad, disponibilidad, mantenimientos, seguridad, etc...

Como requisitos no funcionales consideramos los siguientes:

Requisitos de rendimiento

- Los dispositivos electrónicos donde los usuarios harán uso del sistema, deberán de contar con un navegador web, siendo los navegadores preferentes Google Chrome y Mozilla Firefox.
- El uso simultáneo de este sistema no generará tráfico en la web.
- El sistema responderá de la mejor manera posible ante la concurrencia masiva de usuarios.
- El tiempo de respuesta entre las operaciones que realice el usuario dentro del sistema debe de ser menor o igual a 7 segundos.

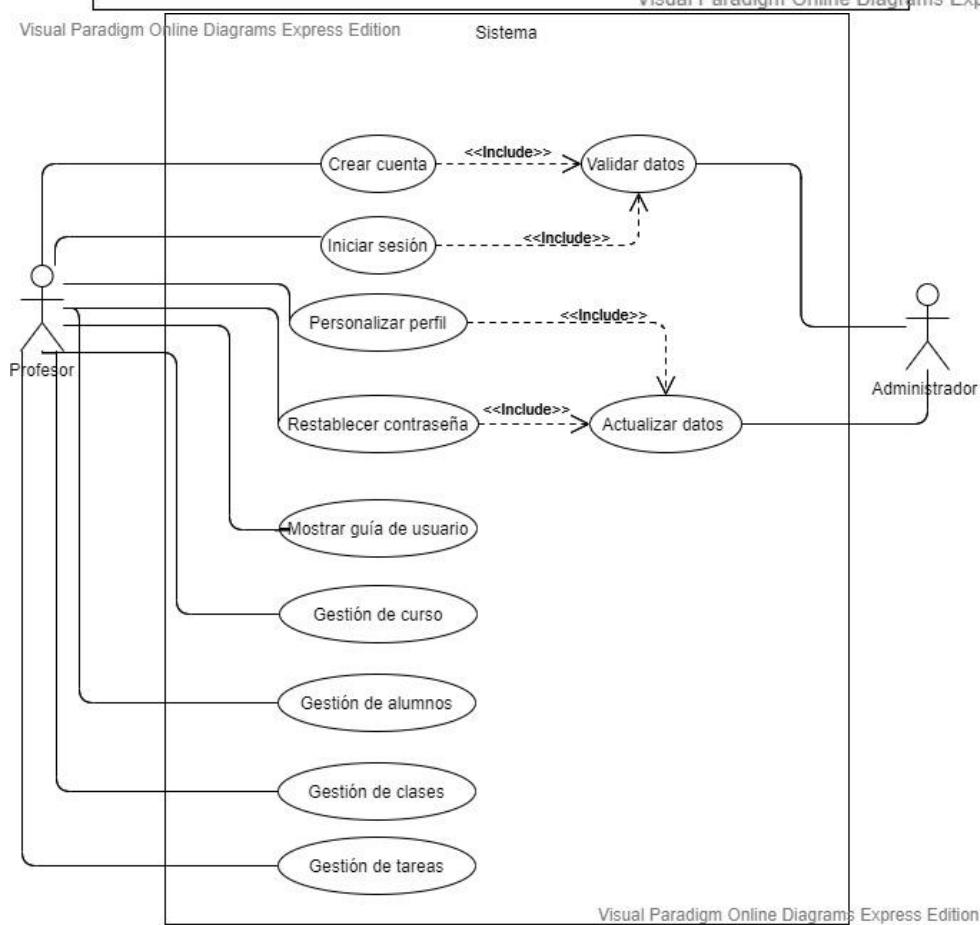
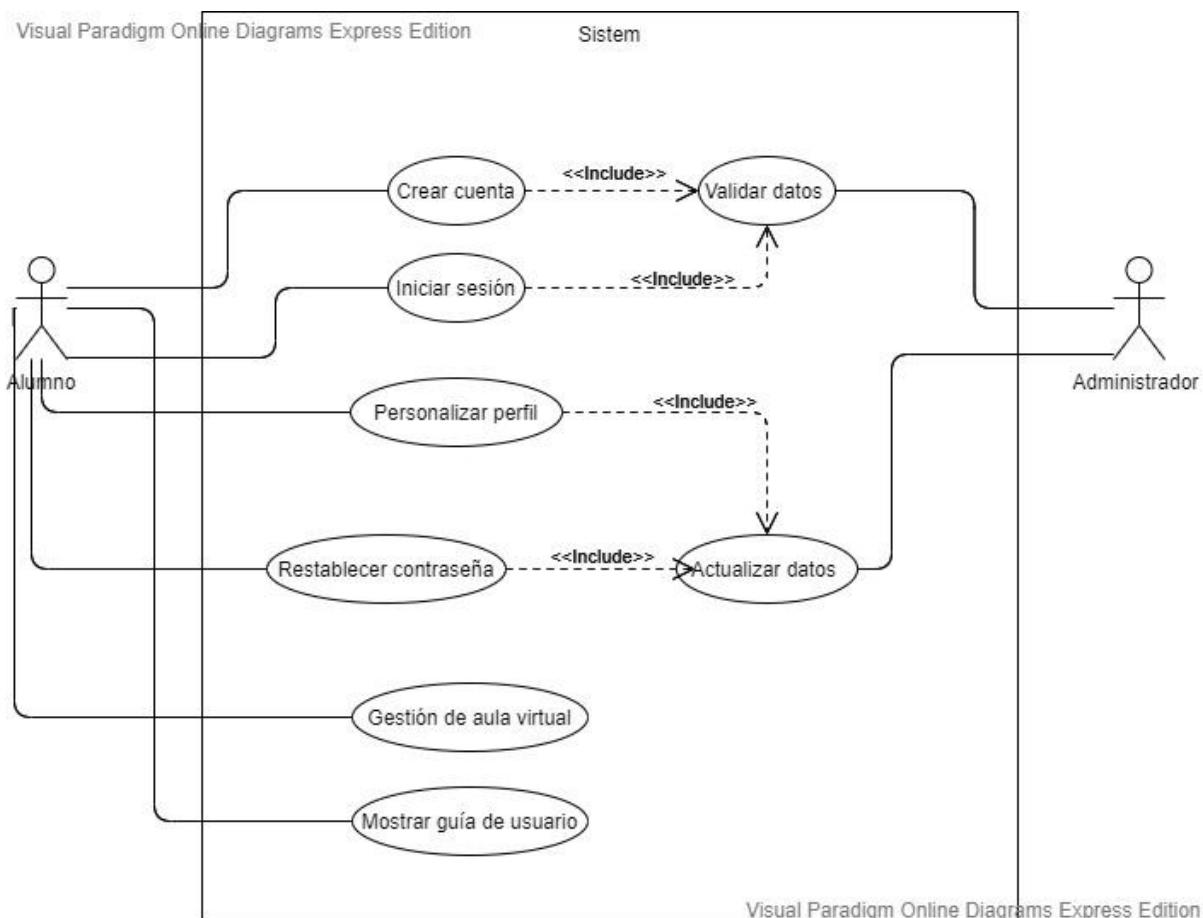
Restricciones de diseño

- El sistema debe de ser intuitivo para los usuarios y fácil de manipular.
- La plataforma deberá ser responsive y adaptarse a dispositivos móviles sin perder usabilidad.

Atributos del sistema

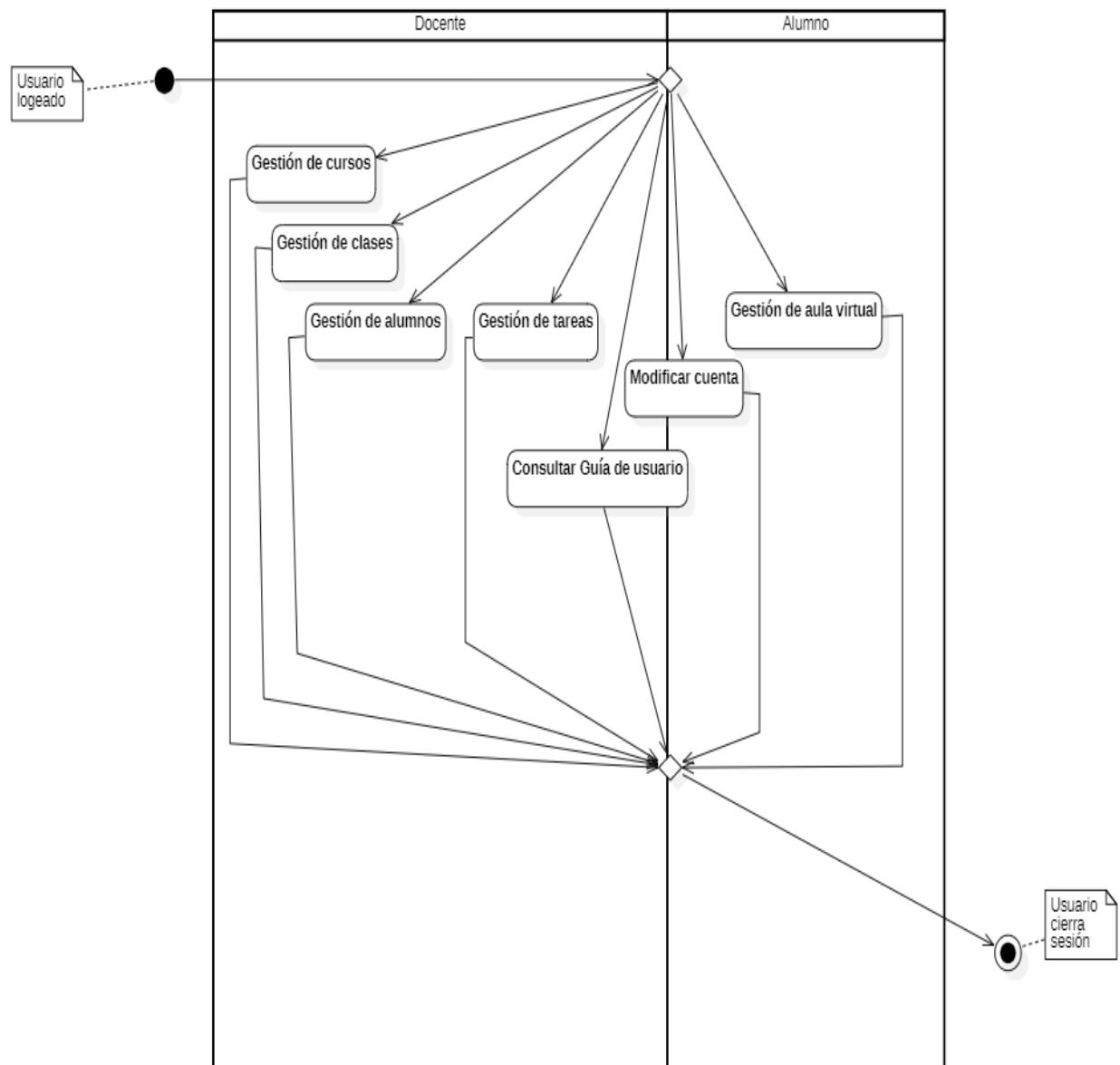
- *Fiabilidad:* Brindar al usuario la suficiente confianza de que el sistema estará en constante mejora, y en caso de algún tipo de error ya sea con la cuenta o con el sistema en general pueda comunicarse con los diseñadores.
El sistema debe de tener una interfaz para los usuarios intuitiva y sencilla, además que debe de ajustarse a las necesidades del sistema.
- *Mantenibilidad:* El sistema cuenta con características parametrizables lo que permitirá futuros mantenimientos y actualizaciones. Estos puntos serán definidos por los diseñadores, quienes estarán verificando las necesidades de los usuarios y actualizando el sistema según estas.
- *Disponibilidad:* La disponibilidad del sistema será de 24 horas, para que los usuarios puedan acceder a ella sin un horario restringido.
- *Seguridad:* El sistema debe garantizar un manejo correcto de los datos personales de los usuarios.

Diagramas de caso de uso



Diseño del proyecto

DIAGRAMA DE ACTIVIDADES GENERAL



El diagrama presentado anteriormente permite tener una vista general del flujo de actividades que se llevarán a cabo en el sistema.

El token inicial es un usuario logueado y previamente autenticado. Los swimlanes delimitan las actividades disponibles para cada actor, alumno o docente, y aquellas que comparten se encuentran entre los límites de los swimlanes. Una vez que el usuario ingresa al sistema va a tener diferentes opciones y según la que elija será el flujo que tomará dicha actividad; una vez terminado ese flujo, el usuario podrá elegir otra opción o cerrar su sesión.

En el Anexo 2 se pueden consultar a detalle los flujos de actividades específicos que complementan el diagrama anterior.

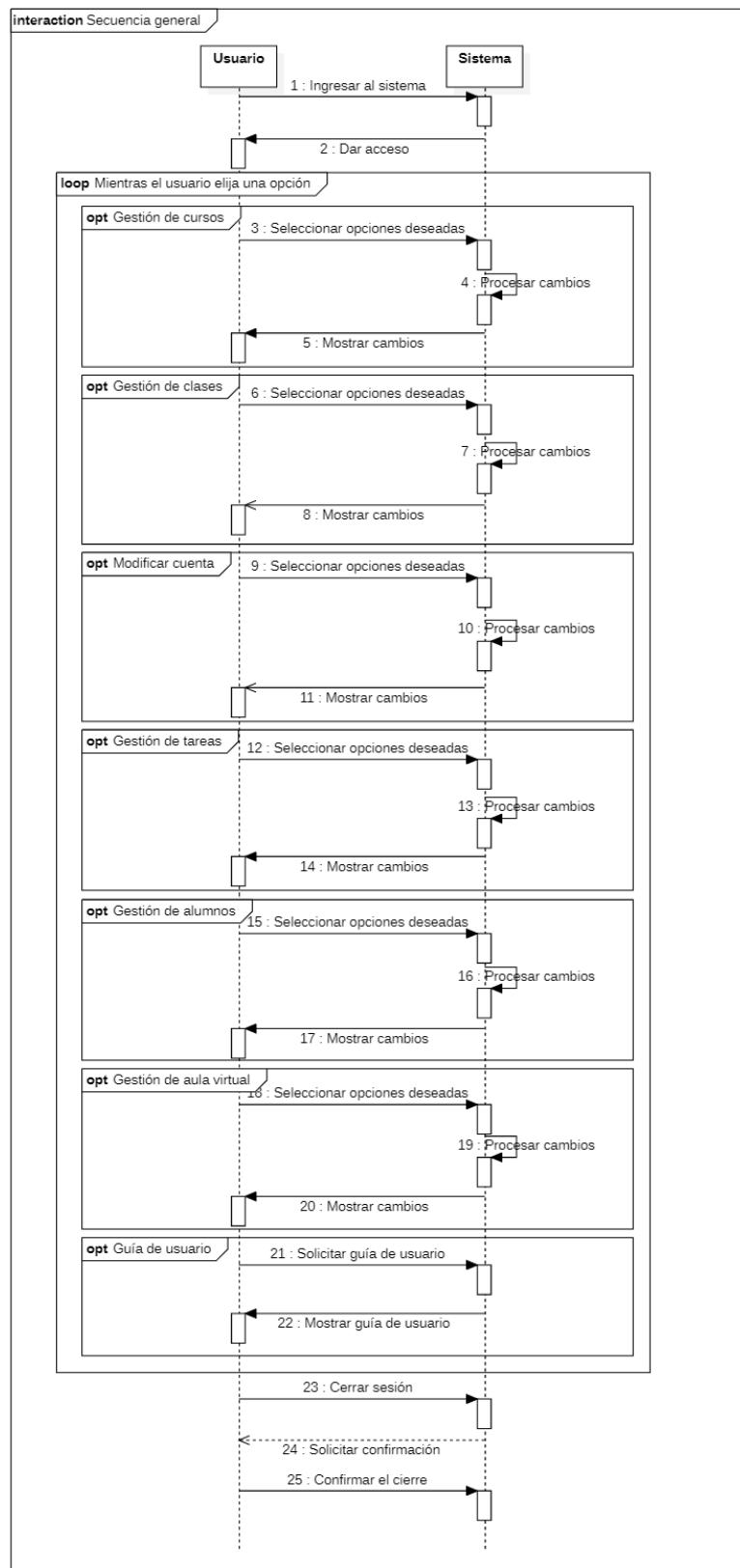
DIAGRAMA DE SECUENCIA GENERAL

Asimismo, se puede observar en el siguiente diagrama de secuencia general la correspondencia de mensajes que existe entre *Usuario* y *Sistema* para describir el comportamiento de los mismos a través de un lapso determinado de tiempo.

Como primer mensaje se tiene el ingreso del usuario al sistema, para lo cual la respuesta es el acceso al sistema, después el usuario elegirá una opción deseada para la cual el sistema responderá de manera específica.

Mientras el usuario siga eligiendo opciones, el sistema responderá ante dichas peticiones y el flujo terminará hasta que el usuario cierre sesión.

Para entender mejor la secuencia propia de cada opción disponible, se pueden consultar los diagramas de secuencia incluidos en el Anexo 2.



Patrones de diseño

Tomando en cuenta la información consultada, encontramos en los patrones de diseño una forma de optimizar las soluciones a los problemas particulares de nuestro proyecto entendiendo que estas herramientas sirven como guía y se tendrán que adaptar estos patrones según las necesidades del sistema.

Dentro de los patrones de comportamiento que se emplearán en este proyecto se encuentran Strategy y Observer. A continuación, describimos la forma en la que adaptamos dichos patrones como parte de la etapa de diseño en el desarrollo del sistema.

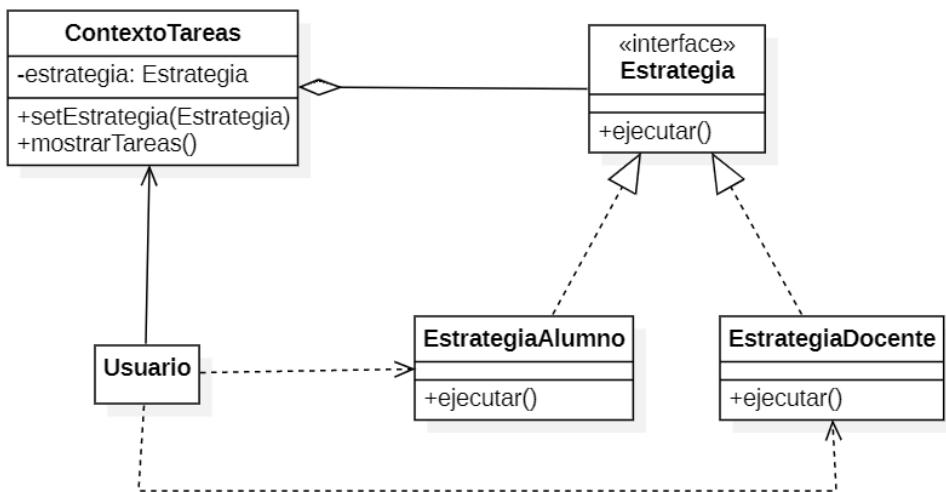
STRATEGY

Si contamos con una clase que hace algo específico pero de formas diversas, entonces podemos adaptar esos algoritmos especializados en clases separadas que llamaremos “estrategias”.

La clase original, conocida como contexto, va a tener un atributo que guarde la referencia a alguna de las estrategias disponibles. Esta clase va a delegar las tareas a un objeto de estrategia en vez de hacerlo por su cuenta.

La clase contexto funciona con las estrategias disponibles a través de una interfaz genérica que contará con un único método disparador del algoritmo especializado encontrado en una estrategia en particular.

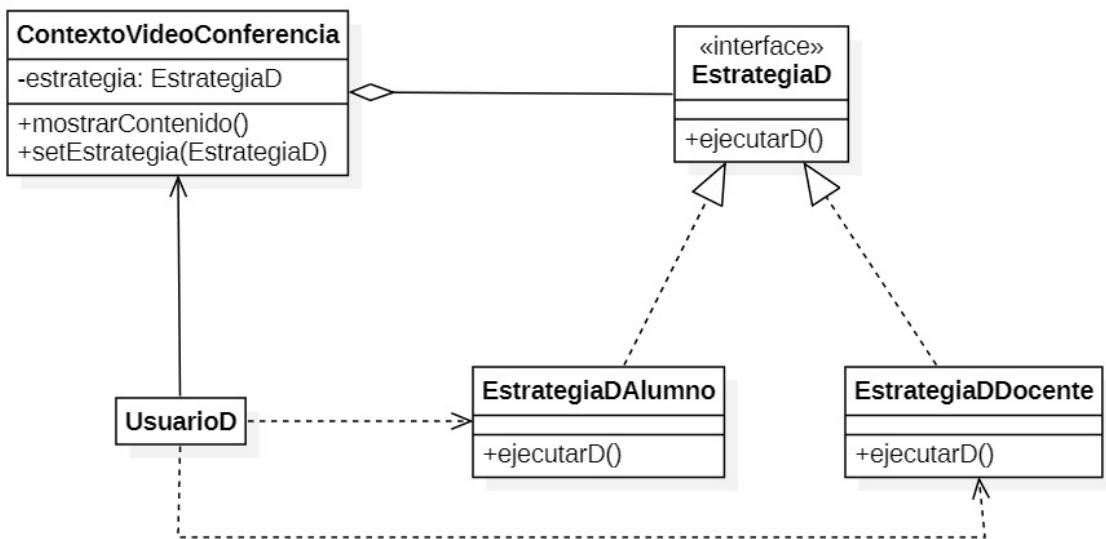
Con lo anterior, permitimos que la clase original sea independiente y cualquier modificación que se requiera en los algoritmos existentes, o incluso si se quieren agregar otros, no afectarán el código de la clase contexto ni de las estrategias ya definidas.



En la imagen anterior podemos observar una aplicación de este patrón para el funcionamiento de la plataforma. Se presenta una solución al caso en el que los usuarios del sistema quieran consultar la sección de Tareas. Cada actor, alumno o docente, podrán ver en el sistema opciones distintas debido a los permisos que se les otorga desde que generan su cuenta y se autentica su información.

La interfaz genérica “estrategia” cuenta con un método ejecutar, este método será implementado por “EstrategiaAlumno” y “EstrategiaDocente”, es en esta implementación donde se especializa el algoritmo.

Por último, el usuario se relaciona directamente con la clase “ContextoTareas” y a través de una instancia de esta clase es como podrá asignar la estrategia deseada y usar el método disponible “mostrarTareas()”.



Dentro de la anterior imagen, podemos observar un funcionamiento muy parecido a la estrategia utilizada para las tareas; esto debido a que pudimos identificar un caso similar para las videoconferencias, y es que en el diseño de nuestro sistema proponemos que el docente obtenga más privilegios al momento de crear y utilizar una videollamada de lo que el alumno obtiene, de esta manera las estrategias de alumno y docente se relacionan con la interfaz, la comunicación del usuario va hacia la clase de **ContextoVideoConferencia** donde se contendrá una de las dos estrategias a elegir que a su vez tienen el método de ejecutar; de esta forma las opciones en pantalla que se mostrarán al profesor van a aumentar, dando la posibilidad de grabar la clase y finalizar la clase, cosa que los alumnos tienen restringido.

Como el diseño anterior, se podrán consultar el resto de las estrategias en el Anexo 3; además, la definición e implementación de las clases en código C++ está disponible en el repositorio de Github cuyo link se encuentra en el Anexo 4.

OBSERVER

Es un patrón de diseño de comportamiento que te permite definir un mecanismo de suscripción para notificar a varios objetos sobre cualquier evento que le suceda al objeto el cual se está observando.

Los objetivos que se buscan lograr, haciendo uso de este patrón de diseño son:

- Reducción de acoplamiento entre las clases a las que pertenecen los object subjects y observers para poder tener mayor capacidad de reusabilidad de ellas.
- Permitir un número limitado de observers a los subjects.
- Modificación de clases subjects y observers independientemente.
- Permitir que dos capas de abstracción se puedan comunicar entre sí sin perder dicha abstracción.
- El poder agregar o eliminar observadores sin problema alguno.
- Mandar información a varios objetos de manera eficiente.

El patrón observer también cuenta con desventajas:

- Para crear una clase personal de observer, no es posible, tiene que ser forzosamente de una subclase.
- La interfaz de observer, forzosamente tiene que ser implementada por una “ConcreteObserver” para su funcionamiento.
- Si se implementa de forma errónea, puede conllevar a complejidad y dar a un error de funcionamiento sin advertencia.

DECORATOR

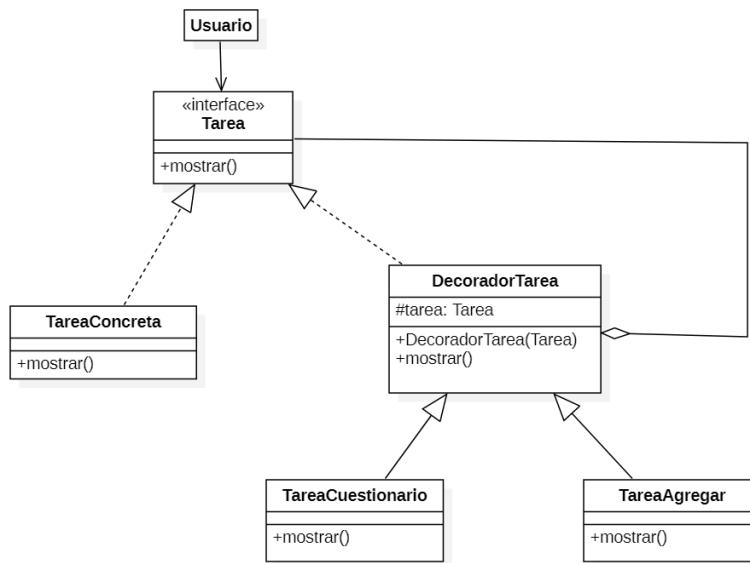
Adicionalmente, uno de los patrones de diseño estructurales disponibles es el Decorator. Este patrón tiene como finalidad principal añadir funcionalidades a los objetos colocándolos dentro de otros objetos encapsuladores.

Normalmente cuando se quiere agregar funcionalidad a un objeto, se busca extenderlo, sin embargo, la herencia tiene las siguientes limitantes:

- Es estática. Si se quiere cambiar funcionalidad en tiempo de ejecución se debe sustituir el objeto completo por otro creado con esas funcionalidades deseadas.
- Las subclases sólo pueden heredar de una clase.

Con la solución Decorator se pueden enfrentar estas limitantes al poder asignar funcionalidades extra a objetos en tiempo de ejecución sin alterar el código que emplean esos objetos.

Observemos a continuación un ejemplo de este patrón enfocado en este proyecto.



Se parte de la idea de que una tarea puede tener adicionalmente la opción de agregar un documento, de tener un cuestionario o incluso ambas.

En el diagrama se representan dos decoradores concretos que van a agregar funcionalidad al objeto “tarea”. Los objetos “tareaconcreta” y “decoradortarea” implementan la misma interfaz; a su vez, “tareaCuestionario” y “tareaAgregar” extienden la funcionalidad de “decoradortarea”. El decorador base (decoradortarea) hace referencia a “tarea” y los decoradores concretos se encargan de sobreescribir los métodos del decorador base y ejecutar su comportamiento.

A través de una instancia de “tarea” se podrá acceder al método disponible mostrar(), el resultado dependerá de si la instancia emplea o no los decoradores disponibles.

Para consultar los diagramas Decorator dirigirse al Anexo 3; además, la definición e implementación de las clases en código C++ está disponible en el repositorio de Github cuyo link se encuentra en el Anexo 4.

FACTORY

El patrón de diseño factory method tiene como función plantear el problema de la creación de objetos sin especificar la clase exacta a la que pertenece y sin tener que acceder a su lógica.

Nos ofrece una interfaz de la cual podemos crear objetos a partir de cada clase, dándonos como oportunidad que a medida que la interfaz los va creando, no se presenten errores de producto que a medida tengan repercusión.

Este patrón cuenta con las siguientes ventajas :

Las ventajas que provee este patrón son las siguientes:

- Elimina la necesidad de instanciar de forma explícita los objetos a utilizar.
- Encapsula en todas las clases Factory la lógica de la creación de los objetos.
- Es fácil y extensible ya que la arquitectura queda abierta a nuevos desarrollos con nuevas clases que puedan extenderse a la Factory.
- Los objetos obtenidos a partir de la fábrica son compatibles.
- Principio de responsabilidad unica
- Principio abierto/Cerrado.

Algunas de sus desventajas son :

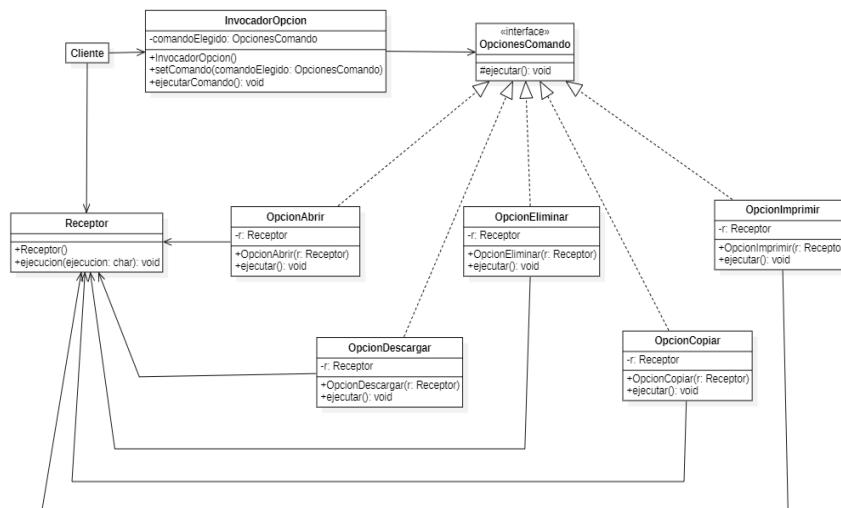
- Pueden emplearse indebidamente ya que cuenta con operaciones frecuentes para un programador.
- Cuando se delegan las funciones se puede volver más complejo encontrar en primera instancia la mecánica del funcionamiento de la aplicación.

Para consultar los diagramas implementados dirigirse al Anexo 3; además, la definición e implementación de las clases en código C++ está disponible en el repositorio de Github cuyo link se encuentra en el Anexo 4.

COMMAND

El siguiente patrón de diseño a tomar en cuenta es Command, el cual nos va a permitir convertir una solicitud en un objeto independiente que va a contener toda la información sobre una solicitud. Este patrón tiene algunos usos como poner operaciones en cola, programar la ejecución de una operación o incluso programar de forma remota, tal como sucede en un control de televisión.

Como cualquier patrón existen ventajas y desventajas, dentro de las ventajas encontramos la opción de ensamblar un grupo de comandos simples para la creación de uno complejo, también poder desacoplar clases que invocan operaciones, entre otras; sin embargo hay que tomar en cuenta que el código inevitablemente se complica ya que se introducen capas entre emisores y receptores, por tanto si se busca sencillez o un código entendible para muchas personas probablemente no sea el patrón de diseño ideal.



En la imagen anterior pudimos observar la aplicación de Command en las opciones de comportamiento del contenido disponible en cualquier materia dentro de la plataforma. El cliente realmente se va a comunicar con la clase invocadora, de hecho nunca puede ver al receptor, sin embargo se encuentran relacionados debido a que el producto final o el objetivo principal es que el cliente obtenga algo del receptor, incluso aunque no hable directamente con él.

El invocador inicializa las solicitudes, además de almacenar una referencia a un objeto de comando. La interfaz Opciones Comando únicamente tiene un método de ejecución, de las cuales las opciones específicas van a heredar e implementar de una manera independiente, estas opciones son poder abrir un documento, eliminarlo, imprimirla, copiarlo y descargarlo.

El receptor es el que realiza el trabajo pesado y es que los comandos específicos no realizan nada más que darle los detalles de la ejecución al receptor para que sepa que es lo que debe de hacer y culmine su trabajo.

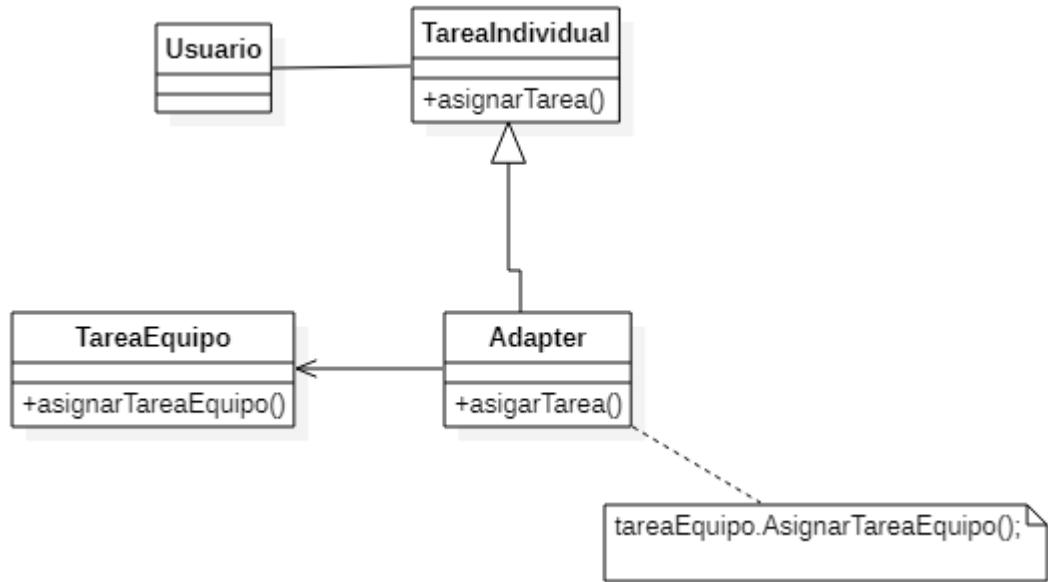
Para consultar los diagramas implementados dirigirse al Anexo 3; además, la definición e implementación de las clases en código C++ está disponible en el repositorio de Github cuyo link se encuentra en el Anexo 4.

ADAPTER

El patrón de diseño *adapter*, pertenece al grupo de patrones estructurales, siendo el objetivo de estos gestionar cómo se combinan las clases y los objetos para dar lugar a estructuras más complejas.

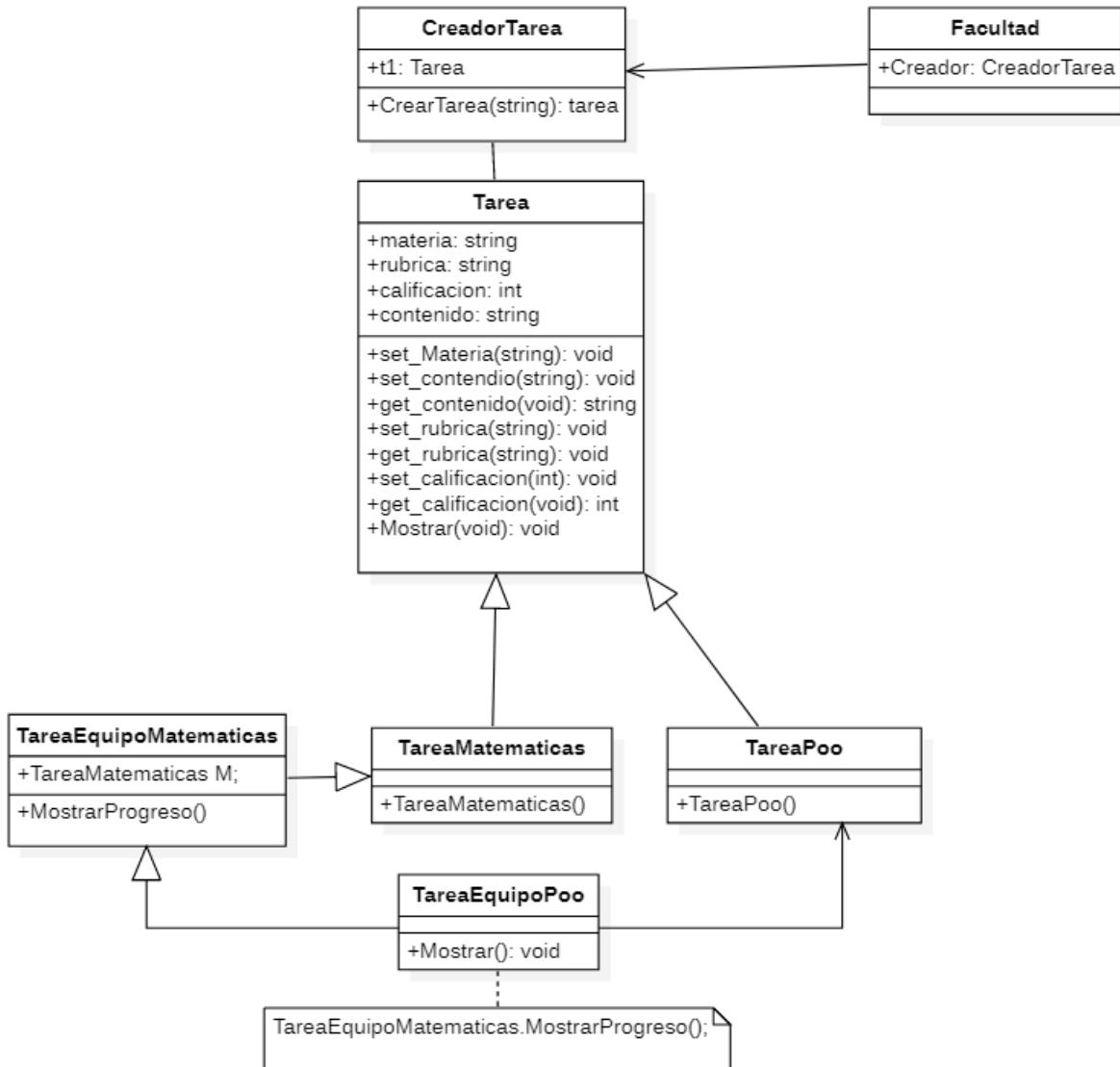
Este patrón es útil cuando se tienen interfaces de software incompatibles, las cuales a pesar de su incompatibilidad tienen una funcionalidad similar, donde la clase adaptador será la encargada de proporcionar los métodos para interactuar con la interfaz no compatible.

Estos se dividen en patrones orientados a clases y patrones orientados a objetos, en este apartado nos orientaremos a las clases, como se muestra en la siguiente imagen.



La clase que se quiere adaptar es la clase `tareaEquipo`, pero cuenta con un método aunque similar no hace la misma función que si se asigna una tarea de forma individual, por ello se crea una clase adaptador que cumplirá con la función de crear un método que se adapte a la funcionalidad de la clase a la cual se quiere adaptar.

Para ver la implementación del código en C++ de este diagrama visite el enlace a repositorio de Github, disponible en el Anexo 3, además si desea ver los demás diagramas de este patrón de diseño está disponible en el Anexo 4.



Por esta parte queremos adaptar la clase **TareaPoo** para que pueda fungir como la clase **TareaEquipoMatematicas** y de esta forma pueda comportarse de manera similar a como lo hace esta última.

Toda la implementación en C++ de esta en el repositorio de Github, disponible en el Anexo 3, además si desea ver los demás diagramas de este patrón de diseño esta disponible en el Anexo 4.

Verificación y validación

Las pruebas tienen como objetivo principal revisar que el programa funcione como se desea, además de encontrar posibles defectos antes de que se utilice.

- Validación: Se debe validar si el programa cumple con los requisitos del usuario.
- Verificación: Corroborar que el programa funcione adecuadamente cumpliendo los requerimientos no funcionales antes determinados.

Después de analizar las diversas pruebas que existen para la validación y verificación del software, se propone el siguiente plan de pruebas que podría ser aplicado para cumplir con esta etapa en la plataforma a desarrollar.

Plan de pruebas de nuestro proyecto	
Clasificación	Técnicas específicas
Técnicas basadas en la especificación	Partición de equivalencia
	Pruebas de robustez
	Tablas de decisión
Técnicas estadísticas	Prueba de sala limpia
Técnicas basadas en el uso	Pruebas de fiabilidad del software

Se dará una breve descripción de cada prueba y de cómo se pretende aplicarla en el proyecto:

- *Partición de equivalencia:* Se clasifican los datos de entrada que se comportan de manera similar, agrupándolos en particiones de equivalencia. Al tomar un caso de una partición se le asigna el nombre de “caso típico”.

Dentro de nuestro proyecto, uno de los casos típicos a considerar es Abrir un documento, ya que no se pueden probar cada uno de los documentos posibles, entonces se opta por clasificar esa entrada en una partición de equivalencia.

- *Pruebas de robustez:* Este tipo de pruebas son de suma importancia ya que van a permitir al software tener una reacción adecuada ante cualquier entrada del usuario, ya sea correcta o incorrecta.

El ingreso a la plataforma deberá ser probado para evitar que las entradas incorrectas de datos ocasionen una falla en el sistema. Lo ideal es que el software notifique al usuario del error y permita modificar los datos incorrectos.

- *Tablas de decisión:* Serán de utilidad para aquellos casos de prueba en los que se puedan aplicar condiciones lógicas que generen acciones específicas para después evaluar nuevos casos con más condiciones.

Por ejemplo, para verificar un dato este puede ser condicionado recurrentemente para su validación, por lo tanto, se tendrá que probar que el sistema no tenga errores frente a estas condiciones.

- *Técnica de sala limpia:* Tiene como objetivo conseguir un control estadístico del número de fallos, tomando en cuenta diversos usos del software para poder separarlo en casos de prueba.

En nuestro software se tomarán en cuenta los usos de la plataforma, tales como hacer videollamada, acceder a una clase, subir archivos, etc. Para poder registrar los posibles fallos.

- *Técnica de fiabilidad:* Pruebas que se hacen durante todo el proceso de desarrollo para asegurar el correcto funcionamiento del software.

Estas pruebas deberían ser aplicadas a la par de la implementación del código para evitar a toda costa fallos en el sistema.

Pruebas según su objeto

- *PRUEBAS DE UNIDAD:* consiste en evaluar las partes individuales del software para verificar que cumple con la funcionalidad establecida.

En nuestro software, se dividiría el programa en módulos específicos como: iniciar sesión, videoconferencia , tareas, mensajes y contenido. Estos módulos serían verificados y cualquier fallo detectado sería registrado.

- *PRUEBAS DE INTEGRACIÓN:* En esta prueba se corrobora el funcionamiento de los módulos trabajando en conjunto.

En el caso particular de la plataforma escolar, estas pruebas se aplicarán en la interfaz en la cual se pueden visualizar todos los módulos. Un posible fallo podría ser que al seleccionar un módulo específico(mensajes), la interfaz no se actualice de manera adecuadas.

De los tipos de pruebas de integración, consideramos que la integración ascendente sería la más adecuada para aplicar en nuestro software, ya que comienza por los componentes de menor nivel, para después integrar los que invocan a los anteriores, y así sucesivamente.

- *PRUEBAS DEL SISTEMA.* Se componen por las siguientes pruebas:
 - Pruebas de funcionalidad y operativa: Estas se realizarían con el objetivo de ver que los requisitos se cumplan de manera adecuada según lo que el cliente solicita. Serían pruebas realizadas por personas ajenas al desarrollo del software (pruebas de caja negra), en este caso particular, diferentes usuarios interesados en el uso de la plataforma.

- Pruebas de rendimiento
 - a. De desgaste: se evaluará la plataforma cuando alcance o supere el número máximo de usuarios conectados.
 - b. De volumen: se comprobará el comportamiento de la plataforma cuando se suban múltiples archivos con alto volumen de datos.
 - c. De configuración: se probará la plataforma en diferentes sistemas operativos y navegadores web.
- Pruebas de aceptación: Estas pruebas se realizan directamente con el cliente para que se apruebe el correcto funcionamiento de la plataforma.
 - a. Pruebas piloto y de instalación: consistirá en subir la plataforma a un hosting para someterla a un trabajo diario promedio y verificar los fallos que pudieran existir.
- Pruebas de usabilidad: Son pruebas muy importantes para la plataforma que se está desarrollando ya que se pretende que el uso de la misma sea muy sencillo e intuitivo. Se espera que el tiempo de aprendizaje y el porcentaje de errores de los usuarios sea el mínimo.

Estas pruebas arrojarán datos estadísticos con los cuales se podrá verificar qué nivel de usabilidad tiene la plataforma.

- Compatibilidad: Se someterá el software a diversas pruebas para comprobar el funcionamiento correcto de la integración de éste con aplicaciones externas, tales como editores y visores de documentos, drivers de sonido y video, entre otros.

Diseño de las pruebas

Se muestran a continuación algunos ejemplos de diseño de casos prueba que puedan funcionar como estructura inicial para la validación de la plataforma.

HISTORIA DE USUARIO 1.1- Como usuario debería poder registrarme en el sistema para tener una cuenta de acceso única al mismo.

CASO PRUEBA 1- El sistema pide los datos personales, un correo institucional y una contraseña segura al usuario. (la contraseña deberá tener 8 caracteres alfanuméricos, al menos una mayúscula y un número).

Pruebas para el nombre y apellidos del usuario	
Entradas válidas	Cadena de caracteres con longitud mínima de 3 y máxima de 40.
Entradas no válidas	Caracteres no alfanuméricos. Cadena de caracteres mayor a 40. Cadena de caracteres menor a 3.
Una prueba exactamente en el límite inferior	Longitud = 3
Una prueba exactamente en el límite inferior -1	Longitud = 2
Una prueba exactamente en el límite superior	Longitud = 40
Una prueba exactamente en el límite superior +1	Longitud = 41
Rango	2 < Longitud < 41 (Válida); 3 > Longitud Longitud > 40 (No válida)
Obligatoriedad	Es obligatorio el campo. En caso de campo vacío se muestra Mensaje de error “El campo es obligatorio”. Referencia Nula no permitida

Pruebas para el correo del usuario

Entradas válidas	Cadena de caracteres alfanuméricos con longitud mínima de 20 y máxima de 60. La cadena deberá contener el siguiente formato: xxxxxxxx@correo.buap.mx xxxxxxxx@alumno.buap.mx
Entradas no válidas	Cadena de caracteres mayor a 60. Cadena de caracteres menor a 20.
Una prueba exactamente en el límite inferior	Longitud = 20
Una prueba exactamente en el límite inferior -1	Longitud = 19
Una prueba exactamente en el límite superior	Longitud = 60
Una prueba exactamente en el límite superior +1	Longitud = 61
Rango	19<Longitud<61 (Válida); 20>Longitud Longitud > 60 (No válida)
Obligatoriedad	Es obligatorio el campo. En caso de campo vacío se muestra Mensaje de error “El campo es obligatorio”. Referencia Nula no permitida

Pruebas para la contraseña	
Entradas válidas	Cadena de caracteres alfanuméricos con longitud mínima de 8 y máxima de 20. La cadena deberá contener al menos una mayúscula y un número.
Entradas no válidas	Cadena de caracteres mayor a 20. Cadena de caracteres menor a 8.
Una prueba exactamente en el límite inferior	Longitud = 8
Una prueba exactamente en el límite inferior -1	Longitud = 7
Una prueba exactamente en el límite superior	Longitud = 20
Una prueba exactamente en el límite superior +1	Longitud = 21
Rango	7<Longitud<21 (Válida); 8>Longitud Longitud > 20 (No válida)
Obligatoriedad	Es obligatorio el campo. En caso de campo vacío se muestra Mensaje de error “El campo es obligatorio”. Referencia Nula no permitida

Registro de datos de usuario	CP-01
<i>Descripción:</i>	
El sistema pide los datos personales del usuario, un correo institucional y una contraseña segura al usuario. (la contraseña deberá tener 8 caracteres alfanuméricos, al menos una mayúscula y un número.)	
<i>Prerrequisitos:</i>	
El ingreso de nombre, apellidos, correo institucional y contraseña.	
<i>Resultado esperado:</i>	
El sistema responde adecuadamente ante la entrada de datos inválidos, notifica al usuario y le permite modificar los datos.	
El sistema muestra una señal de aceptación al usuario cuando sus datos cubren con las especificaciones señaladas.	
El sistema no permite el registro de datos que no cumplan con las especificaciones requeridas.	
El sistema permite al usuario cancelar el registro en caso de ser necesario.	
El funcionamiento del sistema no se ve afectado ante la continua entrada de datos inválidos.	
<i>Resultado obtenido:</i>	

HISTORIA DE USUARIO 2.1- Como usuario debería poder adjuntar archivos en una tarea asignada.

CASO PRUEBA 2- El sistema permite seleccionar el archivo a adjuntar desde el dispositivo usado para su posterior carga. Los archivos permitidos por el sistema deberán tener cualquiera de las siguientes extensiones: .pdf , .doc , .docx , .pptx , .xlsx , .mp4 , .jpg , .png , .html , .css , .cpp , .cs , .txt , .rar , .zip.

Pruebas para adjuntar archivos	
Entradas válidas	Archivos con las siguientes extensiones: .pdf , .doc , .docx , .pptx , .xlsx , .mp4 , .jpg , .png , .html , .css , .cpp , .cs , .txt , .rar , .zip; con tamaño no mayor a 2GB por archivo.
Entradas no válidas	Archivos con formatos no válidos.
Una prueba exactamente en el límite inferior	Tamaño = 0 bytes
Una prueba exactamente en el límite superior	Tamaño = 2 GB
Una prueba exactamente en el límite superior +1	Tamaño = 2.1GB
Rango	0 bytes<=Tamaño< 2.1GB (Válida); Tamaño > 2GB (No válida)
Obligatoriedad	No es obligatorio

Adjuntar un archivo	CP-02
<i>Descripción:</i>	
El sistema permite seleccionar el archivo a adjuntar desde el dispositivo usado para su posterior carga. Los archivos permitidos por el sistema deberán tener cualquiera de las siguientes extensiones: .pdf , .doc , .docx , .pptx , .xlsx , .mp4 , .jpg , .png , .html , .css , .cpp , .cs , .txt , .rar , .zip	
<i>Prerrequisitos:</i>	
El usuario deberá tener una tarea asignada que permita adjuntar archivos.	
<i>Resultado esperado:</i>	
El sistema responde adecuadamente ante la selección de archivos inválidos, le notifica al usuario y le permite elegir otro archivo.	
El sistema muestra una señal de aceptación al usuario cuando los archivos cumplen con el formato adecuado.	
El sistema no permite adjuntar archivos que no cumplan con las especificaciones requeridas.	
El funcionamiento del sistema no se ve afectado ante un continuo intento de adjuntar archivos que no cumplen con el formato establecido.	
<i>Resultado obtenido:</i>	

HISTORIA DE USUARIO 3.1 – Como usuario debería poder enviar mensajes en los chats del sistema.

CASO PRUEBA 3- El sistema permite el envío de mensajes al usuario. Los mensajes deberán tener al menos un carácter y no superar un tamaño de 24 kb por mensaje.

Pruebas para enviar mensajes en el chat	
Entradas válidas	Mensajes con tamaño mínimo de un byte y máximo de 24 kb
Entradas no válidas	Mensajes con tamaño menor a 1 byte Mensajes con tamaño mayor a 24 kb
Una prueba exactamente en el límite inferior	Tamaño = 1 byte
Una prueba exactamente en el límite inferior -1	Tamaño = 0 byte
Una prueba exactamente en el límite superior	Tamaño = 24 kb
Una prueba exactamente en el límite superior +1	Tamaño = 24.1kb
Rango	0 bytes Tamaño < 24.1kb (Válida); 0 > Tamaño Tamaño > 24kb (No válida)
Obligatoriedad	Es obligatorio al menor escribir un carácter. En caso de que el campo esté vacío, se muestra mensaje “Escribe un mensaje para continuar”

Envío de mensajes en un chat	CP-03
<p><i>Descripción:</i></p> <p>El sistema permite el envío de mensajes al usuario. Los mensajes deberán tener al menos un carácter y no superar un tamaño de 24 kb por mensaje.</p>	
<p><i>Prerrequisitos:</i></p> <p>El usuario habrá seleccionado un chat específico para enviar mensajes.</p>	
<p><i>Resultado esperado:</i></p> <p>El sistema responde adecuadamente ante la entrada de mensajes inválidos, notifica al usuario y le permite modificar el mensaje.</p> <p>El sistema no permite la entrada de mensajes que no cumplan con el tamaño establecido.</p> <p>El funcionamiento del sistema no se ve afectado ante la continua entrada de mensajes que estén fuera de los límites de tamaño.</p>	
<p><i>Resultado obtenido:</i></p>	

TRAZABILIDAD DE CASOS DE PRUEBAS- REQUISITOS

Esta tabla permite indicar la correspondencia que existe entre los casos prueba y los requisitos funcionales del sistema.

	RF-01	RF-02	RF-03	RF-04	RF-05	RF-06	RF-07	RF-08	RF-09
CP-01	X	X							
CP-02					X				
CP-03					X				

Mantenimiento

Es la modificación de un producto de software después de la entrega, los principales cambios del software son:

Cambios adaptativos: Volver el software adaptativo a las diferentes condiciones, este tipo de mantenimiento implica desde pequeños retoques hasta una reescritura de todo el código.

Cambios correctivos: Se enfocan en localizar y eliminar los posibles defectos de los programas.

Los cambios correctivos que se logran cubrir en el proyecto está en el uso del manual de usuario y lograr tiempos de respuesta cortos.

El manual de usuario fue hecho porque se piensa ayudar a los usuarios en algún problema que se les pueda presentar con una guía lo más amigable e interactiva posible.

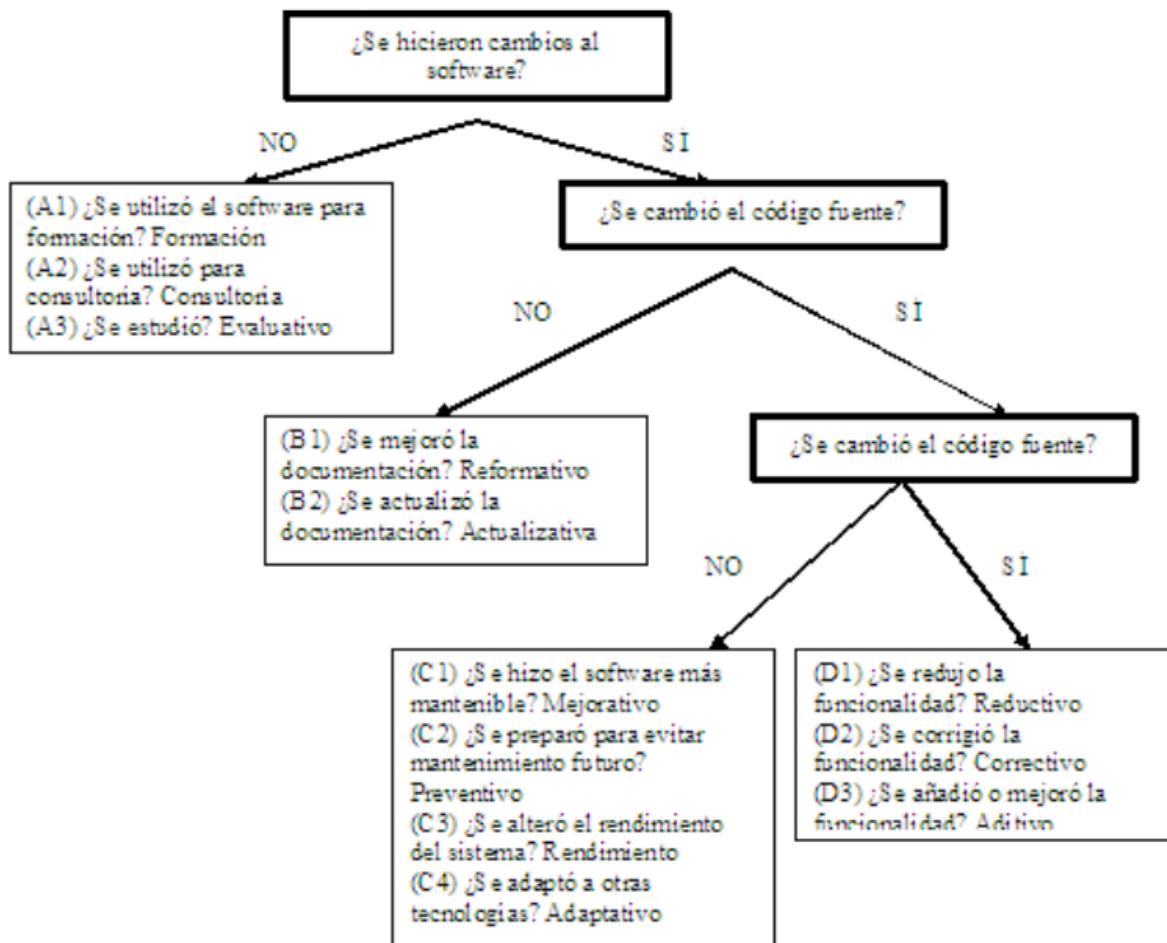
Cambios perfectivos: Mejorar o añadir nuevas funcionalidades que hagan más eficiente el rendimiento del software.

Cambios preventivos: Detener el deterioro y aumentar la capacidad de mantenimiento.

Para lograr este cambio en el proyecto sería tomar sugerencias de los usuarios para tomar en cuenta las necesidades o aspectos que hacen falta y que son esenciales para ellos.

Las diferencias de estos cambios se pueden observar de una manera más sencilla a través de la siguiente tabla y mapa con los cambios de código o cambios de funcionalidad.

	Corrección	Mejora
Proactiva	Preventivo	Perfectivo
Reactiva	Correctivo	Adaptativo



Recuperados de: <https://cnx.org/contents/uuiMsUHT@3/Tipos-de-Mantenimiento>

Conclusiones

Después de la investigación realizada podemos notar que es indispensable entender no sólo qué es y cómo funciona un software, sino los distintos tipos que existen y la finalidad que tiene cada uno de ellos.

Entendiendo de forma general un software como un programa o conjunto de instrucciones que nos permiten tener “interacción” con la computadora, nos queda claro que día con día crece más la necesidad de contar con especialistas en este campo, ya que las computadoras se están convirtiendo en herramientas básicas tanto escolar como profesionalmente.

Aprendimos, además, que la creación de software debe ser organizada y seguir protocolos establecidos que, si bien no garantizan la falta de fallas en el proceso, al menos permiten evitarlas y mejorar considerablemente los tiempos y los costos del desarrollo. Todo lo anterior nos lo brinda la ingeniería de software que con el tiempo ha cobrado más importancia en nuestro medio y que permite que los nuevos desarrolladores cuenten con las habilidades y destrezas necesarias para cumplir con los requerimientos de cualquier proyecto de software.

De lo anterior, consideramos que el software que se necesita para el proyecto es un **software de aplicación** ya que su finalidad es facilitar necesidades específicas de los usuarios. Siendo la finalidad del proyecto la creación de una plataforma escolar que permita un desarrollo óptimo de docentes y alumnos, concluimos que un software de aplicación web, también conocido como WebApp, será el ideal para atender las necesidades de este.

Asimismo, hemos descubierto lo esencial que es actuar con ética en cada tarea que desempeñemos como desarrolladores y lo importante que es establecer un Código

que rige la ética de nuestro equipo para que al implementarlo logremos las mejores prácticas en nuestra profesión y, por ende, los mejores resultados en nuestro proyecto.

Por otro lado, entender las etapas que componen el desarrollo de software nos permite tener una visión más amplia de su complejidad y de la importancia de dividir las responsabilidades en roles específicos. Pudimos notar que aunque los roles se definen claramente no debemos perder la visión sistemática, ya que cada decisión tomada, independientemente de la tarea que se desempeñe, tendrá repercusiones en el proyecto. Confirmamos también lo imprescindible que es tener una correcta comunicación entre todos los involucrados para lograr cubrir los objetivos y requerimientos del software solicitado.

Adicionalmente, encontramos en los procesos de gestión una manera de organizar y administrar de forma eficiente el tiempo y los recursos disponibles para nuestro proyecto. Comprendimos que cada una de las etapas están diseñadas para atender tareas específicas que en conjunto buscan cubrir y respetar todos los requerimientos del cliente; además, identificamos la importancia de evaluar los riesgos que se puedan presentar en nuestro proyecto para evitar que éstos retrasen la entrega del mismo o entorpezcan su desarrollo.

Aunado a lo anterior, reiteramos lo beneficioso que es conocer y entender el tipo de personalidad de los integrantes del equipo para poder trabajar de manera cohesiva y mantener una comunicación adecuada, lo que nos permitirá colaborar activamente para ir cubriendo las etapas y objetivos que se forman a lo largo del proyecto.

Descubrimos que la estimación de tiempo y esfuerzo nos ayudarán a establecer metas más claras y distribuir mejor nuestras tareas, sin embargo, sabemos que hay factores que pueden modificar activamente estas estimaciones, por lo que es

importante contar con planes alternativos y revisar constantemente los avances y modificaciones que se presenten en el desarrollo del software.

Con respecto a los modelos de desarrollo de software, verificamos que lo más importante para definir la forma de trabajar en un proyecto es conocerlo a detalle y comprender sus alcances, ya que de esto dependerá el modelo a seguir que nos permita un desenvolvimiento correcto de cada etapa de desarrollo del software. Partiendo de las definiciones, las ventajas y las desventajas de cada modelo, pudimos concretar reglas que definen los momentos o casos en los que mejor se adapta cada uno de ellos aunque entendemos que en la mayoría de los proyectos se utiliza más de un modelo para cubrir de mejor manera los requerimientos del programa.

Dentro de nuestro proyecto, podemos decir que el modelo que más se acomoda a la forma de trabajo del equipo y los posibles requerimientos del sistema sería un modelo de desarrollo ágil, esto debido a que los requerimientos en este punto del proyecto no se encuentran del todo definidos, están en proceso, aunado al hecho de que incluso con requerimientos bien definidos, no se sabe cuándo pueden cambiar o de qué forma van a cambiar, por lo que este modelo se asimila más a los cambios que se presenten en el desarrollo; esto sin mencionar la importancia del cliente y el papel que toma en la realización del proyecto, ya que es más fácil que los objetivos y requerimientos se adecuen a lo que el cliente desea; y de esa forma, entregar un trabajo bien elaborado, que no requiera tantas modificaciones en el proceso.

En cuanto a las metodologías que se encuentran dentro de los modelos de desarrollo ágil, pudimos analizar las diferentes características que le proporcionan a cada modelo puntos a favor y puntos en contra, así como sus principales diferencias para así poder tomar en cuenta la metodología que más se amolda al proyecto. Con lo anterior, definimos que Scrum es el proceso que se adapta de mejor forma a nuestro trabajo, debido a que esta metodología se enfoca mucho en avanzar y tener

resultados rápidos, lo cual se logra a través de sprints y de reuniones diarias para analizar los avances y retrasos del proyecto, enfatizando así la importancia de la comunicación y del contacto que se tiene con el cliente ya que éste es partícipe del avance del proyecto y no tiene que esperar tanto para ver resultados del mismo. Así entonces, después de analizar todas las ventajas que nos proporciona, consideramos que esta metodología es la adecuada para el desarrollo de este proyecto.

También, detectamos que no existen procesos ideales para la ingeniería de requerimientos y, como lo hemos mencionado anteriormente, la clave está en identificar con claridad las necesidades de los usuarios y las organizaciones para adaptar así los procesos de la mejor manera al proyecto a desarrollar. Confirmamos que un buen entendimiento de los requerimientos es básico para comprender tanto el alcance y como los límites del sistema, es por ello que debemos trabajar detalladamente en la especificación de cada requisito y hacer una correcta documentación del mismo tomando como base los modelos y estándares disponibles para cada caso.

De la misma manera, aprendimos que existen diferentes técnicas que en conjunto permiten la recolección de información necesaria para especificar los requerimientos del sistema. Comprendimos que debido a las limitantes actuales derivadas de la pandemia, necesitamos encontrar alternativas para entender las necesidades de las organizaciones de la forma más exacta posible, lo cual nos obliga a fortalecer nuestras habilidades de interacción con el cliente empleando las herramientas remotas que nos ofrece la tecnología actual.

Por otro lado, descubrimos que realizar la especificación de requerimientos es un proceso que demanda mucho tiempo y trabajo constante, ya que se deben de tomar en cuenta todos los detalles que afecten el cumplimiento de las necesidades de los usuarios. Además, es importante utilizar formatos adecuados que permitan la

redacción de los requisitos de forma clara y completa, lo cual a su vez facilita la aplicación de los mismos en el sistema.

Como parte inicial de la etapa de diseño, elaboramos los diagramas de actividades que nos permiten conocer y detallar los procesos que conlleva cada caso de uso. Lo anterior nos ayuda a entender cómo se va a utilizar el sistema y cómo éste se comporta ante eventos específicos.

Adicionalmente, los diagramas de clases y de secuencia complementan el diseño de nuestro proyecto, teniendo como finalidad facilitar la implementación del sistema de tal manera que se integre toda la funcionalidad solicitada.

Asimismo, aprendimos que existen patrones de diseño que nos ayudan a mejorar el desarrollo del sistema y evitan que las modificaciones que pueda sufrir en un futuro el código sean en exceso costosas y laboriosas. Los diferentes tipos de patrones permitirán brindar soluciones ante la creación, estructura y comportamiento del sistema, sin embargo, saber cuáles y cuántos se utilizarán dependerá completamente de la naturaleza del problema.

Por último, conocimos algunas de las múltiples pruebas de validación que existen para asegurar el correcto funcionamiento de un software. Destacamos la importancia de realizar esta etapa contemplando diferentes técnicas de verificación y validación que permitan cubrir los requerimientos funcionales y no funcionales previamente establecidos. De la misma forma, se analizaron los criterios de calidad y mantenimiento que brindarán garantías en el funcionamiento del software a largo plazo, así como los criterios más importantes para realizar la entrega y liberación del mismo como etapa final del desarrollo de software.

Bibliografía

Sommerville, I., & Domínguez Torres, J. (2011). *Ingeniería de software* (9º ed.). Addison-Wesley, Pearson Educación.

Aurora Camero. (2015). *Importancia de la ingeniería de Software*. Visto el 19 de Agosto de 2020. Recuperado de: https://prezi.com/fxdnkvhk_xt0/la-importancia-de-la-ingenieria-del-software/.

Perez, G. (2019). *Clasificación de Software*[Archivo de video]. Visto el 19 de agosto de 2020. Recuperado de https://www.youtube.com/watch?reload=9&v=9etxPJHX260&feature=emb_title.

García Peñalvo, F. (2020). Capítulo 7. *Ingeniería del Software*. Visto el 22 de agosto del 2020. Recuperado de: <https://repositorio.grial.eu/bitstream/grial/1228/1/07-rep.pdf>.

Graterol, L. [StarsConf]. (2018). *Evitemos hacer daño: Ética en la ingeniería de software* [Archivo de video]. Visto el 25 de agosto de 2020. Recuperado de: https://www.youtube.com/watch?time_continue=1&v=6QFY7Gr9k0I&feature=emb_logo.

Boyd, M. (2020). *The Five Principles of Software Ethics – The New Stack*. Visto el 25 de agosto de 2020, Recuperado de: <https://thenewstack.io/five-principles-software-ethics/>.

Código de Ética y Conducta Profesional de ACM (s.f). Visto el 27 de agosto de 2020. Recuperado de <https://www.acm.org/about-acm/code-of-ethics-in-spanish>.

González-Calleros, J. M., & Guerrero-García, J. (2020, 28 julio). Ciclo de Vida y Roles [Diapositivas]. Visto el 27 de agosto de 2020. Recuperado de: <https://www.slideshare.net/jumagoca78/ciclo-de-vida-y-roles>.

Guerrero-García, J., González Calleros, J. M. (2020, 20 julio). Unidad 1 Introducción al Desarrollo de proyectos (1) [Diapositivas]. Visto el 01 de septiembre de 2020. Recuperado de: <https://www.slideshare.net/JosefinaGuerreroGarc1/unidad-1-introduccion-al-desarrollo-de-proyectos-1-237055185>

Ch22 Project Management. Software Engineering Book, 10th Edition. (2005, 2 junio). [Diapositivas]. Visto el 01 de septiembre de 2020. Recuperado de: [https://www.slideshare.net/software-engineering-book/ch22-project-managem](https://www.slideshare.net/software-engineering-book/ch22-project-management)ent

Ch23 Project Planning, Software Engineering Book, 10th edition. (2015, 2 junio). [Diapositivas]. Visto el 02 de septiembre de 2020. Recuperado de: <https://www.slideshare.net/software-engineering-book/ch23-project-planning>

1 presentacion_1_-_ingenieria_de_software[1] (2011, 2 marzo). Visto el 09 de septiembre de 2020. Recuperado de <https://es.slideshare.net/rolmary/1-presentacion1ingenieriaedesoftware1>

Ch2 Software processes. Software Engineering Book. 10th Edition. (2016, 2 junio). [Diapositivas]. Visto el 08 de septiembre de 2020. Recuperado de: <https://www.slideshare.net/software-engineering-book/ch2-sw-processes>

Reutilización de Software - Desarrollo de Software. (s. f.). Google Sites. Visto el 9 de septiembre de 2020. Recuperado de: <https://sites.google.com/site/desarrollodesoftwareuba/reutilizacion>

Engineering Software Products: 2. Agile Software Engineering. Software Engineering Book, 10 th Edition. (2019, 10 febrero). [Diapositivas]. Visto el 15 de septiembre de 2020. Recuperado de: <https://www.slideshare.net/software-engineering-book/engineering-software-products-2-agile-software-engineering>

Meléndez Valladarez, S. M., Gaitán, M. E., & Pérez Reyes, N. N. (2016). Sistema WEB de evaluación al desempeño Docente UNAN-Managua, empleando la metodología Ágil Programación Extrema, en el II Semestre del 2015 (Doctoral dissertation, Universidad Nacional Autónoma de Nicaragua, Managua). Visto el 17 de septiembre de 2020. Recuperado de: <https://repositorio.unan.edu.ni/1365/1/62161.pdf>

González, A. J., & a Objeto, O. (2007). Ingeniería de software: Metodologías. Visto el 18 de septiembre de 2020. Recuperado de: <http://profesores.elo.utfsm.cl/~agv/elo329/1s10/lectures/SoftwareEngineeringParte2.pdf>

Sommerville, I. (2011, 13 octubre). Chap2 RE processes [Diapositivas]. Visto el 23 de septiembre de 2020. Recuperado de: <https://www.slideshare.net/SE9/chap2-re-processes>

Báez, M. G., & Brunner, S. I. B. (2001, November). Metodología DoRCU para la Ingeniería de Requerimientos. In WER (pp. 210-222). Visto el 24 de septiembre de 2020. Recuperado de: http://www.inf.puc-rio.br/wer/WERpapers/artigos/artigos_wer01/baez.pdf

González-Calleros, J. M., & Guerrero-García, J. (2020b, julio 22). Guía de Entrevistas [Diapositivas]. Visto el 30 de septiembre 2020. Recuperado de: <https://www.slideshare.net/jumagoca78/gua-de-entrevistas-para-diseo-interactivo>

Sommerville, I. (2011c, octubre 13). Chap3 RE elicitation [Diapositivas]. Visto el 30

de septiembre de 2020. Recuperado de:

<https://www.slideshare.net/SE9/chap3-re-elicitation>

Fernandez, D. (2014, Agosto 29). Metodologías de Sistemas Blandos. Metodologías

de Sistemas Blandos. Visto el 2 de Octubre de 2020. Recuperado de:

Lucidchart. (2018, agosto 27). Cómo hacer un Diagrama de Secuencias UML

[Vídeo]. Visto el 29 de Octubre de 2020. Recuperado de:

<https://www.youtube.com/watch?v=pCK6prSq8aw>

Universitat Politècnica de València - UPV(2017, Octubre 19). ¿Cómo construir un

diagrama de secuencia? Ejemplo de recuperación de información. Visto el 30

de Octubre de 2020. Recuperado de:

<https://www.youtube.com/watch?v=Q1kH7XKxK5I&t=182s>

Refactoring.Guru. (s. f.-b). Strategy. Visto el 6 de Noviembre de 2020. Recuperado

de: <https://refactoring.guru/design-patterns/strategy>

Refactoring.Guru. (s. f.). Observer. Visto el 6 de Noviembre de 2020. Recuperado

de: <https://refactoring.guru/design-patterns/observer>

Gonzalez Calleros, J.M.. (2020, 08 Noviembre). Patrón Adaptador - Patrones de Diseño [Vídeo]. Visto el 11 de Noviembre de 2020. Recuperado de: <https://www.youtube.com/watch?v=gDN5aThMhFQ>

Refactoring Guru. (s. f.). Decorator. Visto el 11 de Noviembre de 2020. Recuperado de: <https://refactoring.guru/design-patterns/decorator>

Gonzalez Calleros, J.M.. (2020, 08 Noviembre). Patrón Fábrica - Patrones de Diseño Creacionales [Vídeo]. Visto el 12 de Noviembre de 2020. Recuperado de: YouTube. <https://www.youtube.com/watch?v=9wn0rNiK-uQ>

Refactoring Guru. (s. f.-a). Visto el 12 de Noviembre de 2020. Recuperado de: <https://refactoring.guru/design-patterns/abstract-factory>

Refactoring Guru. (s. f.-c). Visto el 12 de Noviembre de 2020. Recuperado de: <https://refactoring.guru/design-patterns/factory-method>

Gonzalez Calleros, J.M.. (2020, 08 Noviembre). Patrón Método Fabrica - Patrón de Diseño Creacional [Vídeo]. Visto el 13 de Noviembre de 2020. Recuperado de: YouTube. <https://www.youtube.com/watch?v=boMaEHEWTsk>

Refactoring Guru. (s. f.-b). Adapter. Visto el 19 de Noviembre de 2020. Recuperado de: <https://refactoring.guru/design-patterns/adapter>

Refactoring Guru. (s. f.-c). Command. Visto el 19 de Noviembre de 2020. Recuperado de: <https://refactoring.guru/design-patterns/command>

Gonzalez Calleros, J.M.. (2020, 23 Octubre). Command Pattern - Patrones de Diseño [Vídeo]. Visto el 19 de Noviembre de 2020. Recuperado de: https://www.youtube.com/watch?v=1_9iy3cWfJk

Guerrero-García, J., & González-Calleros, J. M. (2020, 12 agosto). Unidad 5.

Pruebas de Software [Diapositivas]. Visto el 26 de Noviembre de 2020.

Sicilia, M. A. (2008, 24 noviembre). Tipos de mantenimiento. OpenStax CNX.

<https://cnx.org/contents/uuiMsUHT@3/Tipos-de-Mantenimiento>

ANEXO 1. Requerimientos del proyecto

CASO DE USO: CREAR CUENTA EN LA PLATAFORMA

Revision History

Date	Author	Description of change
07-octubre-2020	Alhelí Moreno Gaytán	Descripción general del caso de uso

Use Case: Crear cuenta en la plataforma

Id: UC- 001

Description

El usuario podrá generar una cuenta en la plataforma escolar por medio de su correo institucional y la creación de una contraseña personal para el acceso al sistema.

Level: High Level

Primary Actors

Alumno , docente

Supporting Actors

No aplica

Stakeholders and Interests

Personal de administración escolar - tiene interés en conocer cuántos usuarios se registraron al sistema.

Pre-Conditions

Los usuarios deberán contar con un correo institucional vigente.

Post Conditions

Success end condition:

- Los usuarios tendrán una cuenta y contraseña con las que podrán acceder a la plataforma.
- El sistema notificará vía email la creación de la cuenta en la plataforma.

Failure end condition:

- El usuario no puede crear su cuenta.

Minimal Guarantee

- El sistema garantiza una única asignación de cuenta por correo institucional.

Trigger

El usuario entra al enlace de la plataforma en su navegador online preferido.

Main Success Scenario

1. El actor da clic en “Crear cuenta”.
2. El sistema muestra una pantalla solicitando los datos personales del actor.
3. El actor introduce los datos solicitados.
4. El sistema verifica los datos y envía un correo de confirmación de correo.
5. El actor confirma el correo.
6. El sistema solicita la creación de una contraseña.
7. El actor crea y confirma su contraseña.
8. El sistema verifica la contraseña.
9. El sistema notifica la creación correcta de la cuenta en pantalla y vía email.

Extensions

4a. En el paso 4, si hay datos incorrectos el sistema notifica al actor.

1. El sistema solicita corregir los datos.
2. El caso de uso continúa en el paso 4.

8a. En el paso 8, si la contraseña creada no coincide con la confirmación el sistema notifica al actor y solicita verificar la contraseña.

1. El caso de uso continúa en el paso 8.

Variations: No aplica

Frequency: Una única vez.

Assumptions

- El usuario conoce el enlace de la plataforma escolar.

Special Requirements

User Interface

1. La interfaz deberá ser amigable e intuitiva para el usuario.

Security

1. El sistema no permitirá que se cree una cuenta si no se confirma a través del enlace enviado al correo que se quiere registrar.
2. El sistema verificará que la cuenta concuerde con el formato institucional.

Issues

1. ¿Cuánto tiempo de vigencia tiene el enlace para confirmar la cuenta?

To do

1. Actualizar base de datos.

CASO DE USO: INGRESAR A LA PLATAFORMA

Revision History

Date	Author	Description of change
07-octubre-2020	Martín A. Paniagua Velázquez.	Descripción general del caso de uso

Use Case: Ingresar a la plataforma.

Id: UC-002

Description

El sistema permitirá el ingreso a cualquier usuario que requiera utilizarla mediante el correo institucional y la contraseña previamente establecida.

Level: High Level.

Primary Actors

Alumno, Docente.

Supporting Actors

Stakeholders and Interests

Desarrolladores - Tienen interés en la cantidad de usuarios que ingresan a la plataforma.

Pre-Conditions

El usuario debe de cubrir con UC-001.

Post Conditions

Success end condition

El usuario podrá ingresar a la plataforma.

Failure end condition:

El usuario no puede acceder al sistema, se queda fuera de la plataforma.

Minimal Guaranteee

El sistema esconde la contraseña ingresada para asegurar que ninguna otra persona pueda robarla.

Trigger

El usuario quiere acceder a la plataforma.

Main Success Scenario

1. El actor coloca la liga en su navegador.
2. El sistema solicita el correo y contraseña registrado.
3. El actor ingresa su correo institucional y su contraseña.
4. El actor da click en Ingresar.
5. El sistema permite el acceso al usuario.

Extensions

4a. En el paso 4, si el usuario colocó una contraseña errónea.

1. El sistema muestra un mensaje que notifica que la contraseña es incorrecta.
2. El actor coloca una contraseña diferente.
3. El caso de uso continúa en el paso 4.

Variations

Frequency: 5 veces al día.

Assumptions

El usuario tiene una cuenta.

Special Requirements

User Interface

1. Interfaz amigable e intuitiva para el usuario.

Security

1. El sistema va a enmascarar la contraseña proporcionada (ejemplo: ****), con el propósito de asegurar que nadie pueda copiarla y hacer mal uso de ella.

Issues

1. ¿Existe un límite de acceso?

To do

1. Registrar el ingreso a la base de datos.

CASO DE USO: VALIDAR USUARIO

Revision History

Date	Author	Description of change
08-octubre-2020	Milcha Sarai Oropeza López	Descripción general del caso de uso

Use Case: Validar y autenticar usuario

Id: UC- 003

Description:

El administrador validará los datos de entrada del usuario para su registro.

Level: High-Level

Primary Actors:

Administrador

Supporting Actors

Usuario

Stakeholders and Interests

Que los datos de entrada emitidos por el usuario, sean verificados y validados

Pre-Conditions

El usuario debe de cumplir con UCC-01

Post Conditions:

Success end condition:

El usuario queda registrado dentro de la plataforma.

Failure end condition:

El usuario no podrá registrarse.

Minimal Guarantee

Trigger

El usuario quiere validar sus datos..

Main Success Scenario

1. El usuario ingresa sus datos en los campos correspondientes
2. El administrador verifica que los datos estén correctos según el campo
3. Al usuario se le da acceso a la plataforma

Extensions

2a. Los datos son incorrectos

1. Se notificará al usuario el tipo de error.
2. El usuario acepta el mensaje.

Variations

No aplica

Frequency.

Una sola vez.

Assumptions

El usuario tiene conocimientos previos de como crear una cuenta y los datos que deben de ir en cada campo.

Special Requirements

User Interface

1. La interfaz debe de ser amigable y fácil de usar.

Issues

¿Cuál es el máximo de cambios que puede hacer un usuario en su perfil?

To do

Registrar en la base de datos.

CASO DE USO: CREAR UN CURSO

Revision History

Date	Author	Description of change
09-octubre-2020	Martín A. Paniagua Velázquez.	Descripción general del caso de uso

Use Case: Inscribirse a un curso.

Id: UC-004

Description

El sistema permitirá al docente crear un curso deseado.

Level: High Level Summary.

Primary Actors

Docente.

Supporting Actors

Alumno.

Stakeholders and Interests

Dirección de la facultad - Tienen interés en conocer los cursos creados dentro del sistema.

Pre-Conditions

El usuario debe de cubrir con UC-001 y con UC-002, además de contar con los datos del curso a crear.

Post Conditions

Success end condition

El usuario podrá crear el curso deseado.

Failure end condition:

El usuario no puede crear el curso deseado.

Minimal Guarantee

El sistema registra el intento de creación del curso y los datos del mismo.

Trigger

El usuario quiere crear un curso.

Main Success Scenario

1. El actor coloca la liga en su navegador.
2. El sistema solicita el correo y contraseña registrado.
3. El actor ingresa su correo institucional y su contraseña.
4. El actor da click en Ingresar.
5. El sistema permite el acceso al usuario.
6. El actor selecciona el botón de Cursos.
7. El actor selecciona Crear nuevo curso.
8. El sistema solicita los datos del curso.
9. El actor selecciona Crear.
10. El sistema muestra los datos del curso y da acceso al mismo.

Extensions

4a. En el paso 4, si el usuario colocó una contraseña errónea.

1. El sistema muestra un mensaje que notifica que la contraseña es incorrecta.
2. El actor coloca una contraseña diferente.
3. El caso de uso continúa en el paso 4.

9a. En el paso 9, si el usuario colocó una clave incorrecta.

1. El sistema muestra un mensaje notificando que la clave es incorrecta.
2. El actor coloca una clave distinta.
3. El caso de uso continúa en el paso 9.

Variations

Frequency: 1 vez por curso.

Assumptions

El usuario tiene una cuenta, el usuario quiere crear un curso y tiene los datos correspondientes.

Special Requirements

User Interface

1. Interfaz amigable e intuitiva para el usuario.

Security

1. El sistema va a asegurar que el usuario que creó el curso sea el único que tenga acceso a las configuraciones del mismo.

Issues

1. ¿Cual es el máximo de cursos que un usuario puede crear?

To do

1. Registrar el curso a la base de datos.
2. Notificar al usuario acerca de los datos del curso.

CASO DE USO: DAR DE BAJA UN CURSO

Revision History

Date	Author	Description of change
09-octubre-2020	Martín A. Paniagua Velázquez.	Descripción general del caso de uso

Use Case: Dar de baja un curso

Id: UC-005

Description

El sistema permitirá a los usuarios dar de baja un curso deseado siempre y cuando cuente con las características para realizar dicha operación.

Level: High Level Summary.

Primary Actors

Alumno.

Supporting Actors

No existe en este caso.

Stakeholders and Interests

Docente - Tiene interés en conocer a los alumnos que se han dado de baja en un curso.

Pre-Conditions

El usuario debe de cubrir con UC-001 y con UC-002, además de estar inscrito a por lo menos un curso.

Post Conditions

Success end condition

El usuario podrá dar de baja un curso.

Failure end condition:

El usuario no podrá dar de baja el curso deseado.

Minimal Guarantee

El sistema manda una notificación cuando el usuario intenta dar de baja un curso.

Trigger

El usuario quiere iniciar o necesita dar de baja un curso.

Main Success Scenario

1. El actor coloca la liga en su navegador.
2. El sistema solicita el correo y contraseña registrado.
3. El actor ingresa su correo institucional y su contraseña.
4. El actor da click en Ingresar.
5. El sistema permite el acceso al usuario.
6. El actor selecciona el botón de Cursos.
7. El actor selecciona el curso deseado.
8. El sistema muestra la pantalla de inicio del curso.

9. El actor selecciona Dar de baja.
10. El sistema solicita una confirmación de solicitud.
11. El actor selecciona Acepto.
12. El sistema da de baja la materia.

Extensions

- 4a. En el paso 4, si el usuario colocó una contraseña errónea.
 1. El sistema muestra un mensaje que notifica que la contraseña es incorrecta.
 2. El actor coloca una contraseña diferente.
 3. El caso de uso continúa en el paso 4.
- 11a. En el paso 11, si existe un error técnico.
 1. El sistema muestra un mensaje notificando que hay un error.
 2. El sistema da la opción de intentar de nuevo.
 3. El caso de uso continúa en el paso 11.

Variations

No existen variaciones en este caso.

Frequency: 1 vez por materia.

Assumptions

El usuario tiene una cuenta, el usuario está inscrito al menos a un curso, el usuario desea darse de baja de un curso.

Special Requirements

User Interface

1. Interfaz amigable e intuitiva para el usuario.

Security

1. El sistema asegura que una persona que no esté autorizada será incapaz de dar de baja un curso que no le corresponde.

Issues

1. ¿Cuál es el máximo de materias a dar de baja?

To do

1. Eliminar al usuario de la base de datos.
2. Notificar al docente de la baja.

CASO DE USO: CREAR GRUPOS ENTRE LOS USUARIOS

Revision History

Date	Author	Description of change
09-octubre-2020	Ramiro Vidal Lumbreras	Descripción general del caso de uso
16-octubre-2020	Ramiro Vidal Lumbreras	Actualización de ID, Main Success, Extensions

Use Case: Crear grupos entre los usuarios

Id: UC-006

Description:

El software permite a los usuarios poder crear grupos entre ellos, diferente a los de un curso, para chats, llamadas grupales o compartir contenido.

Level: High Level

Primary Actors:

Alumnos.

Supporting Actors:

Stakeholders and Interests:

Principalmente a los alumnos que por cualquier situación tengan que comentar cuestiones escolares, compartir material o realizar llamadas entre ellos.

Pre-Conditions:

Debe de cumplir con: UC-001, UC-002, UC-003 y UC-004.

Post Conditions:

Success end condition:

Failure end condition:

- El usuario no termina el proceso de crear un grupo.
- Los usuarios no están registrados en la plataforma o falta validar.

Minimal Guarantee:

- Cuando el proceso de crear un grupo no se termina, se queda guardado el progreso para poder finalizarlo.

Trigger:

Un usuario quiere crear un grupo por las cuestiones que sean.

Main Success Scenario

1. En el menú de opciones.
2. Seleccionar el apartado de grupos.
3. Click en crear grupo
4. Registrar miembros
5. Personalizar grupo.
6. Guardar grupo

Extensions

4a. Exceder el límite de miembros

1. Mensaje donde notifica que se excedió el límite de miembros.
2. Quitar miembros.

4b. El usuario no está registrado

1. Mensaje avisando que hay datos incorrectos o el usuario no está registrado.
2. Se continúa agregando miembros.

6a. No se guarda el grupo.

1. Se guarda el progreso para poder terminarlo correctamente.

Variations

Agregar miembros ya sea por:

- Nombre completo
- Matricula
- Correo institucional

Frequency: 0 o 1 vez al día.

Assumptions:

- El usuario sabe crear un grupo.
- El usuario conoce los datos para agregar a los miembros.

Special Requirements:

User interface

- Interfaz amigable con el usuario.
- El sistema te notifica cuando creas o seas añadido a un grupo.

Security

- Solo las personas con permisos podrán añadir miembros.

Issues:

- Quien crea el grupo tenga datos incorrectos de los miembros.
- Errores de conexión.

To do:

Hacer que los usuarios puedan crear grupos con los demás usuarios.

CASO DE USO: CREACIÓN DE NOTAS EN SESIÓN

Revision History

Date	Author	Description of change
07-octubre-2020	Erick Nuñez Grajales	Descripción general del caso de uso
16-Octubre-2020	Erick Nuñez Grajales	Modificación ID

Use Case: Creación de notas de la sesión

Id: UC- 007

Description:

El alumno podrá crear notas de la sesión de lo que crean necesario.

Level:

Low-Level Summary

Primary Actors:

Alumnos

Supporting Actors

No se presentan en este caso.

Stakeholders and Interests

Estudiantes - Interés en poder anotar cosas importantes, y quedarán registradas

Pre-Conditions

El usuario debe de cumplir con UCC-01

Post Conditions:Success end condition:

El alumno podrá tomar notas de la sesión y acceder después a ellas.

Failure end condition:

El alumno no habrá podido tomar notas o no se habrán quedado guardadas.

Minimal Guarantee

El alumno tendrá un registro (log) de todo lo que haya anotado en las notas.

Trigger

El alumno quiere tomar notas de la sesión.

Main Success Scenario

1. El alumno accede al portal académico.
2. El alumno accede a la sesión
3. El alumno hace clic en el símbolo de notas y empieza a escribir.

Extensions

4a. En caso de que se no se guardarán las notas.

1. El alumno tiene en la sección rápida, un acceso a las notas guardadas.
2. El software guardará las notas aunque no se hayan guardado correctamente

3. El alumno descarga las notas o las copiara como prefiera más.

Frequency:

Dependiendo de las veces que el alumno requiera tomar notas virtuales.

Assumptions

El alumno tiene conocimiento previo de la plataforma y tiene las notas virtuales.

Special Requirements

User Interface

1. La interfaz debe de ser amigable y fácil de usar.

Security

1. El sistema debe de haber hecho un autoguardado cada vez que el alumno haya dado un enter

To do

1. Registro en Base de datos.

CASO DE USO: PROGRAMAR VIDEOCONFERENCIA / SESIÓN

Revision History

Date	Author	Description of change
07-octubre-2020	Erick Nuñez Grajales	Descripción general del caso de uso
16-Octubre-2020	Erick Nuñez Grajales	Modificación ID

Use Case: Programar una videoconferencia / sesión

Id: UC- 008

Description:

El docente podrá programar una o varias videoconferencias.

Level: High-Level

Primary Actors:

Docentes

Supporting Actors

No se presentan en este caso.

Stakeholders and Interests

Docentes - Interés en poder dejar registradas con anticipación futuras clases.

Pre-Conditions

El usuario debe de cumplir con UCC-01

Post Conditions:

Success end condition:

El docente programará una videoconferencia / clase.

Failure end condition:

El profesor no podrá programar la clase / videoconferencia.

Minimal Guarantee

El profesor podrá crear la clase el día para el cual la haya programado.

Trigger

El profesor quiere con anticipación dejar programada una clase.

Main Success Scenario

1. El profesor accede al portal académico.
2. El profesor hace clic en agregar videoconferencia.
3. El profesor logra programar con éxito la videoconferencia.

Extensions

2a. El profesor no logra programar la videoconferencia.

1. El profesor les informa a sus alumnos de la conferencia en el tablero del grupo.
2. El profesor crea una videoconferencia el día para el cual estaba programado.

Frequency:

Las veces que el profesor desee

Assumptions

El profesor tiene conocimiento previo del sistema.

Special Requirements

User Interface

2. La interfaz debe de ser amigable y fácil de usar.

To do

1. Registro en línea de la videoconferencia
2. Notificación a los estudiantes de la clase

CASO DE USO: ALUMNOS GRABAR CLASE

Revision History

Date	Author	Description of change
07-octubre-2020	Erick Nuñez Grajales	Descripción general del caso de uso
16-Octubre-2020	Erick Nuñez Grajales	Modificación ID

Use Case: Grabar clase

Id: UC- 009

Description:

El alumno podrá grabar la clase en caso de que quiera repetirla después con la autorización del docente.

Level:

Low-Level Summary

Primary Actors:

Alumnos

Supporting Actors

Docente

Stakeholders and Interests

Estudiantes - Interés en poder repasar más tarde en caso de que hayan omitido algo o no hayan entendido la clase para mejor entendimiento.

Pre-Conditions

El usuario debe de cumplir con UCC-01

Post Conditions:

Success end condition:

El alumno habrá podido grabar la clase

Failure end condition:

El alumno no habrá podido grabar la clase

Minimal Guarantee

El alumno podrá tomar fotos de pantalla a datos que crea relevantes en la clase

Trigger

El alumno requiere grabar la clase porque quiere repasar el tema.

Main Success Scenario

1. El alumno accede a la sesión de la materia.
2. El alumno oprime el botón de grabar.
3. El profesor permite la grabación de la clase.

Extensions

3a. En caso de que el profesor no permita que el alumno pueda grabar la materia.

1. El alumno abrirá la aplicación de editor de texto preferida.
2. El alumno tomará una foto de pantalla de la sesión.
3. El alumno pegara la foto en el editor de texto
4. Repetirá los pasos que crea necesario.

Frequency:

Todas las veces que el alumno requiera necesario.

Assumptions

El alumno tiene conocimiento previo del uso del sistema

Special Requirements

User Interface

1. La interfaz debe de ser amigable y fácil de usar.

To do

1. El alumno guardará el video de la clase

CASO DE USO: TOMAR ASISTENCIA

Revision History

Date	Author	Description of change
16-octubre-2020	Erick Nuñez Grajales	Descripción general del caso de uso

Use Case: Revisar asistencia

Id: UC- 010

Description:

El docente podrá revisar que alumnos ingresaron a la videoconferencia/sesión y de quienes llegaron tarde y cuanto tiempo estuvieron.

Level:

High-Level Summary

Primary Actors:

Docentes

Supporting Actors

Alumnos

Stakeholders and Interests

Docentes - Interés en no perder tiempo en tomar lista y retardos.

Pre-Conditions

El usuario debe de cumplir con UCC-01

Post Conditions:

Success end condition:

El docente podrá llevar una asistencia más certera

Failure end condition:

No se llevó correctamente la asistencia de los de la clase.

Minimal Guarantee

El sistema de todos modos tomó nota de que alumnos entraron a la videoconferencia y a que hora.

Trigger

El profesor desea llevar registro de quien entró en la sesión

Main Success Scenario

1. El profesor inicia la videoconferencia.
2. El profesor, una vez finalizada la videoconferencia, checa en la lista de quienes entraron en la sesión.
3. La asistencia se tomó automática

Extensions

3a. En caso de que no se haya tomado correctamente la asistencia, el sistema habrá hecho un registro de quienes entraron y a que hora.

1. El profesor accede en la parte de la videoconferencia, a las propiedades.
2. El profesor dará clic en detalles y encontrará información de los alumnos que ingresaron a la sesión

Frequency:

Todas las veces que el profesor haya hecho una sesión.

Assumptions

El docente tiene conocimiento del funcionamiento del software.

Special Requirements

User Interface

1. La interfaz debe de ser amigable y fácil de usar.

To do

1. El alumno guardará el video de la clase

CASO DE USO: INICIAR VIDEOCONFERENCIA.

Revision History

Date	Author	Description of change
09-octubre-2020	Martín A. Paniagua Velázquez.	Descripción general del caso de uso

Use Case: Iniciar una videoconferencia.

Id: UC-011

Description

El sistema permitirá a los usuarios comenzar con una videoconferencia en cualquier momento.

Level: High Level Summary.

Primary Actors

Alumno, Docente.

Supporting Actors

No existe en este caso.

Stakeholders and Interests

Invitados- Tienen interés en conocer el curso por lo que entran a la videoconferencia.

Pre-Conditions

El usuario debe de cubrir con UC-001 y con UC-002, además de estar inscrito a por lo menos un curso.

Post Conditions

Success end condition

El usuario podrá iniciar una videoconferencia.

Failure end condition:

El usuario no podrá iniciar la videoconferencia.

Minimal Guarantee

El sistema manda una notificación al momento de tratar de iniciar la videoconferencia.

Trigger

El usuario quiere iniciar una videollamada.

Main Success Scenario

1. El actor coloca la liga en su navegador.
2. El sistema solicita el correo y contraseña registrado.
3. El actor ingresa su correo institucional y su contraseña.
4. El actor da click en Ingresar.
5. El sistema permite el acceso al usuario.
6. El actor selecciona el botón de Cursos.
7. El actor selecciona el curso deseado.
8. El sistema muestra la pantalla de inicio del curso.
9. El actor selecciona Iniciar Videoconferencia.
10. El sistema solicita la configuración de entrada.
11. El actor selecciona sus preferencias y da click en Iniciar.
12. El sistema inicia la videoconferencia.

Extensions

- 4a. En el paso 4, si el usuario colocó una contraseña errónea.
 4. El sistema muestra un mensaje que notifica que la contraseña es incorrecta.
 5. El actor coloca una contraseña diferente.
 6. El caso de uso continúa en el paso 4.
- 11a. En el paso 11, si existe un error técnico.
 4. El sistema muestra un mensaje notificando que hay un error.
 5. El sistema da la opción de intentar de nuevo.
 6. El caso de uso continúa en el paso 11.

Variations

Las preferencias de la videoconferencia pueden seleccionarse de manera táctil.

Frequency: 6 veces al día.

Assumptions

El usuario tiene una cuenta, el usuario está inscrito al menos a un curso.

Special Requirements

User Interface

1. Interfaz amigable e intuitiva para el usuario.

Security

1. El sistema va a asegurar que las preferencias seleccionadas se cumplan en todos los casos.

Issues

2. ¿Existen usuarios que no pueden iniciar una videoconferencia?

To do

Registrar la videoconferencia en la base de datos.

Notificar a los usuarios inscritos de la videollamada entrante.

CASO DE USO: DESCARGAR CONTENIDO

Revision History

Date	Author	Description of change
16-octubre -2020	Alhelí Moreno Gaytán	Descripción general del caso de uso

Use Case: Descargar contenido

Id: UC- 012

Description

Los usuarios podrán descargar el contenido que esté disponible y cuente con esta opción.

Level: Mid Level

Primary Actors

Alumno , docente

Supporting Actors

No aplica

Stakeholders and Interests

Docentes y alumnos: tienen interés en descargar contenido de los cursos.

Pre-Conditions

Los usuarios deberán cubrir el UC-001, UC-002, además de que deberá existir contenido disponible y habilitado para su descarga.

Post Conditions

Success end condition:

- Los usuarios habrán descargado con éxito el contenido deseado.

Failure end condition:

- La descarga del contenido no se realiza.

Minimal Guarantee

- El sistema permitirá la descarga de archivos garantizando que éstos no ocasionarán inconvenientes en los dispositivos en los que se guarden.

Trigger

El usuario desea descargar contenido de un curso.

Main Success Scenario

1. El usuario entra a la opción de “Contenido del curso”
2. El sistema muestra el contenido disponible.
3. El usuario accede a las opciones del contenido y da click en descargar.
4. El sistema muestra el avance de la descarga.
5. El sistema notifica la descarga finalizada.

Extensions

4a. En el paso 4, si existe una falla en la descarga.

1. El sistema notifica al usuario un fallo en la descarga y pide que lo intente de nuevo.

Variations: No aplica

Frequency: 2 veces al día.

Assumptions

- El usuario identifica la sección de contenido del curso.

Special Requirements

User Interface

1. La interfaz deberá ser amigable e intuitiva para el usuario.

Security

1. El sistema garantiza la descarga segura del contenido en los dispositivos desde los que se acceda a la plataforma.

Issues

1. ¿Existe un límite de descargas del contenido?

To do

1. Actualizar el historial de descargas

CASO DE USO: REVISAR CONTENIDO DE UN CURSO

Revision History

Date	Author	Description of change
07-octubre-2020	Alhelí Moreno Gaytán	Descripción general del caso de uso
16-octubre-2020	Alhelí Moreno Gaytán	Actualización de ID.

Use Case: Revisar contenido de un curso

Id: UC- 013

Description

El usuario podrá acceder al contenido disponible de cualquier curso al que esté inscrito.

Level: High Level

Primary Actors

Alumno

Supporting Actors

No aplica

Stakeholders and Interests

Docente: tiene interés por conocer la frecuencia con la que se consulta el contenido del curso.

Pre-Conditions

Los actores deberán cubrir los casos de uso UC-001, UC-002, además de estar inscritos en al menos un curso.

Post Conditions

Success end condition:

- Los usuarios habrán revisado el contenido del curso deseado.

Failure end condition:

- El usuario no puede acceder al contenido del curso

Minimal Guarantee

- El sistema garantiza la visualización adecuada de todos los contenidos asignados al curso.

Trigger

El usuario tiene interés en revisar el contenido de un curso en específico.

Main Success Scenario

1. El alumno accede al curso de interés.
2. El alumno dará clic en “Contenido del curso”.
3. El sistema mostrará en pantalla el contenido disponible.
4. El alumno seleccionará el enlace con el contenido de interés.

5. El alumno podrá previsualizar el contenido en la plataforma.

Extensions

3a. En el paso 3, si no hay contenido disponible:

1. El sistema notifica que no hay contenido disponible por el momento.

4a. En el paso 4, si el enlace seleccionado no tiene contenido disponible.

1. El sistema notifica que no hay contenido disponible por el momento.

Variations No aplica

Frequency: 5 veces al día.

Assumptions

- El alumno entiende cómo acceder a un enlace.

Special Requirements

User Interface

1. La interfaz deberá ser amigable e intuitiva para el usuario.
2. El contenido mostrado deberá tener opciones para una mejor visualización (zoom in, zoom out).

Security

1. El sistema no mostrará contenido que pueda afectar el software de los usuarios.

Issues

1. ¿El usuario podrá descargar el contenido disponible?

To do

1. Asegurar que el contenido precargado en el curso no tenga problemas de formato o lectura.

CASO DE USO: REALIZAR ENTREGA DE UNA TAREA

Revision History

Date	Author	Description of change
08-octubre-2020	Alhelí Moreno Gaytán	Descripción general del caso de uso
14-octubre-2020	Alhelí Moreno Gaytán	Modificación en preconditions, issues, to do. Actualización de ID.

Use Case: Realizar entrega de una tarea

Id: UC- 014

Description

El usuario podrá realizar la entrega de tareas asignadas mientras éstas se encuentren activas.

Level: High Level

Primary Actors

Alumno

Supporting Actors

No aplica

Stakeholders and Interests

Docente: tiene interés por conocer quiénes entregaron la tarea asignada.

Pre-Conditions

Los actores deberán cubrir los casos de uso UC-001, UC-002, además de estar inscritos en al menos un curso y tener una tarea asignada. Los documentos por adjuntar deberán cumplir con los formatos permitidos (pdf, xlsx, doc, pptx, docx, jpg, jpeg, png) con un tamaño no mayor a 10MB si se adjuntan directamente.

La tarea deberá estar activa.

Post Conditions

Success end condition:

- El alumno realiza la entrega de la tarea con éxito.

- El sistema arroja un mensaje de entrega exitosa incluyendo la fecha y la hora.
- El sistema notifica la entrega al docente del curso correspondiente.

Failure end condition:

- El alumno no concreta la entrega de la tarea.

Minimal Guarantee

- El sistema permite la carga de archivos con extensión pdf, xlsx, doc, pptx, docx, jpg, jpeg, png y la visualización correcta de las tareas asignadas.
- El sistema permite cargar de 0 a 10 documentos por tarea mientras estos no rebasen el límite de tamaño.

Trigger

El alumno quiere hacer la entrega de una tarea asignada

Main Success Scenario

1. El alumno da clic en “ver tarea”.
2. El alumno da clic en “agregar tarea”.
3. El sistema permite al alumno cargar un archivo desde su dispositivo.
4. El alumno selecciona la opción deseada.
5. El alumno da clic en “adjuntar archivo”.
6. El sistema carga el archivo.
7. El sistema muestra el archivo adjunto en la pantalla.
8. El alumno da clic en “entregar tarea”.
9. El sistema pide confirmación de entrega.
10. El alumno confirma la entrega.
11. El sistema muestra un mensaje de entrega exitosa.

Extensions

6a. En el paso 6, el sistema no puede cargar el archivo.

1. El sistema notifica el fallo y solicita al alumno verificar el archivo.
2. El caso de uso continúa en el paso 3.

10a. En el paso 10, el alumno no confirma la entrega.

1. El sistema vuelve a la pantalla inicial de la tarea.

Variations No aplica

Frequency: 4 veces al día.

Assumptions

- El alumno sabe cómo buscar un archivo en su dispositivo.

Special Requirements

User Interface

1. La interfaz deberá ser amigable e intuitiva para el usuario.

Security

1. Los archivos cargados sólo podrán ser accedidos por el alumno que los adjuntó y el docente asociado al curso.

Issues

1. ¿Será posible adjuntar videos?
2. ¿Qué sucede si los archivos rebasan el límite de tamaño?

To do

1. Tener disponible en la plataforma todo el material adjunto de las tareas.

CASO DE USO: CREAR RECORDATORIOS

Revision History

Date	Author	Description of change
09-octubre-2020	Ramiro Vidal Lumbreras	Descripción general del caso de uso
16-octubre-2020	Ramiro Vidal Lumbreras	Actualización de ID

Use Case: Crear recordatorios.

Id: UC-015

Description:

El software permite a los usuarios poder crear sus propios recordatorios para que tengan algo parecido a una agenda.

Level: High Level

Primary Actors:

Alumnos - Docentes

Supporting Actors:

Stakeholders and Interests:

Tanto como alumnos y docentes tengan recordatorios para poder tener un mejor control de su tiempo y actividades.

Pre-Conditions:

Debe de cumplir con: UC-001, UC-002, UC-003 y UC-004.

Post Conditions:

Success end condition:

Failure end condition:

- El usuario no termina el proceso de crear el recordatorio.
- Los usuarios no están registrados en la plataforma o falta validar.
- El usuario no sabe cómo hacer un recordatorio.

Minimal Guarantee:

- Un proceso intuitivo para crear recordatorios.
- En caso de no terminar el recordatorio, guarda la información para terminar el proceso después si así lo desea el usuario.
- Notificación de recordatorios.

Trigger:

Un usuario quiere crear un recordatorio o aviso para el mismo.

Main Success Scenario

1. En el apartado de recordatorios.

2. Seleccionar “crear recordatorio”.
3. Llenar información del recordatorio.
4. Seleccionar fecha y hora.
5. Guardar recordatorio.

Extensions

- 4a. La fecha es incorrecta.
 1. Cambiar fecha y hora
- 5a. No se guarde el recordatorio.
 1. Notificar al usuario.
 2. Revisar conexión.
 3. Volver a intentarlo, guardando los datos rellenados por el usuario.

Variations

Frequency: 2 o 6 veces al día.

Assumptions:

- El usuario sabe crear un recordatorio.

Special Requirements:

User interface

- Interfaz amigable con el usuario.
- Notificación diferente cuando sea un recordatorio.

Security

- Solo el dueño de la cuenta puede interactuar con sus recordatorios.

Issues:

- Errores de conexión.

To do:

- Los usuarios puedan crear sus recordatorios.

CASO DE USO: ENVIAR MENSAJES

Revision History

Date	Author	Description of change
16-octubre -2020	Alhelí Moreno Gaytán	Descripción general del caso de uso

Use Case: Enviar mensajes

Id: UC- 016

Description

Los usuarios podrán enviar mensajes a través del chat de la plataforma para comunicarse con otros usuarios de la misma.

Level: High Level

Primary Actors

Alumno , docente

Supporting Actors

No aplica

Stakeholders and Interests

Docentes y alumnos: tienen interés en comunicarse con los usuarios de la plataforma.

Pre-Conditions

Los usuarios deberán cubrir el UC-001, UC-002.

Post Conditions

Success end condition:

- Los usuarios habrán enviado de forma exitosa el(los) mensaje(s) deseado(s).
- El sistema indica por medio de un ícono el éxito del envío de mensaje.

Failure end condition:

- El mensaje no es enviado.

Minimal Guarantee

- El sistema permitirá el envío de mensajes de forma individual y también en los chats de cada curso.

Trigger

El usuario desea enviar un mensaje.

Main Success Scenario

1. El usuario da click en la sección de chat.
2. El sistema muestra una pantalla con las conversaciones existentes.
3. El usuario escribe el nombre o correo de la persona a la que desea enviar el mensaje.
4. El sistema da una vista previa del contacto elegido.
5. El usuario selecciona al contacto o contactos a los cuales desea enviar un mensaje.
6. El sistema muestra en pantalla un espacio para enviar mensajes.
7. El usuario escribe el mensaje que desea enviar.
8. El usuario da clic en “enviar mensaje”.
9. El sistema muestra en pantalla el mensaje enviado.

Extensions

3a. En el paso 3, si no hay coincidencia de datos del nombre o correo escritos:

1. El sistema notifica que no hay resultados disponibles.

9a. En el paso 9, si el sistema no pudo enviar el mensaje.

1. El sistema notifica que no se pudo enviar el mensaje y pide al usuario intentarlo de nuevo.

Variations: No aplica

Frequency: 3 veces al día.

Assumptions

- El usuario reconoce la sección de chat en la plataforma.

Special Requirements

User Interface

1. La interfaz deberá ser amigable e intuitiva para el usuario.

Security

1. El sistema garantiza la privacidad de los mensajes y notifica cuando la información es compartida con más de un usuario.

Issues

1. ¿Cuál es el tamaño máximo de cada mensaje?

To do

1. Actualizar el historial de mensajes de cada usuario.

CASO DE USO: PREVISUALIZACIÓN DE ARCHIVOS

Revision History

Date	Author	Description of change
08-octubre-2020	Jose Carlos Mejia Ramirez	Descripción general del caso de uso

Use Case: Previsualización de archivos

Id: UC- 017

Description

El sistema contará con una previsualización para que el alumno pueda ver que está a punto de enviar antes de enviarlo.

Level: High-Level

Primary Actors

Alumno

Supporting Actors

No cuenta con otros actores.

Stakeholders and Interests

Alumnos que quieran estar seguros de que estén mandando el documento correcto y que cuente con buena presentación.

Pre-Conditions

Debe cumplir primero con UC-001,UC-002,UC-003,UC-010

Post Conditions

Failure end condition:

- El alumno no puede visualizar lo que desea entregar.

Minimal Guarantee:

- Aún podrá mandar el archivo pero sin ver lo que contiene antes.

Trigger

El docente crea una actividad que necesita ser entregada (UC-10).

Main Success Scenario

1. El docente crea una actividad para ser entregada.

2. El alumno sube el archivo a ser entregado.
3. El sistema muestra solo una previsualización del archivo.
4. El alumno entrega el archivo.

Extensions

- 3a. El alumno no puede ver la previsualización.
 1. El sistema da a elegir si entregar o cancelar la entrega.
 2. El alumno entrega el trabajo.

Variations

1. El alumno sube un archivo invalido.
 1. El sistema le dirá que no soporta ese archivo para la entrega y sugerirá otro tipo de formato.

Frequency: 1 vez al día.

Assumptions

El alumno conoce el tipo de formato de sus archivos.

El alumno sabe como subir y entregar los archivos.

Special Requirements

User Interface

1. Mostrar de manera clara la actividad que se requiere para ser entregada.

Security

1. No se pueden entregar archivos con el mismo nombre y formato.

Performance

1. El tiempo de subida del archivo debe durar máximo 10 minutos.

Issues

1. No se pueden visualizar los archivos antes de enviarlos.

To do

1. Mostrar las actividades pendientes.

CASO DE USO: ACCEDER A VIDEOLLAMADA ACTIVA

Revision History

Date	Author	Description of change
09-octubre-2020	Alhelí Moreno Gaytán	Descripción general del caso de uso
16-octubre-2020	Alhelí Moreno Gaytán	Actualización de ID.

Use Case: Acceder a videollamada activa

Id: UC- 018

Description

El usuario podrá acceder a una videollamada/sesión activa de los cursos en los que se encuentre inscrito.

Level: High Level

Primary Actors

Alumno

Supporting Actors

No aplica

Stakeholders and Interests

Docente - tiene interés en conocer a los alumnos que acceden a las videoconferencias.

Pre-Conditions

Los actores deberán cubrir con los casos de uso UC-001, UC-002, estar inscritos en al menos un curso.

Post Conditions

Success end condition:

- Los alumnos podrán acceder y participar en la videoconferencia activa.

Failure end condition:

- El usuario no puede acceder a videoconferencia.

Minimal Guarantee

- El sistema únicamente mostrará las videoconferencias activas a los alumnos dados de alta en el curso correspondiente.

Trigger

Existe una videoconferencia activa a la cual desea ingresar el alumno.

El alumno está en la sección del curso con la videoconferencia activa.

Main Success Scenario

1. El alumno da clic en “Unirse a videoconferencia”.
2. El sistema solicita que se confirme el uso de video y audio.
3. El alumno elige las opciones deseadas y da clic en “Acceder”.
4. El sistema configura las opciones y permite el acceso a la videoconferencia.
5. El alumno accede a la videoconferencia.

Extensions

4a. En el paso 4, en caso de que la configuración de la videoconferencia requiera de la confirmación del creador de la misma:

1. El sistema notifica al alumno que se espera la confirmación de acceso.
2. El caso de uso continúa en el paso 5.

Variations: No aplica

Frequency: 5 veces al día.

Assumptions

- El alumno entiende las indicaciones que brinda el sistema para el acceso a la videoconferencia.

Special Requirements

User Interface

1. La interfaz deberá ser amigable e intuitiva para el usuario.

Security

2. El sistema deberá asegurar los datos de los usuarios y les solicitará permisos para el acceso de audio y video.

Issues

1.

To do

Actualizar la base de datos con la nueva contraseña

CASO DE USO: BLOQUEAR MENSAJES/LLAMADAS DE OTROS USUARIOS.

Revision History

Date	Author	Description of change
08-octubre-2020	Jose Carlos Mejia Ramirez	Descripción general del caso de uso

Use Case: Bloquear mensajes/llamadas de otros usuarios.

Id: UC- 019

Description

El sistema permitirá a tanto docentes como alumnos la opción de bloquear mensajes/llamadas de otros usuarios.

Level: Mid-Level

Primary Actors

Alumno , docente

Supporting Actors

El usuario al cual se desea bloquear.

Stakeholders and Interests

Alumnos o Docentes que deseen dejar de recibir mensajes de miembros específicos por razones personales.

Pre-Conditions

Debe cumplir primero con UCC-01

Post Conditions

Failure end condition:

- El usuario no puede bloquear a los miembros y sigue recibiendo mensajes/llamadas.

Minimal Guarantee:

- No se notifican los mensajes recibidos.

Trigger

Para el usuario que bloquea: Indicará que usuario desea bloquear y confirma la acción.

Para el usuario bloqueado: Se desactiva la opción de poder enviar mensajes/llamadas a la persona que decidió bloquearlo.

Main Success Scenario

1. El usuario que se quiere bloquear es bloqueado con éxito sin posibilidad de seguir recibiendo mensajes/llamadas del usuario bloqueado.
2. El usuario bloqueado no puede mandar mensajes/llamadas al usuario el cual ha decidido bloquearlo.

Extensions

Variations

En lugar de bloquearlo puede solo deshabilitar las notificaciones de los mensajes o llamadas.

Frequency: 50 veces al dia.

Assumptions

- El usuario conoce el perfil de la persona que quiere bloquear.

Special Requirements

User Interface

1. La interfaz debe dar claro que puedes bloquear usuarios.
2. Debe ser claro a quien se quiere bloquear.

Security

3. El usuario no puede saber que ha sido bloqueado, solo el contacto desaparecerá de la lista de contactos.

Issues

1. El usuario bloqueado pueda seguir teniendo comunicación por otros medios de forma privada.

To do

1. Seguir recibiendo mensajes/llamadas de otros usuarios.

CASO DE USO: VISUALIZAR TAREAS O ACTIVIDADES EN MODO OFFLINE

Revision History

Date	Author	Description of change
09-octubre-2020	Ramiro Vidal Lumbreras	Descripción general del caso de uso
16-octubre-2020	Ramiro Vidal Lumbreras	Actualización de ID, Main Success, Extensions

Use Case: Visualizar tareas o actividades en modo OFFLINE

Id: UC-020

Description:

El software permite a los usuarios poder visualizar tareas o actividades cuando no tenga conexión a internet, siempre y cuando la tarea o actividad haya sido registrada por el sistema cuando tenía una conexión a internet.

Level: High Level

Primary Actors:

Alumnos.

Supporting Actors:

Stakeholders and Interests:

Principalmente a los alumnos que por cualquier situación no tengan internet en ese momento y quieran seguir haciendo su tarea sin problemas pude visualizarla.

Pre-Conditions:

Debe de cumplir con: UC-001, UC-003 y UC-004.

Y que la tarea la haya descargado el sistema cuando tenía conexión a internet.

Post Conditions:

Success end condition:

- El sistema descargo la tarea para su visualización en modo offline.

Failure end condition:

- El sistema no logró descargar la tarea porque no tenía conexión a internet.
- Poco almacenamiento donde descargarla.

Minimal Guarantee:

- Siempre que llegue una tarea o actividad, y se cuente con internet, se descargara automáticamente para una visualización en modo offline.

Trigger:

El docente deja una tarea o actividad al alumno.

Main Success Scenario

1. En modo offline, tendrá el apartado de visualizar tareas o actividades.
2. Elegir cual tarea o actividad visualizar.

Extensions

2a. En caso de no tener tareas o actividades.

1. Notificará al usuario la hora de la última sincronización cuando tenía internet.

Variations

Frequency: 0 a 6 veces al día.

Assumptions:

- El usuario conoce el modo offline.
- El sistema haya hecho una descarga de tareas o actividades.

Special Requirements:

User interface

- Interfaz amigable con el usuario.
- Esperar unos segundos para que aparezcan las tareas para visualizar.

Security

Issues:

- El sistema no realizó la descarga de las tareas o actividades por cuestiones de conexión

To do:

- Hacer que los usuarios solo puedan visualizar sus tareas o actividades sin tener una conexión.

CASO DE USO: ACCEDER A EDITORES

Revision History

Date	Author	Description of change
------	--------	-----------------------

09-octubre-2020	Milcha Sarai Oropeza López	Descripción general del caso de uso
16-octubre-2020	Milcha Sara Oropeza López	Modificación de ID

Use Case: Acceder a editores

Id: UC- 021

Description

Los usuarios podrán abrir editores.

Level: Low Level

Primary Actors

Alumno, Docente

Supporting Actors

No aplica

Stakeholders and Interests

Docente - tiene interés en abrir un editor online.

Alumno - tiene interés en abrir un editor online.

Pre-Conditions

Los actores deberán cubrir con los casos de uso UC-001, UC-002..

Post Conditions

Success end condition:

- Los usuarios abrirán un editor.

Failure end condition:

- El alumno o el docente no pueden abrir el editor

Minimal Guarantee

- El sistema únicamente abrirá el editor.

Trigger

Existen editores por abrir.

Main Success Scenario

1. El usuario da click en más opciones.
2. Selecciona la opción del editor de su preferencia.
3. Se abre el editor seleccionado

Extensions

No aplica.

Variations: No se puede abrir el editor

Frequency: Indefinida.

Assumptions

- El usuario sabe en qué parte de la plataforma se debe de abrir el editor

Special Requirements

User Interface

La interfaz deberá ser amigable e intuitiva para el usuario.

Security

El sistema deberá asegurar los datos de los usuarios.

Issues

To do

Abrir el editor.

CASO DE USO: COMPARTIR ARCHIVOS PARA TRABAJAR DE MANERA COLABORATIVA

Revision History

Date	Author	Description of change
09-octubre-2020	Ramiro Vidal Lumbreras	Descripción general del caso de uso

16-octubre-2020	Ramiro Vidal Lumbreras	Actualización de ID, Main Success, Extensions
-----------------	---------------------------	---

Use Case: Compartir archivos para trabajar de manera colaborativa o compartir con usuarios.

Id: UC-022

Description:

El software permite a los usuarios compartir documentos entre los demás usuarios para así poder trabajar de manera colaborativa.

Level: High Level

Primary Actors:

Alumnos - Docentes

Supporting Actors:

Stakeholders and Interests:

Principalmente a los alumnos porque son los que tienen que realizar trabajos en equipos o de manera colaborativa, pero también al profesor para supervisar aspectos que él considere hacia a sus alumnos.

Pre-Conditions:

Debe de cumplir con: UC-001, UC-002 , UC-003 y UC-004.

Post Conditions:

Success end condition:

- Un usuario es registrado para trabajar de manera colaborativa con otros.

Failure end condition:

- El usuario no tenga conexión a internet.
- No termine de agregar a los usuarios para compartir el documento.

Minimal Guarantee:

- En caso de no terminar el proceso para compartir archivos, se queda pendiente para su finalización.
- El usuario con el que se comparte el archivo podrá aceptar o no si quiere trabajar de manera colaborativa.

- Se le notifica al usuario si se comparte con él algún archivo.

Trigger:

Si el usuario requiere compartir un archivo con otros usuarios.

Main Success Scenario

Suceso 1

1. Los usuarios entran a archivos.
2. Eligen un documento para compartir.
3. Escogen la opción de compartir.
4. Agregan a los usuarios.
5. Y click en la opción “LISTO”

Suceso 2

1. Los usuarios entran a archivos.
2. Crean un documento.
3. Eligen el formato para crear el archivo.
4. Escogen la opción de compartir.
5. Agregan a los usuarios.
6. Y click en la opción “LISTO”

Suceso 3

1. Los usuarios entran a archivos.
2. Cargan un documento para compartir.
3. Escogen la opción de compartir.
4. Agregan a los usuarios.
5. Y click en la opción “LISTO”

Extensions

Suceso 1:

2a. En caso de no tener un documento.

1. Se creará un nuevo archivo.

4a. En caso de exceder de usuarios para agregar.

1. Mensaje notificando que ya se excedió el límite.
2. Quitar usuarios.

Suceso 2:

5a. En caso de exceder de usuarios para agregar.

1. Mensaje notificando que ya se excedió el límite.
2. Quitar usuarios.

Suceso 3:

4a. En caso de exceder de usuarios para agregar.

1. Mensaje notificando que ya se excedió el límite.
2. Quitar usuarios.

Variations

Agregar usuarios para compartir ya sea por:

- Nombre completo
- Matrícula
- Correo institucional

Frequency: 0 a 3 veces al día.

Assumptions:

- El usuario tiene idea donde está el apartado de archivos.
- El usuario sabe cómo compartir archivos.
- El usuario tiene los datos de los usuarios a agregar.

Special Requirements:

User interface

- Interfaz amigable con el usuario.

Security

- El usuario que no quiera trabajar de manera colaborativa se puede deslindar del archivo.
- El usuario que sea dueño del archivo será el único en agregar a más usuarios.

Issues:

- Máximo de 20 personas para colaborar al mismo tiempo

To do:

- Hacer que varios usuarios trabajen de manera colaborativa en un documento o archivo.
- En caso de ser un archivo diferente a un documento podrá visualizar el archivo compartido.

CASO DE USO: DAR DE ALTA ALUMNOS DEL CURSO

Revision History

Date	Author	Description of change
07-octubre-2020	Erick Nuñez Grajales	Descripción general del caso de uso
16-Octubre-2020	Erick Nuñez Grajales	Modificación ID

Use Case: Dar de alta alumnos del curso

Id: UC- 023

Description:

El docente tiene el poder para dar de alta nuevos alumno al curso en caso de que no estuvieran desde un principio.

Level:

Mid-Level Summary

Primary Actors:

Docente.

Supporting Actors

No se presentan en este caso.

Stakeholders and Interests

Docentes - Interés en poder agregar estudiantes a su curso.

Pre-Conditions

El usuario debe de cumplir con UCC-01

Post Conditions:

Success end condition:

El docente habrá podido agregar correctamente al usuario con éxito.

Failure end condition:

El docente no pudo agregar al alumno al curso.

Minimal Guarantee

El alumno, recibirá una invitación para poder ver las clases al menos desde una perspectiva de espectador, y en caso de dar de baja, se podrá bloquear a ese alumno prohibiendo el acceso a clases, tareas, etc...

Trigger

El profesor desea dar de alta a un alumno.

Main Success Scenario

1. El docente ingresa al portal académico
2. Ingresa al grupo correspondiente donde desea dar de alta dicho alumno
3. El docente ingresará a los miembros del grupo
4. El docente registrará al nuevo alumno utilizando su matrícula

Extensions

3a. En caso de no haber encontrado la matrícula

1. El sistema notificará que no se encuentra la matrícula.
2. El docente creará en la parte de registrar alumnos, un link el cual provisionará la posibilidad de integrarse al grupo en lo que se encuentra solución al error de matrícula.
3. El docente le enviará el link al alumno.

Variations: No aplica

Frequency:

Las veces que lo requiera el docente

Assumptions

El docente tiene conocimiento previo de la plataforma.

Special Requirements

User Interface

1. La interfaz deberá ser amigable e intuitiva para el docente.

Performance

1. El sistema debe de funcionar correctamente para dar de alta al alumno.

Issues

1. El sistema puede llegar a tener un cupo máximo de estudiantes por grupo.

To do

1. Registrar base de datos

CASO DE USO: DAR DE BAJA ALUMNOS DEL CURSO

Revision History

Date	Author	Description of change
07-octubre-2020	Erick Nuñez Grajales	Descripción general del caso de uso
16-Octubre-2020	Erick Nuñez Grajales	Modificación ID

Use Case: Dar de baja alumnos del curso

Id: UC- 024

Description:

El docente tiene el poder para dar de baja al alumno del curso en caso de que así lo requiera.

Level:

Mid-Level Summary

Primary Actors:

Docente, alumnos.

Supporting Actors

No se presentan en este caso.

Stakeholders and Interests

Docentes - Interés en poder deslindar estudiantes a su curso.

Pre-Conditions

El usuario debe de cumplir con UCC-01

Post Conditions:

Success end condition:

El usuario se ha podido registrar con éxito.

Failure end condition:

El usuario no se ha podido registrar con éxito.

Minimal Guarantee.

Trigger

El usuario ya no desea registrarse..

Main Success Scenario

1. El usuario ingresa al portal académico
2. Ingresa al grupo correspondiente donde se encuentra el alumno a dar de baja.
3. El docente ingresará a los miembros del grupo
4. El docente busca al alumno
5. Selecciona opciones encontrada a un lado del nombre del alumno
6. Oprime en Eliminar del curso

Extensions

6a. En caso de no haber podido deslindar al alumno

 1. El profesor oprimira bloquear en lugar de Eliminar del curso

 2. El alumno no podrá ver lo que se pondrá en el grupo de la materia ni comentar

Frequency:

Las veces que lo requiera el docente

Assumptions

El docente tiene conocimiento previo de la plataforma.

Special Requirements

User Interface

1. La interfaz debe de ser amigable y fácil de usar.

Performance

1. El sistema debe de funcionar correctamente para dar de alta al alumno.

To do

1. Actualizar base de datos

CASO DE USO: ASIGNACIÓN DE TAREAS.

Revision History

Date	Author	Description of change
09-octubre-2020	Martín A. Paniagua Velázquez.	Descripción general del caso de uso

Use Case: Asignar una tarea.

Id: UC-025

Description

El sistema permitirá a los usuarios asignar una o más tareas en un curso determinado.

Level: High Level Summary.

Primary Actors

Docente.

Supporting Actors

Alumno.

Stakeholders and Interests

Alumnos - Tienen interés en recibir actualizaciones de las tareas asignadas.

Pre-Conditions

El usuario debe de cubrir con UC-001 y con UC-002, además de estar inscrito a un curso.

Post Conditions

Success end condition

El usuario podrá asignar una tarea.

Failure end condition:

El usuario no podrá asignar la tarea de manera correcta.

Minimal Guarantee

El sistema manda una notificación a los usuarios inscritos de que se hizo un intento de asignar la tarea.

Trigger

El usuario desea asignar una tarea a los alumnos.

Main Success Scenario

1. El actor coloca la liga en su navegador.
2. El sistema solicita el correo y contraseña registrado.
3. El actor ingresa su correo institucional y su contraseña.
4. El actor da click en Ingresar.
5. El sistema permite el acceso al usuario.
6. El actor selecciona el botón de Cursos.
7. El actor selecciona el curso deseado.
8. El sistema muestra la pantalla de inicio del curso.
9. El actor selecciona Asignar Tarea.
10. El sistema solicita la información de la tarea.
11. El actor escribe la información y da click en Asignar.
12. El sistema manda la notificación de la tarea.

Extensions

- 4a. En el paso 4, si el usuario colocó una contraseña errónea.
7. El sistema muestra un mensaje que notifica que la contraseña es incorrecta.
 8. El actor coloca una contraseña diferente.
 9. El caso de uso continúa en el paso 4.
- 11a. En el paso 11, si existe un error técnico.
7. El sistema muestra un mensaje notificando que hay un error.
 8. El sistema da la opción de intentar de nuevo.
 9. El caso de uso continúa en el paso 11.

Variations

No hay variaciones.

Frequency: 1 vez por tarea.

Assumptions

El usuario tiene una cuenta, el usuario está inscrito a un curso.

Special Requirements

User Interface

1. Interfaz amigable e intuitiva para el usuario.

Security

1. El sistema va a asegurar que la tarea sea enviada solamente a los usuarios inscritos al curso.

Issues

3. ¿Habrá diversas categorías de usuarios que cuenten con limitaciones en el curso?

To do

Registrar la tarea en la base de datos.

Notificar a los usuarios inscritos de la tarea asignada.

CASO DE USO: MODIFICAR ENTREGA DE UNA TAREA

Revision History

Date	Author	Description of change
08-octubre-2020	Alhelí Moreno Gaytán	Descripción general del caso de uso
16-octubre-2020	Alhelí Moreno Gaytán	Actualización de ID.

Use Case: Modificar entrega de una tarea

Id: UC- 026

Description

El usuario podrá modificar una entrega de tarea mientras ésta se encuentre activa.

Level: High Level

Primary Actors

Alumno

Supporting Actors

No aplica

Stakeholders and Interests

Docente: tiene interés por conocer cuántas modificaciones se hicieron en las tareas.

Pre-Conditions

Los actores deberán cubrir los casos de uso UC-001, UC-002, UC-010, además de estar inscritos en al menos un curso.

La tarea deberá estar activa.

Post Conditions

Success end condition:

- El alumno realiza la modificación con éxito.
- El sistema arroja un mensaje de modificación exitosa incluyendo la fecha y la hora.
- El sistema notifica la modificación al docente del curso correspondiente.

Failure end condition:

- El alumno no concreta la modificación de la tarea.

Minimal Guarantee

- El sistema permite la carga de archivos y la visualización correcta de las tareas asignadas.

Trigger

El alumno quiere hacer una modificación en la entrega de una tarea.

Main Success Scenario

1. El alumno da clic en la tarea de interés.
2. El alumno da clic en “modificar entrega”.
3. El sistema permite al alumno agregar un archivo desde su dispositivo.
4. El alumno selecciona la opción deseada.
5. El alumno da clic en “adjuntar archivo”.
6. El sistema carga el archivo.
7. El sistema muestra los archivos adjuntos en la pantalla.
8. El alumno revisa los archivos adjuntos y da clic en “entregar tarea”.
9. El sistema pide confirmación de entrega.
10. El alumno confirma la entrega.
11. El sistema muestra un mensaje de entrega exitosa.

Extensions

6a. En el paso 6, el sistema no puede cargar el archivo.

1. El sistema notifica el fallo y solicita al alumno verificar el archivo.
2. El caso de uso continúa en el paso 3.

10a. En el paso 10, el alumno no confirma la entrega.

1. El sistema vuelve a la pantalla inicial de la tarea.

Variations

8'. En el paso 8 el alumno puede revisar los archivos y eliminar los que no quiera entregar.

1. El caso de uso continúa en el paso 9.

Frequency: 3 veces al mes.

Assumptions

- El alumno sabe cómo buscar un archivo en su dispositivo.

Special Requirements

User Interface

1. La interfaz deberá ser amigable e intuitiva para el usuario.

Security

1. Los archivos cargados sólo podrán ser accedidos por el alumno que los adjuntó y el docente asociado al curso.

Issues

1. ¿Cuántas modificaciones se pueden hacer por tarea?

To do

1. Actualizar base de datos e información de entrega.

CASO DE USO: ASIGNAR CALIFICACIÓN A TAREA.

Revision History

Date	Author	Description of change
09-octubre-2020	Martín A. Paniagua Velázquez.	Descripción general del caso de uso

Use Case: Asignarle una calificación a una tarea.

Id: UC-027

Description

El sistema permitirá a los usuarios asignar una calificación a las tareas que aparezcan dentro del sistema.

Level: High Level Summary.

Primary Actors

Docente.

Supporting Actors

Alumno.

Stakeholders and Interests

No hay en este caso.

Pre-Conditions

El usuario debe de cubrir con UC-001 y con UC-002, además de estar inscrito a un curso y haber asignado por lo menos una tarea previamente.

Post Conditions

Success end condition

El usuario podrá asignar una calificación a las tareas.

Failure end condition:

El usuario no podrá asignar calificaciones.

Minimal Guarantee

La calificación no es enviada hasta que el proceso esté completamente finalizado y verificado por el usuario.

Trigger

El usuario quiere darle una calificación a las tareas.

Main Success Scenario

1. El actor coloca la liga en su navegador.
2. El sistema solicita el correo y contraseña registrado.
3. El actor ingresa su correo institucional y su contraseña.
4. El actor da click en Ingresar.
5. El sistema permite el acceso al usuario.
6. El actor selecciona el botón de Cursos.
7. El actor selecciona el curso deseado.

8. El sistema muestra la pantalla de inicio del curso.
9. El actor selecciona Tareas asignadas.
10. El sistema muestra las tareas pendientes por calificar.
11. El actor selecciona una de las tareas.
12. El sistema muestra la tarea.
13. El actor selecciona Calificar.
14. El sistema solicita una calificación y una retroalimentación.
15. El actor coloca los datos y le da click a Calificar.
16. El sistema envía una notificación de la calificación.

Extensions

4a. En el paso 4, si el usuario colocó una contraseña errónea.

1. El sistema muestra un mensaje que notifica que la contraseña es incorrecta.
2. El actor coloca una contraseña diferente.
3. El caso de uso continúa en el paso 4.

15a. En el paso 15, si existe un error técnico.

1. El sistema muestra un mensaje notificando que hay un error.
2. El sistema da la opción de intentar de nuevo.
3. El caso de uso continúa en el paso 15.

Variations

La calificación puede seleccionarse de manera táctil.

Frequency: 1 vez por tarea.

Assumptions

El usuario tiene una cuenta, el usuario está inscrito al menos a un curso, el usuario ya asignó una tarea previamente y el tiempo ya venció, por lo que hay tareas pendientes por calificar.

Special Requirements

User Interface

1. Interfaz amigable e intuitiva para el usuario.

Security

1. El sistema va a asegurar que la calificación asignada no se puede modificar por ningún otro usuario.

Issues

4. ¿Se pueden calificar tareas con retraso?

To do

Registrar la calificación en la base de datos.

Notificar a los alumnos su calificación.

CASO DE USO: NOTIFICAR SOBRE UN ERROR EN LA CALIFICACIÓN

Revision History

Date	Author	Description of change
08-octubre-2020	Jose Carlos Mejia Ramirez	Descripción general del caso de uso

Use Case: Notificar sobre un error en la calificación.

Id: UC- 028

Description

El sistema contará con una opción para que el alumno pueda notificar al docente sobre un error en una calificación de alguna actividad.

Level: High-Level

Primary Actors

Alumno

Supporting Actors

Docente

Stakeholders and Interests

Alumnos que soliciten aclaración sobre su calificación.

Pre-Conditions

Debe cumplir primero con UC-001,UC-002,UC-003,UC-010

Post Conditions

Failure end condition:

- El alumno no puede notificar sobre el error.

Minimal Guarantee:

- Puede solicitar una corrección por un mensaje directo.

Trigger

El alumno entrega una actividad (UC-10).

Main Success Scenario

1. El alumno requiere una aclaración.
2. Notifica la aclaración al docente.
3. El profesor analiza su petición.
4. Hace un cambio en la calificación de la actividad.

Extensions

2a. El alumno no puede notificar la aclaración.

1. El sistema recomendará el uso de un mensaje directo para la aclaración.

Variations

1. El alumno pide una aclaración por error.

1. El sistema dará la opción de cancelar la aclaración.
2. El alumno cancela la aclaración.
3. Se elimina la petición.

Frequency: 1 vez al día.

Assumptions

El alumno entregó una tarea.

Special Requirements

User Interface

1. Mostrar la opción para poder solicitar una aclaración.

Security

1. No puedes pedir una aclaración a no ser que el trabajo haya sido calificado.

Issues

1. La herramienta no notifica al docente sobre la aclaración.

To do

1. Mostrar todas las actividades entregadas con sus respectivas calificaciones y valor.

CASO DE USO: PERSONALIZACIÓN DE ACTIVIDADES

Revision History

Date	Author	Description of change
08-octubre-2020	Jose Carlos Mejia Ramirez	Descripción general del caso de uso

Use Case: Personalización de actividades.

Id: UC- 029

Description

El sistema permitirá a los docentes establecer una actividad para poder manejar los grupos de actividades con un valor igualitario entre cada una dependiendo el tipo de actividad.

Level: Mid-Level

Primary Actors

Docente, Alumno

Supporting Actors

No cuenta con otros actores.

Stakeholders and Interests

Docentes que quieren agrupar las actividades para darles un orden, y los alumnos para llevar un mejor control de sus trabajos para poder priorizar.

Pre-Conditions

Debe cumplir primero con UCC-001,UCC-002,UCC-003

Post Conditions

Failure end condition:

- El docente no puede personalizar la actividad.

Minimal Guarantee:

- Creará una actividad más con un perfil default.

Trigger

El docente crea una actividad y el sistema le da la opción de elegir un perfil para la actividad.

Main Success Scenario

1. El docente crea la actividad con el perfil especificado con éxito y se anexa a las actividades ya existentes.
2. El alumno ve la actividad y el tipo de actividad asignada.

Extensions

1. En caso de que no se cree la actividad con el perfil especificado.
 1. Se creará una actividad marcada solo como “Actividad”.
 2. El alumno recibirá una actividad con el perfil default.
 3. El docente podrá cambiar después qué tipo de actividad es.

Variations

En lugar de darle un perfil específico a la actividad solo la crea como una “actividad sin perfil”

Frequency: 50 veces al dia.

Assumptions

El docente conoce cómo crear actividades.

El docente sabe cómo darle un perfil a la actividad.

Special Requirements

User Interface

1. La interfaz debe mostrar la opción de asignar perfil cuando se crea la actividad.

Security

1. Solo el docente puede añadir y modificar las actividades.

Issues

To do

1. Mostrar las actividades creadas posteriormente.

CASO DE USO: MODIFICACIÓN DE LA CALIFICACIÓN DE UNA ACTIVIDAD

Revision History

Date	Author	Description of change
08-octubre-2020	Jose Carlos Mejia Ramirez	Descripción general del caso de uso

Use Case: Modificación de la calificación de una actividad.

Id: UC- 030

Description

El docente podrá modificar la calificación a una actividad siempre y cuando sea reportada por un alumno.

Level: High-Level

Primary Actors

Docente

Supporting Actors

Alumno

Stakeholders and Interests

Alumnos que soliciten aclaración sobre su calificación y docentes que accedan a modificar la calificación por un error.

Pre-Conditions

Debe cumplir primero con UC-001,UC-002,UC-003,UC-010,UCC-205.

Post Conditions

Failure end condition:

- No puede ser modificada la calificación, no se altera la calificación actual.

Minimal Guarantee:

- El profesor tiene conciencia del cambio a la calificación.

Trigger

El alumno entrega una actividad (UC-10) y después solicita una aclaración (UCC-204)

Main Success Scenario

1. El alumno entrega una actividad.
2. El docente califica la actividad de manera errónea.
3. El alumno solicita una aclaración.
4. El docente accede y cambia la calificación.
5. La calificación es modificada con éxito.

Extensions

5a. El docente no puede cambiar la calificación.

1. La actividad se queda marcada como “aclaración”

Variations

1. El docente desea cambiar la calificación por un error de su parte.

1. Se notificará al alumno el cambio en su calificación.

Frequency: 2 veces a la semana.

Assumptions

El alumno nota un error en su calificación.

El profesor no nota un error en la nota asignada.

Special Requirements

User Interface

1. Mostrar que se ha solicitado una aclaración de un alumno en la actividad específica.

Issues

1. No es posible cambiar la calificación.

To do

1. Mostrar las actividades entregadas por cada alumno.

CASO DE USO:PERSONALIZAR PERFIL DE USUARIO.

Revision History

Date	Author	Description of change
08-octubre-2020	Jose Carlos Mejia Ramirez	Descripción general del caso de uso

Use Case: Personalizar perfil de usuario.

Id: UC- 031

Description

El sistema permitirá a tanto docentes como alumnos la personalización de su perfil.

Level: Low Level

Primary Actors

Alumno , docente

Supporting Actors

No se presentan en este caso.

Stakeholders and Interests

Tanto alumnos como docentes que quieran darle un estilo más personal a su cuenta.

Pre-Conditions

Debe cumplir con UCC-01

Post Conditions

Failure end condition:

- El usuario no puede crear su cuenta.

Minimal Guarantee:

- Contarán con una cuenta funcional pero estará establecida por defecto.

Trigger

El usuario debe indagar en la configuración de perfil para cambiar sus aspectos.

Main Success Scenario

1. El usuario tanto docente como alumno da click en configuración de perfil.
2. Elige los apartados de su agrado para cambiar.
3. Confirma los cambios hechos a su perfil.

Extensions

2a. En el caso que no quiera cambiar nada.

1. Se revertirán los cambios hechos.
2. Se retomarán los últimos cambios hechos.

3a. En caso de no tomar los cambios.

1. Se retomarán los últimos cambios hechos.
2. Se notificará a sistemas el error para poder ser arreglado.

Variations

Frequency: 10 veces al día.

Assumptions

- El usuario sabe como entrar a la configuración.

Special Requirements

User Interface

1. La interfaz debe dar claro que puedes entrar a configurar los apartados del perfil.

Security

1. No puedes cambiar la configuración del perfil a no ser que ya hayas ingresado primero.

Issues

1. No se elijan los formatos que soporte para la visualización del perfil.

To do

1. Actualizar los datos del perfil.

CASO DE USO: ACTUALIZAR DATOS

Revision History

Date	Author	Description of change
09-octubre-2020	Milcha Sarai Oropeza López	Descripción general del caso de uso
16-Octubre-2020	Milcha Sarai oropeza López	Modificación del ID

Use Case: Actualizar datos

Id: UC- 032

Description

Los usuarios podrán actualizar sus datos..

Level: Medium Level

Primary Actors

Alumno, Docente

Supporting Actors

Administrador

Stakeholders and Interests

Docente - tiene interés en actualizar sus datos

Alumno - tiene interés en actualizar sus datos.

Pre-Conditions

Tanto docente como alumno deberán cubrir con los casos de uso UC-001, UC-002..

Post Conditions

Success end condition:

- Los usuarios tendrán sus datos actualizados.

Failure end condition:

- El administrador no puede actualizar los datos.
- El docente y el alumno no ingresan datos correspondientes a los campos

Minimal Guarantee

- El sistema únicamente actualizará los datos del usuario

Trigger

Existen datos que actualizar .

Main Success Scenario

1. Los usuarios seleccionan la opción de modificar datos.
2. Colocan los datos que quieren modificar.
3. Dan click en actualizar.
4. La base de datos actualiza los datos de los usuarios.

Extensions

3a.Mensaje de error en caso de error en los campos.

1. El usuario verifica los datos.
2. El usuario vuelve al paso 2.

Variations: No se actualizan los datos

Frequency: 1 vez a la semana.

Assumptions

- El usuario sabe los datos que deben de ir en los campos..

Special Requirements

User Interface

La interfaz deberá ser amigable e intuitiva para el usuario.

Security

El sistema deberá asegurar los datos de los usuarios.

Issues

To do

Actualizar la base de datos.

CASO DE USO: CONFIGURACIÓN DE NOTIFICACIONES

Revision History

Date	Author	Description of change
09-octubre-2020	Ramiro Vidal Lumbreras	Descripción general del caso de uso
16-octubre-2020	Ramiro Vidal Lumbreras	Actualización de ID, Main Success, Extensions

Use Case: Configuración de notificaciones

Id: UC-033

Description:

El software permite a los usuarios recibir notificaciones incluso sin tener abierta o usando el sistema.

Level: High Level

Primary Actors:

Alumnos - Docentes

Supporting Actors:

Stakeholders and Interests:

Pre-Conditions:

Debe de cumplir con: UC-001, UC-002, UC-003 y UC-004.

Si quiere recibir una notificación de alguna actividad en particular: UC-006, UC-008, UC-016, UC-022, UC-026, UC-027, UC-028 y UC-030.

Post Conditions:

Success end condition:

- Algún usuario realice alguna actividad, reunión, o el docente crea una tarea, el usuario recibirá una notificación.

Failure end condition:

- El usuario no tenga conexión a internet.
- Desactivada la función de recibir notificaciones.

Minimal Guarantee:

- No tenga abierto el sistema, la notificación llegará.
- Las notificaciones se guardarán para visualizarlas cuando lo requiera.

Trigger:

La creación de un evento, actividad o tarea.

Main Success Scenario

1. Los usuarios entran a configuración.
2. Eligen el apartado de notificaciones.
3. Eligen si quieren recibir notificaciones cuando el sistema no esté abierto.
4. Eligen cuáles actividades van a recibir notificaciones.
5. Guardan la configuración

Extensions

2a. En caso de no hacer cambios.

1. Se mantendrá la configuración que tenía.

3a. En caso de no guardar la configuración

1. Los cambios no se realizarán
2. Se mantendrá la última configuración

Variations

Frequency: 5 veces al día o más.

Assumptions:

- El usuario tiene idea donde está el apartado de configuración.
- El usuario sabe como cambiar las notificaciones.

Special Requirements:

User interface

- Interfaz amigable con el usuario.

Security

- Solo el dueño de la cuenta podrá realizar el cambio a las notificaciones.

Issues:

To do:

- Activar las notificaciones para los usuarios cuando el sistema no esté en uso.

CASO DE USO: RESTABLECER CONTRASEÑA

Revision History

Date	Author	Description of change
09-octubre-2020	Milcha Sarai Oropeza López	Descripción general del caso de uso
16-octubre-202	Milcha Sarai Oropeza López	Modificación de ID

Use Case: Restablecer contraseña

Id: UC- 034

Description

Los usuarios podrán restablecer su contraseña, en dado caso que no se acuerden de este.

Level: Low Level

Primary Actors

Alumno, Docente

Supporting Actors

Administrador

Stakeholders and Interests

Docente - tiene interés en restablecer su contraseña

Alumno - tiene interés en restablecer su contraseña.

Pre-Conditions

Los actores deberán cubrir con los casos de uso UC-001, UC-002..

Post Conditions

Success end condition:

- Los usuarios tendrán acceso a su cuenta nuevamente.

Failure end condition:

- El administrador no puede hacer el restablecimiento de contraseña.
- El usuario no cuenta con los datos suficientes para restablecer su contraseña

Minimal Guarantee

- El sistema únicamente hará el cambio de contraseña

Trigger

Contraseña actualizada y restablecida .

Main Success Scenario

1. El usuario entra a la plataforma sin acceso a su contenido.
2. Selecciona la opción de restablecer contraseña.
3. El usuario da los datos que se requieren para restablecerla.
4. Se envía un mensaje al correo institucional con la nueva contraseña.
5. El usuario accede al contenido de la plataforma con la nueva contraseña.

Extensions

3a. El usuario no puede otorgar los datos que el sistema le solicita.

1. El usuario se comunica con el administrador

Variations: No se restablece la contraseña.

Frequency: 1 vez a la semana.

Assumptions

- El usuario conoce los datos requeridos para el restablecimiento de su contraseña.

Special Requirements

User Interface

1. La interfaz deberá ser amigable e intuitiva para el usuario.

Security

El sistema deberá asegurar los datos de los usuarios.

Issues

To do

Actualizar contraseña.

CASO DE USO: MOSTRAR GUÍA DE USUARIO

Revision History

Date	Author	Description of change
09-octubre-2020	Milcha Sarai Oropeza López	Descripción general del caso de uso
16-octubre-202	Milcha Sarai Oropeza López	Modificación de ID

Use Case:Mostrar guía de usuario

Id: UC- 035

Description

El usuario podrá seleccionar la opción de mostrar guía de usuario, en su primera visita a la plataforma.

Level: LOW-LEVEL

Primary Actors

Usuario

Supporting Actors

No aplica

Stakeholders and Interests

Alumnos y docentes que deseen dar una ligera introducción, con el fin de aprender sus funciones y lo que pueden o no hacer dentro de la plataforma

Pre-Conditions

Contar con los siguientes casos de uso : UC-001,UC-002

Post Conditions

Failure end condition:

No se requiere la visualización de la guía para usuario.

No se puede mostrar la guía

Minimal Guarantee:

- Los usuarios pueden visualizar la guía en su primera visita al sistema.

Trigger

La visualización de la guía

Main Success Scenario

1. El usuario entra a la plataforma.
2. Se visualiza la opción para ir a la guía de usuario
3. El usuario acepta la opción.
4. Se muestra la guía.

Extensions

Variations

3a. El usuario no acepta la guía.

1. Se manda un mensaje aclarando que el usuario no desea ver la guía.

Frequency: 1 sola vez

Assumptions

El usuario tiene idea de cómo funciona ya el sistema

El usuario no tiene una idea clara de las funciones del sistema.

Special Requirements

No aplica

User Interface

Issues

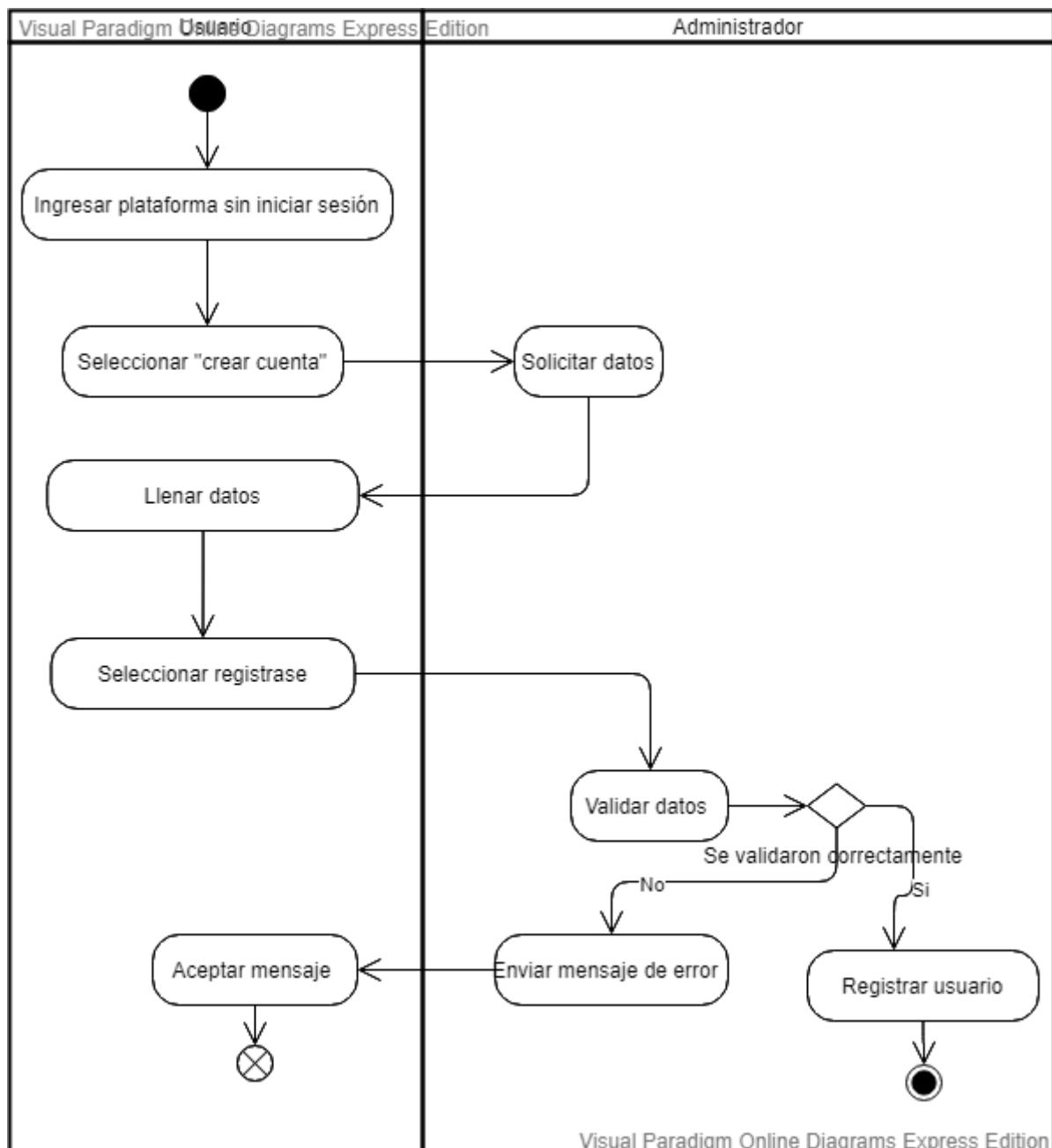
To do

1. Mostrar guía de usuario.

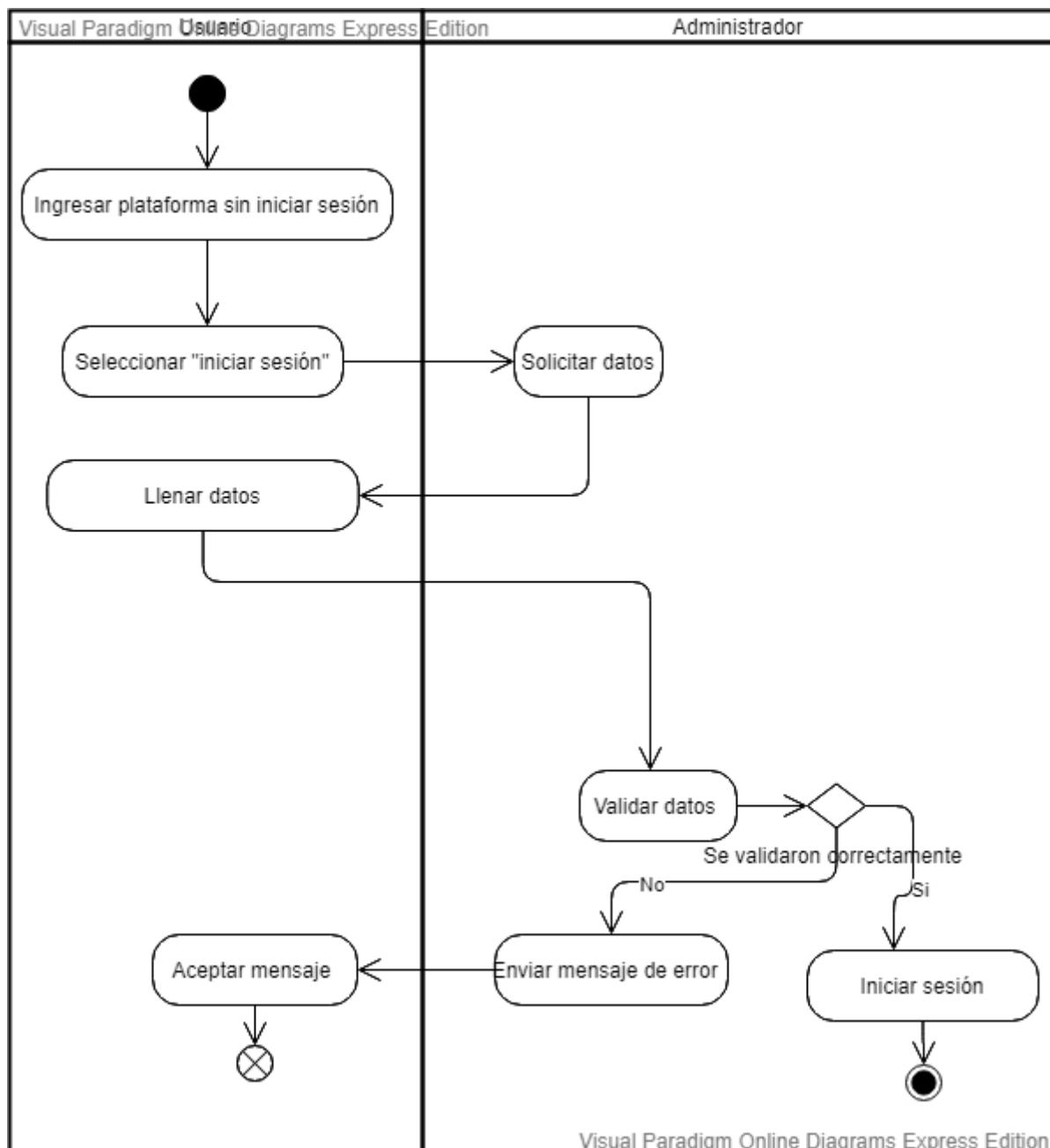
ANEXO 2. Diseño del sistema

DIAGRAMAS DE ACTIVIDADES

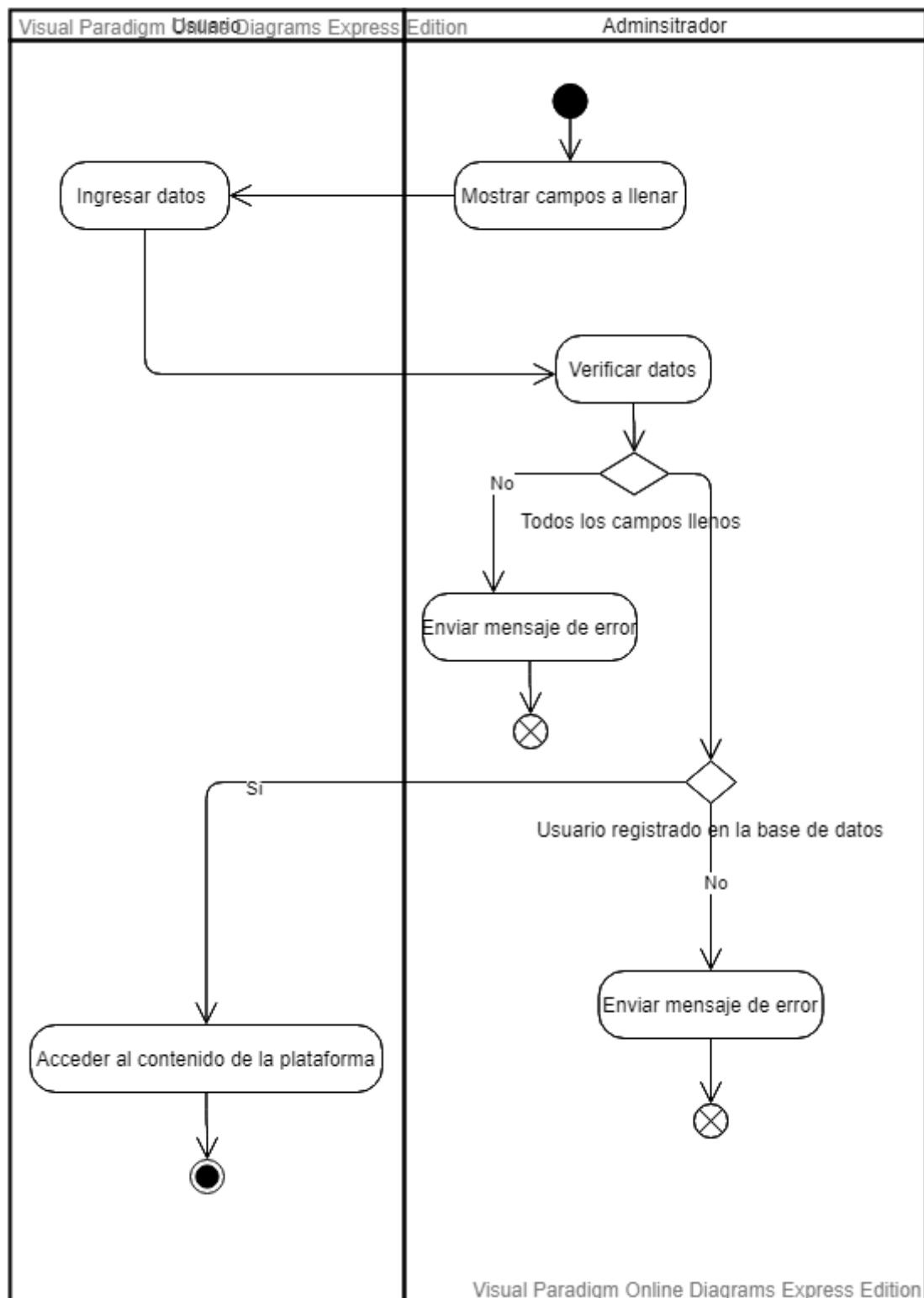
1. Crear cuenta



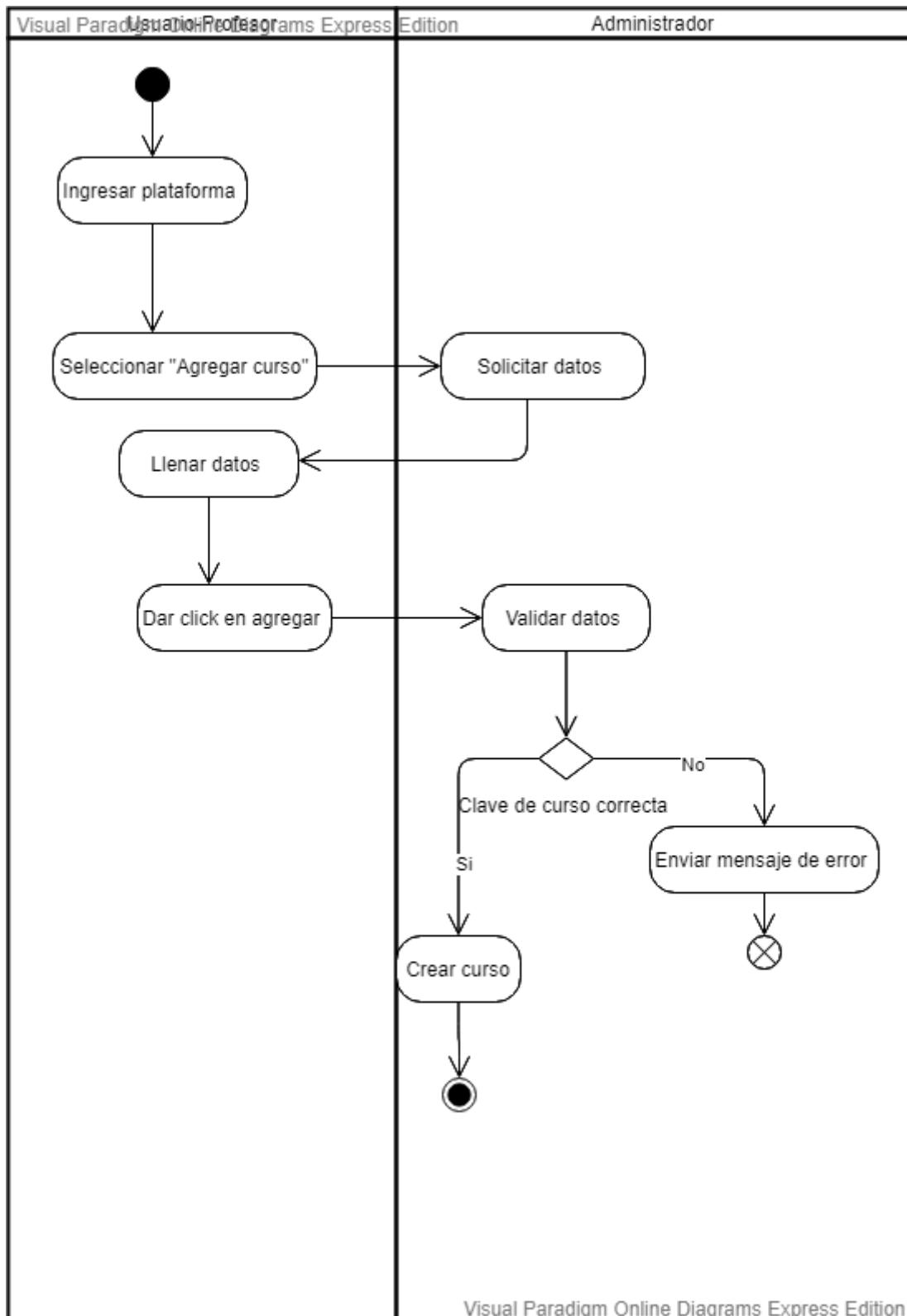
2. Iniciar sesión



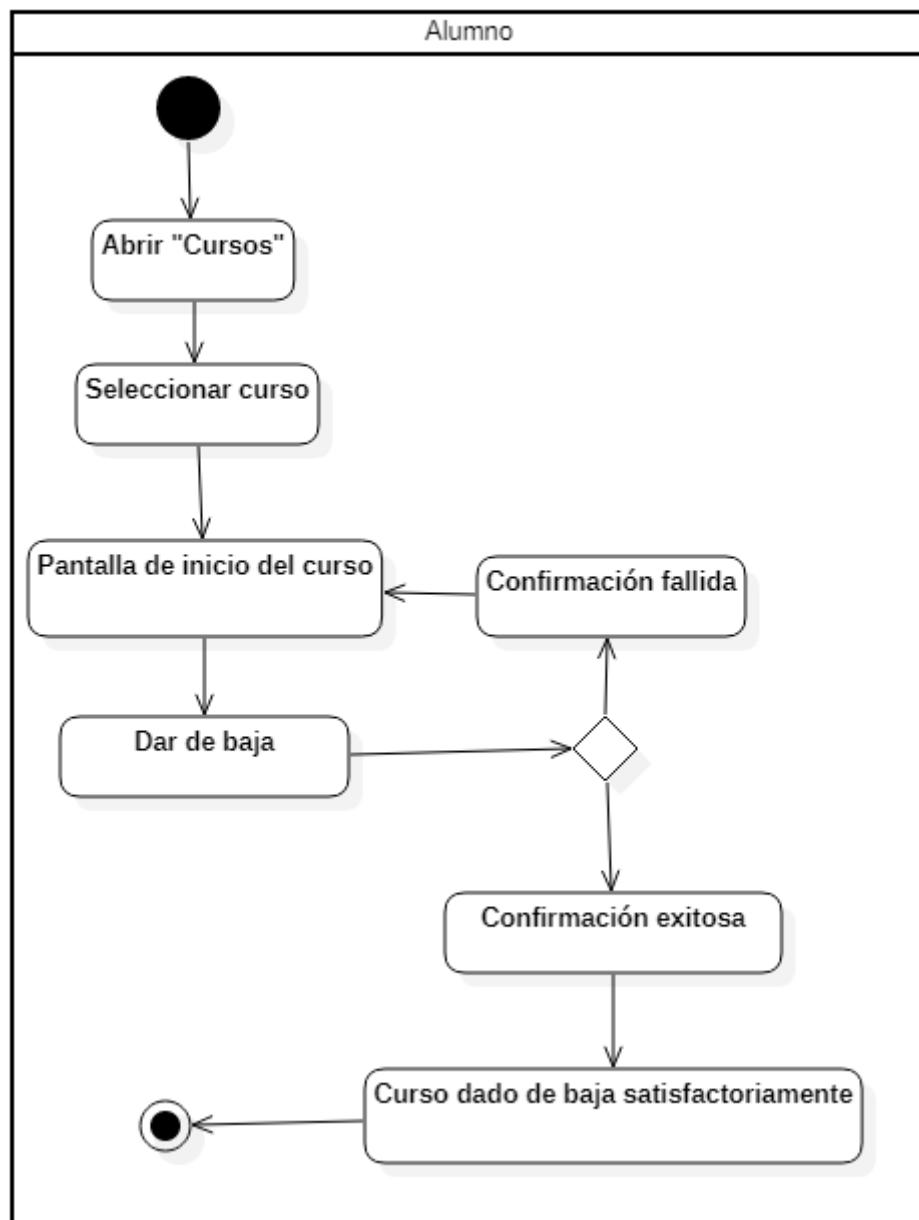
3. Validar usuario



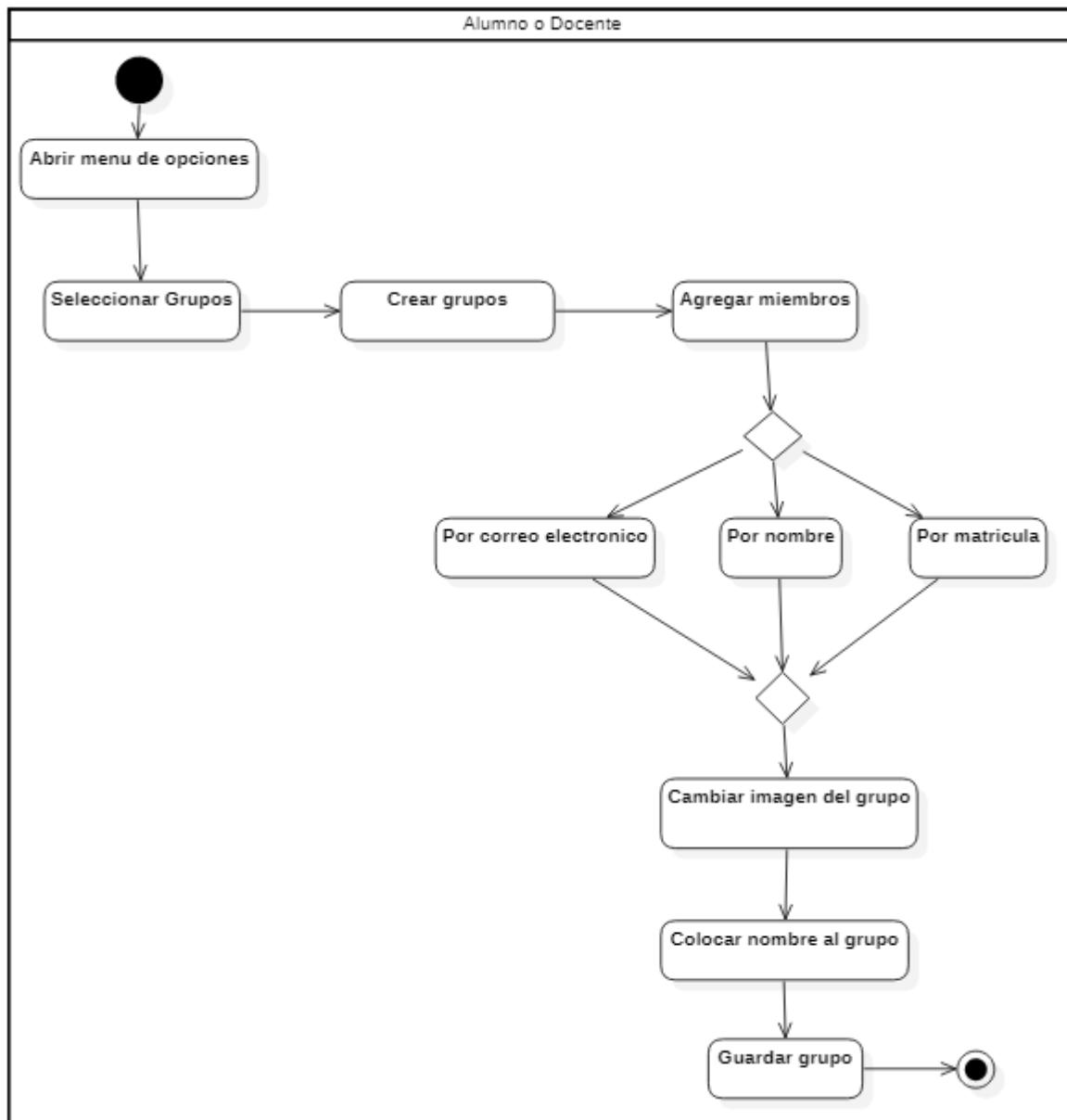
4. Inscripción a curso



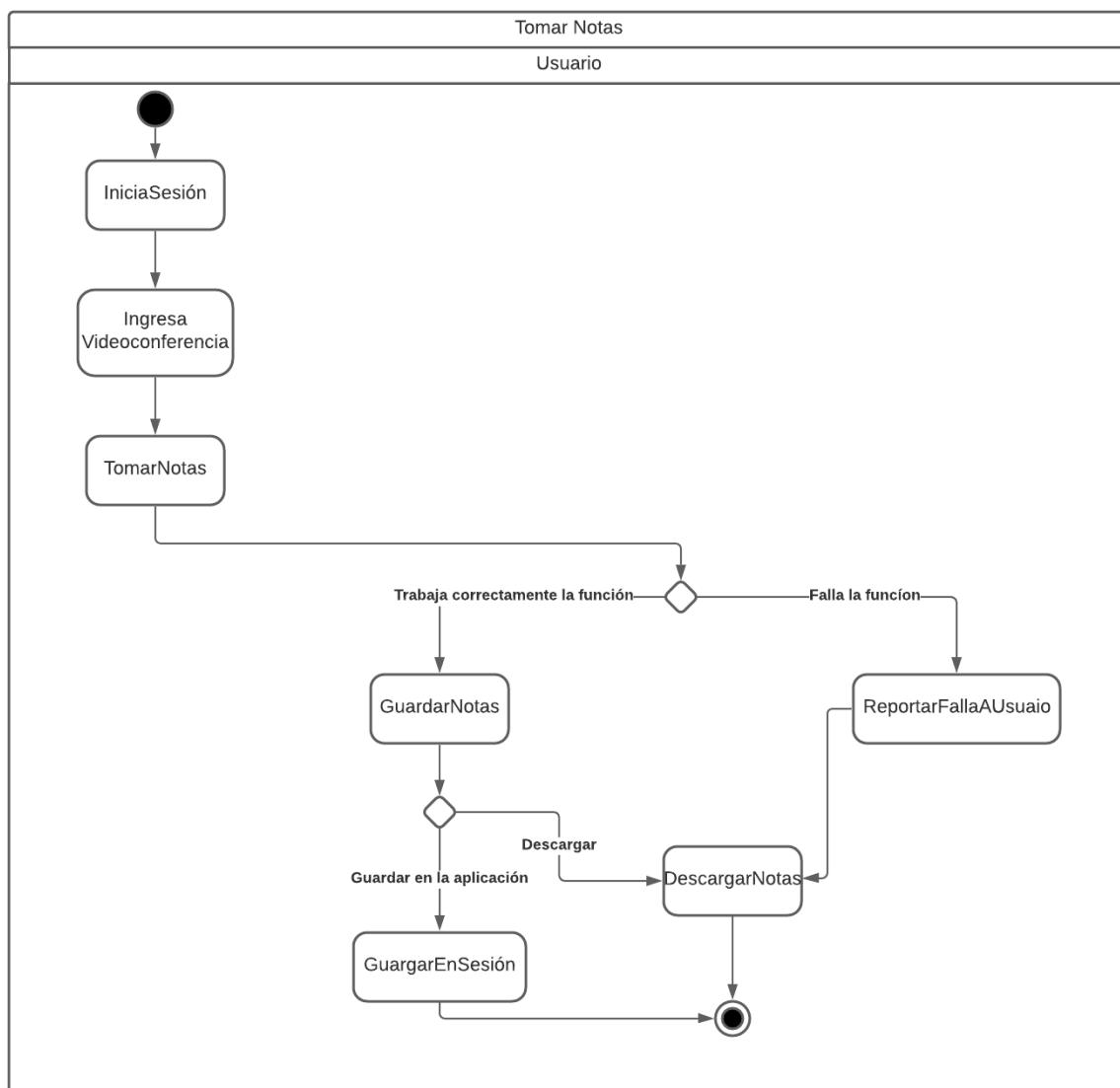
5. Dar de baja curso



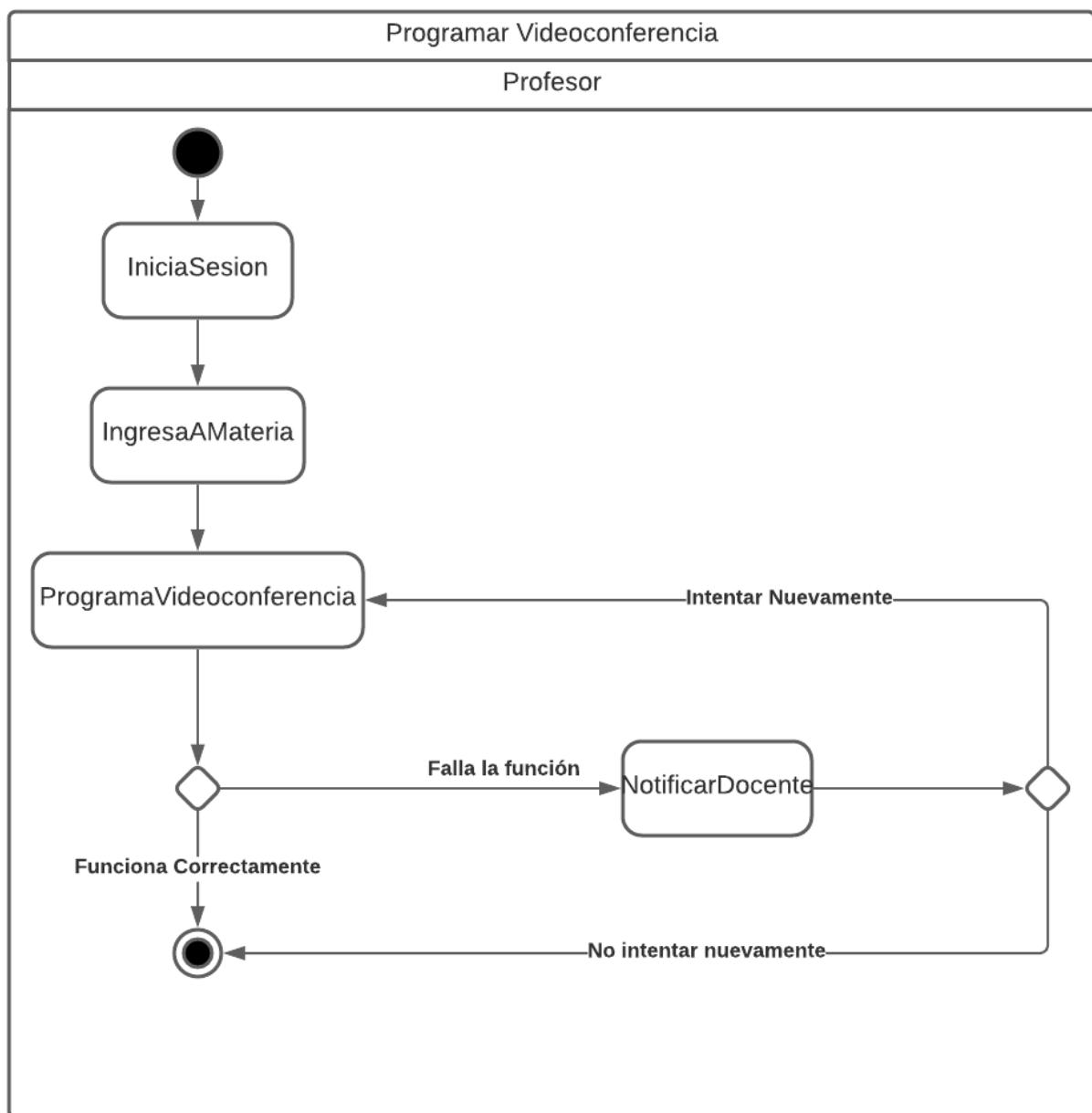
6. Crear grupos entre usuarios



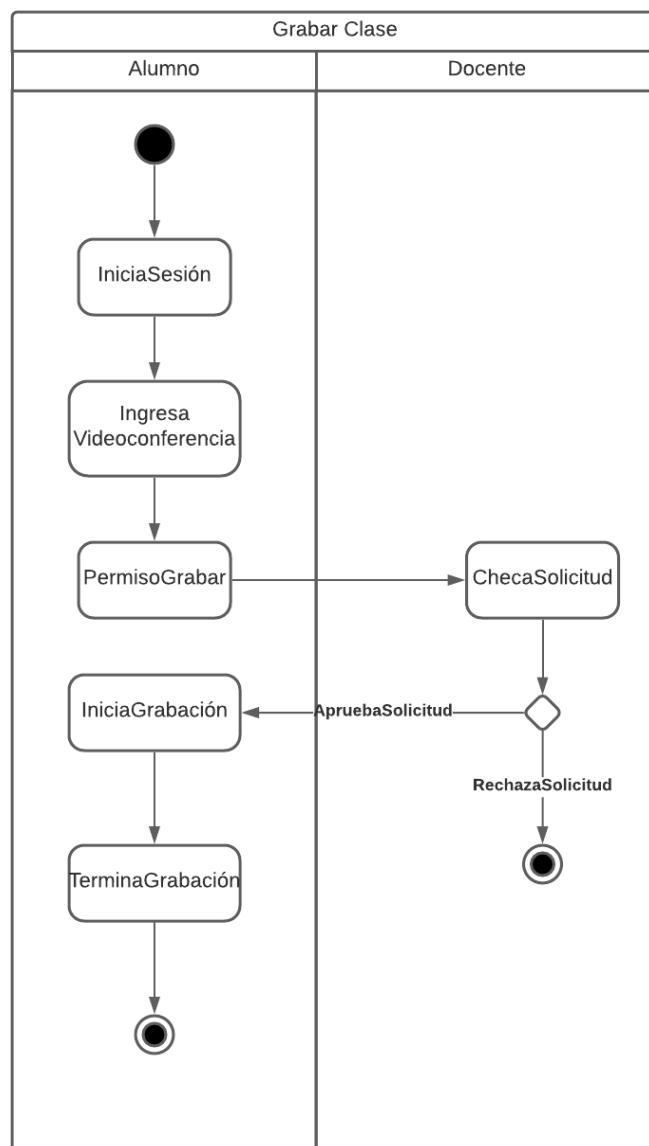
7. Creación de notas de sesión



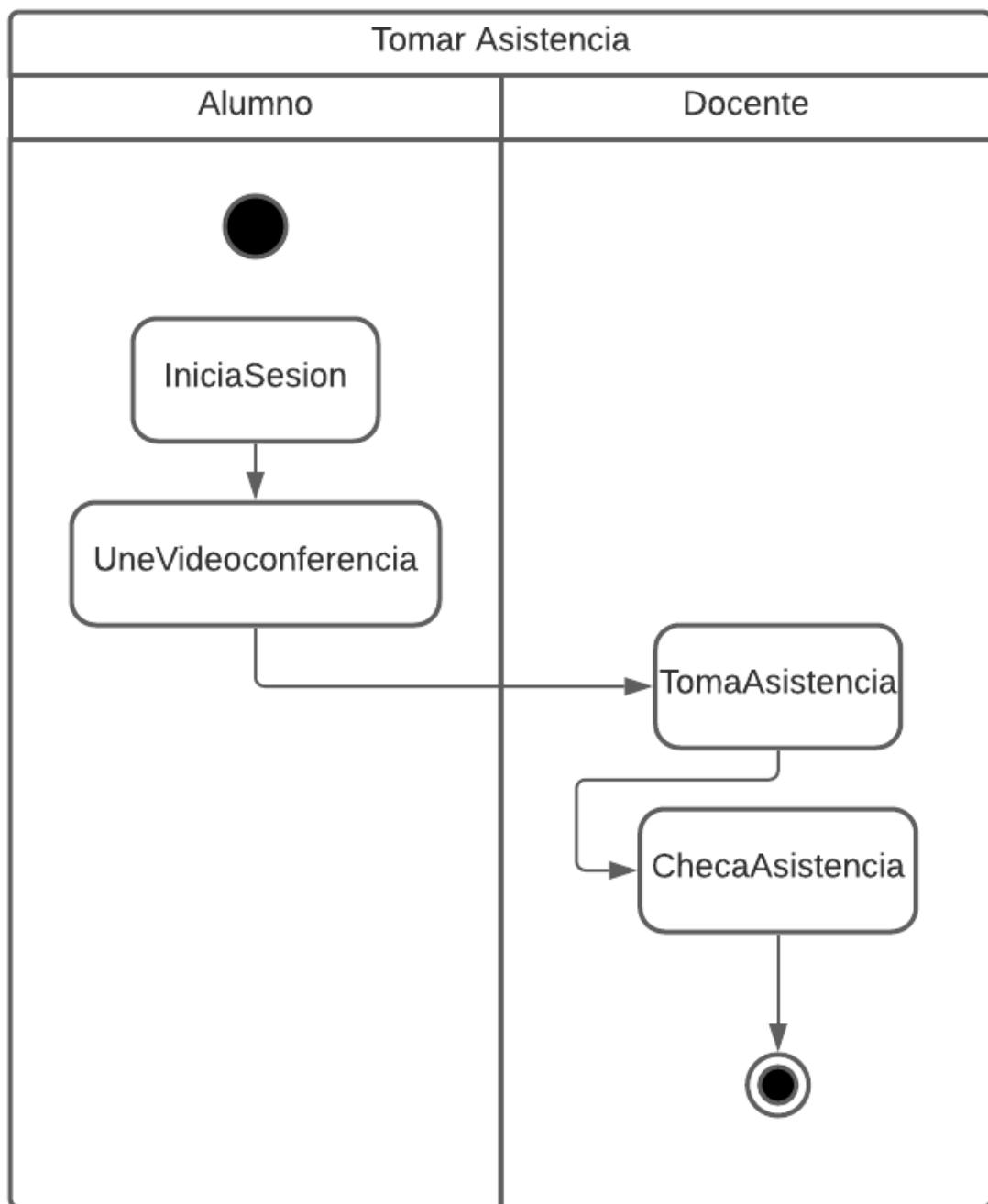
8. Programar videoconferencia



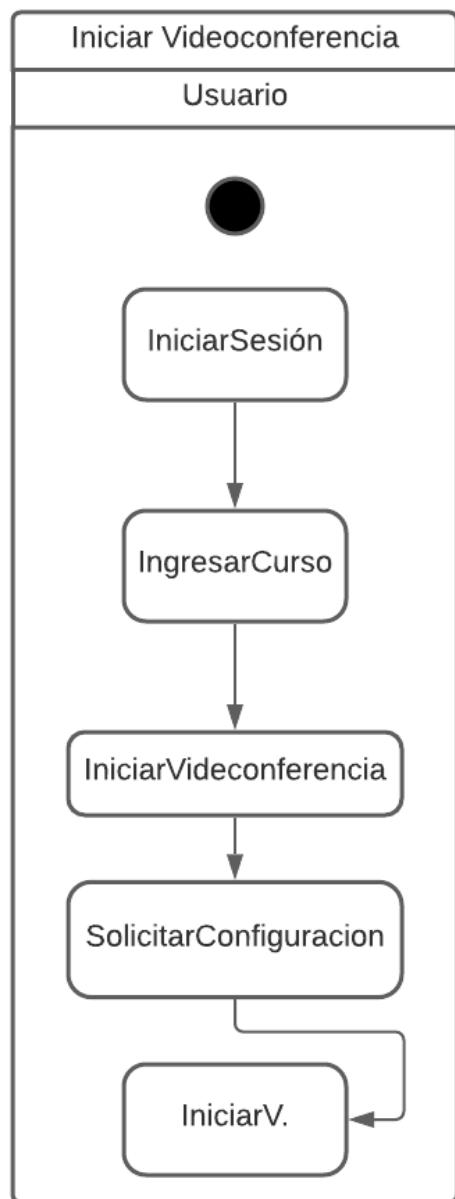
9. Alumnos-grabar clase



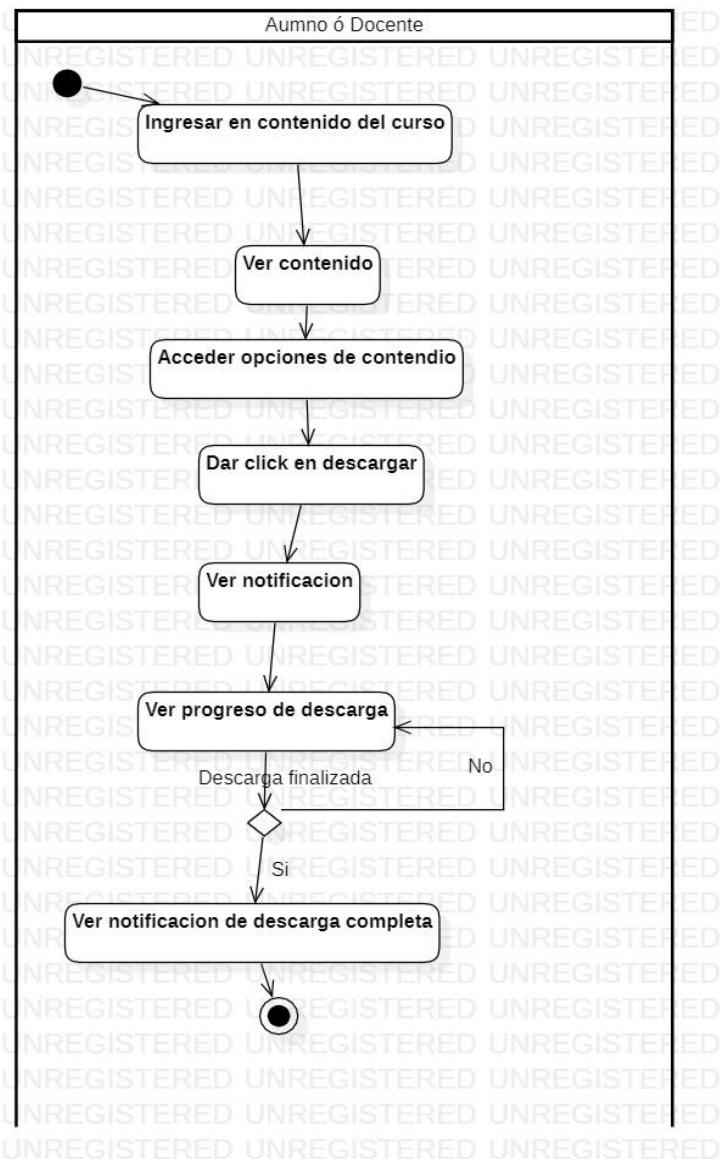
10. Revisar asistencia



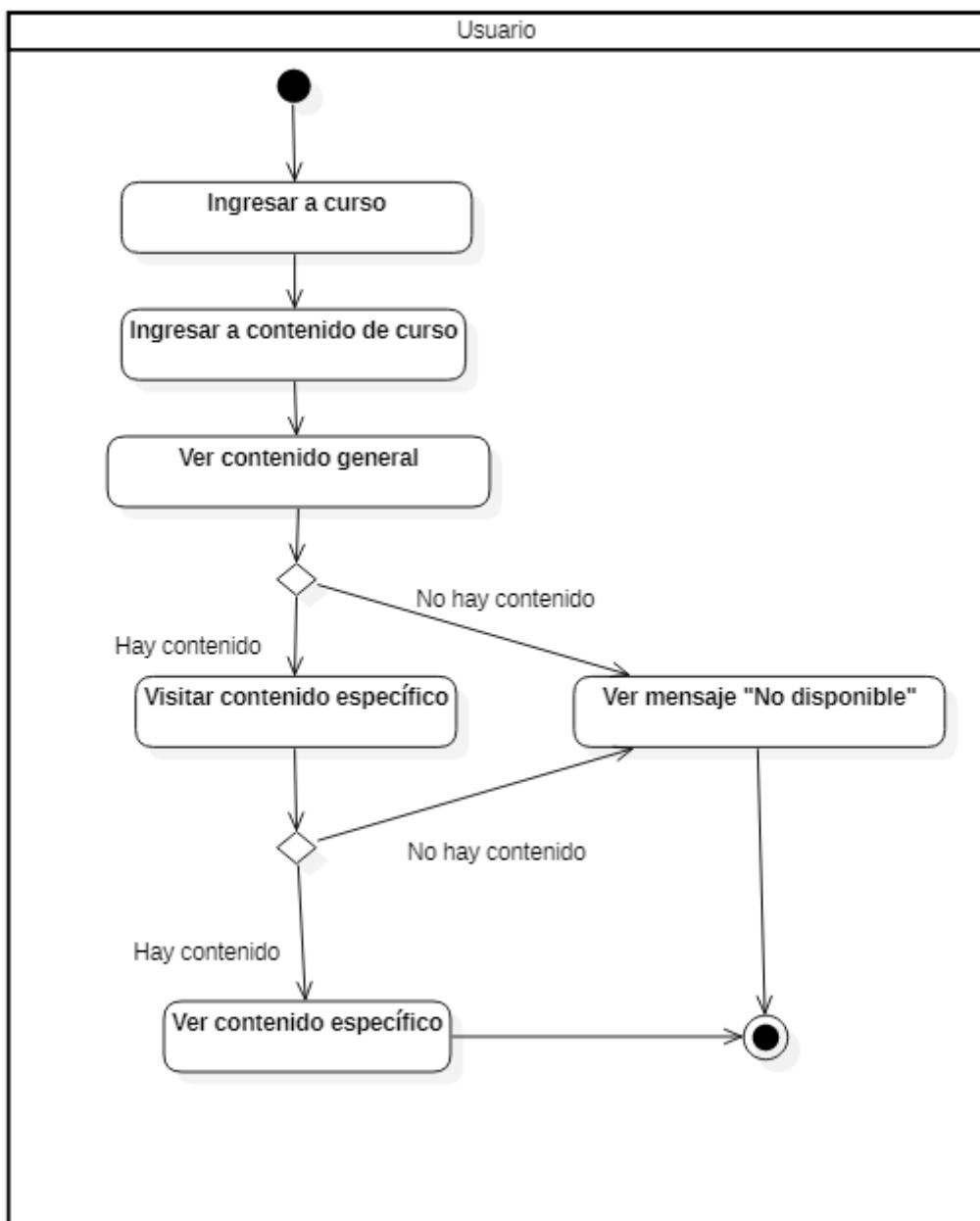
11. Iniciar videoconferencia



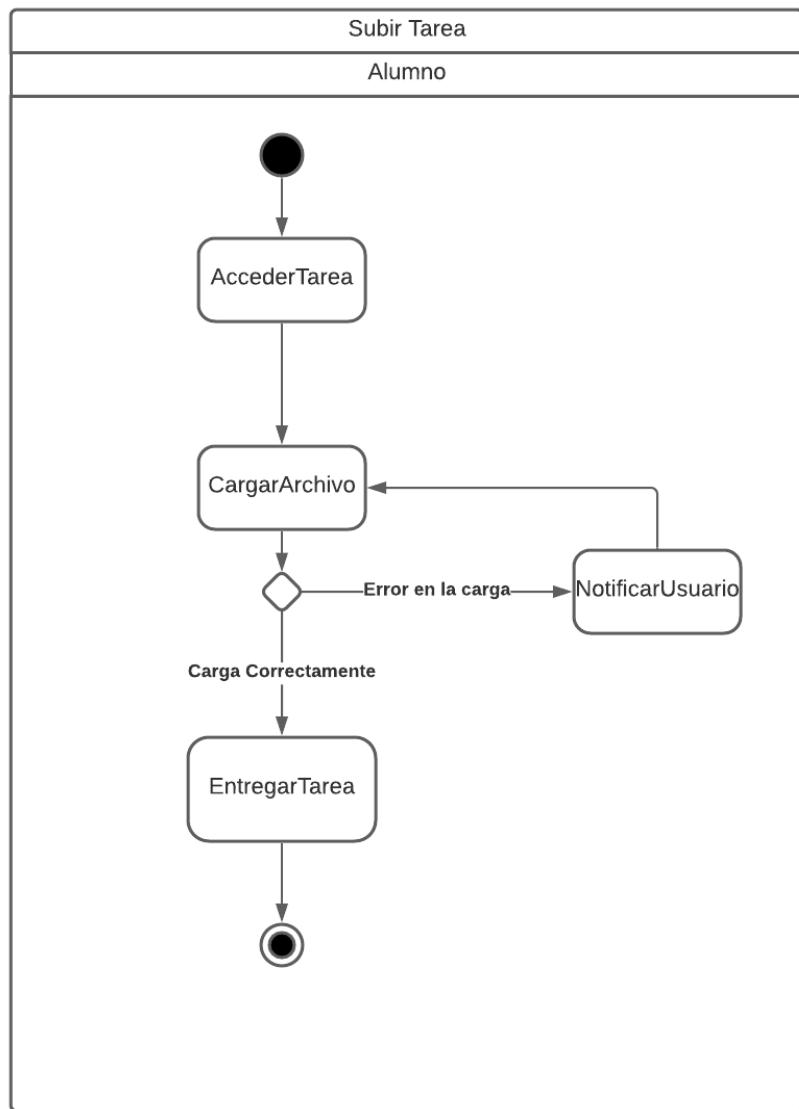
12. Descargar contenido



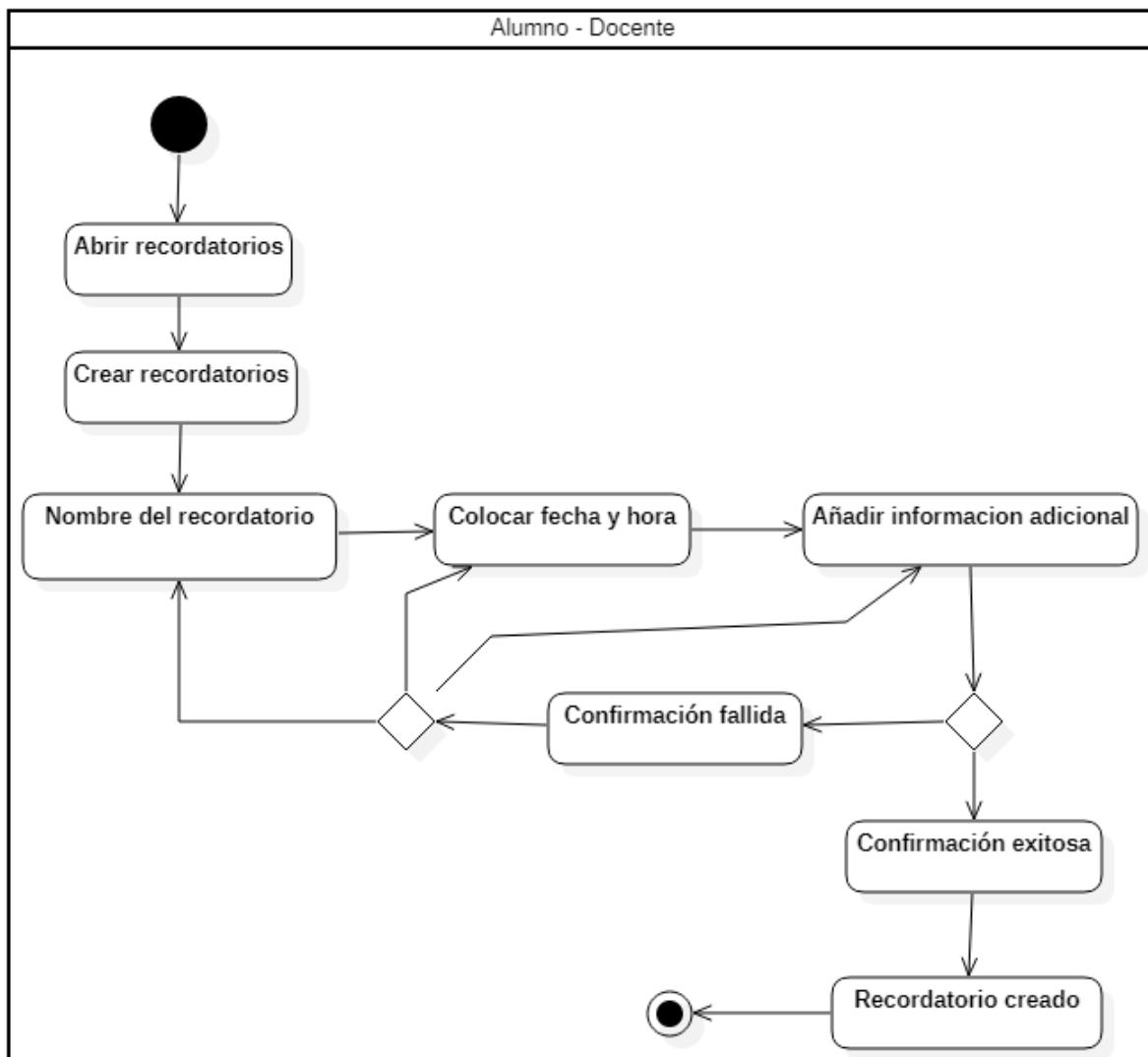
13. Revisar contenido del curso



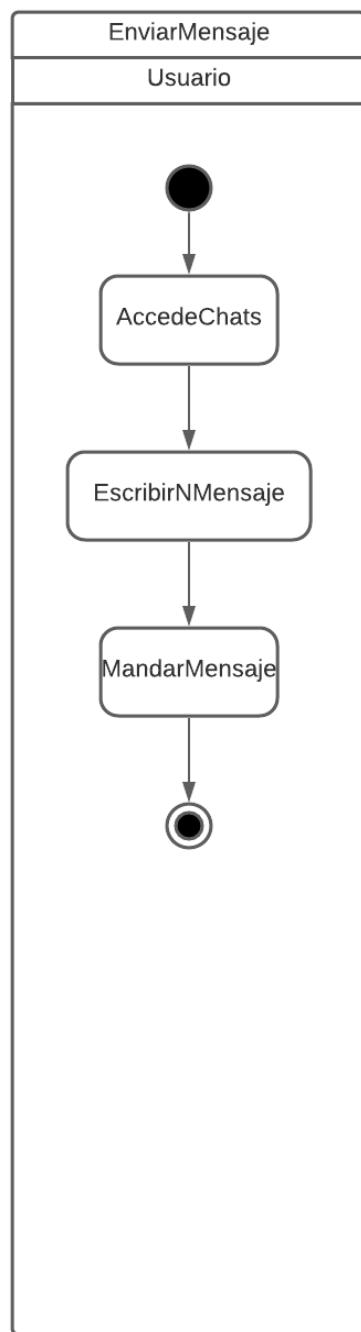
14. Realizar entrega de una tarea



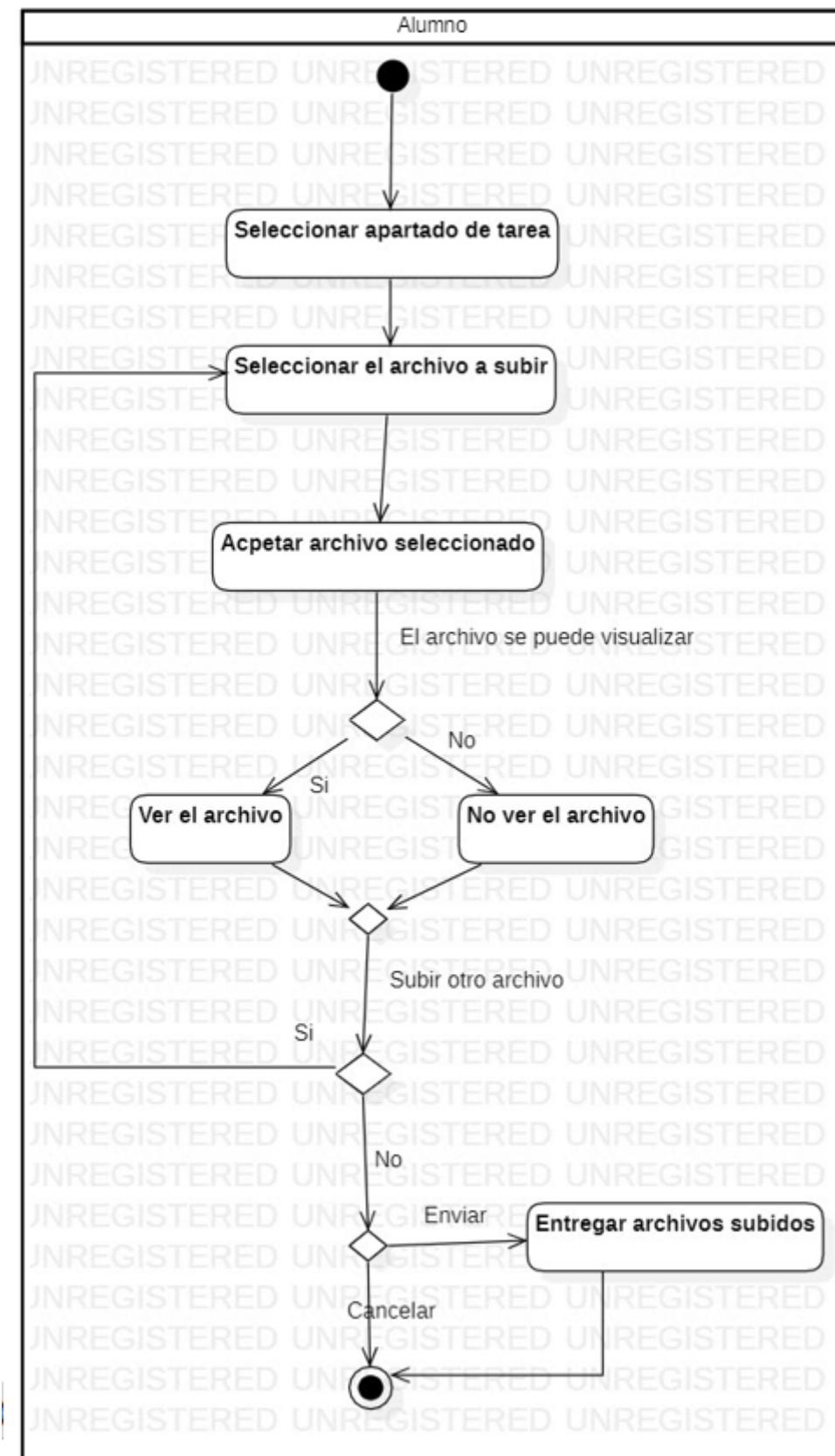
15. Crear recordatorios



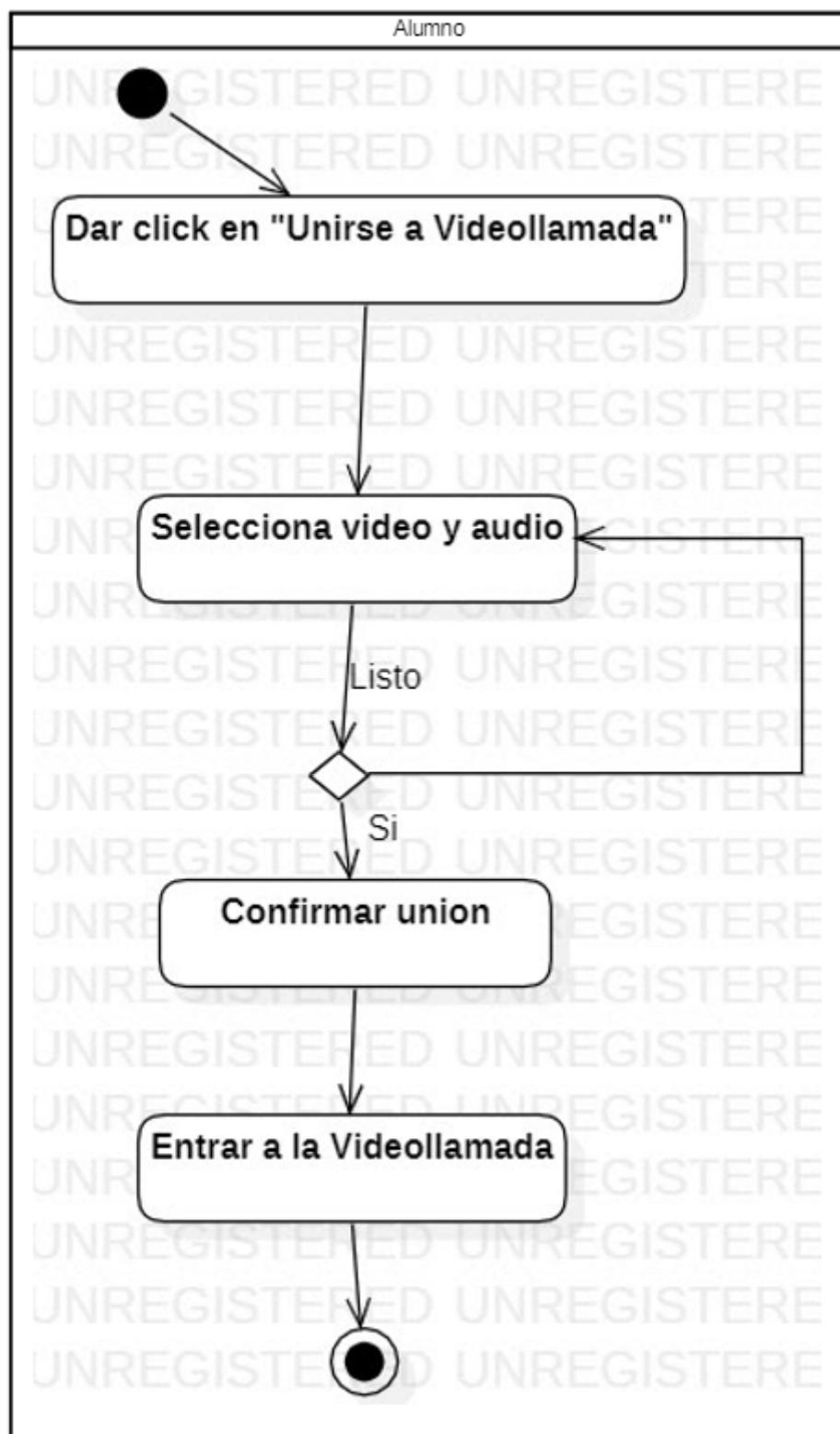
16. Enviar mensajes



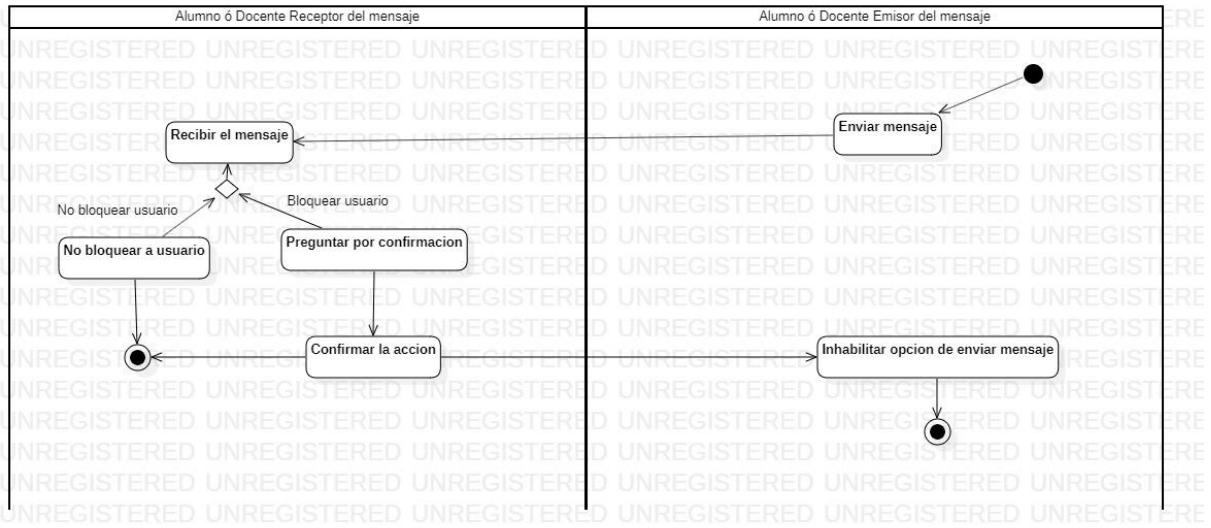
17. Previsualización de archivos



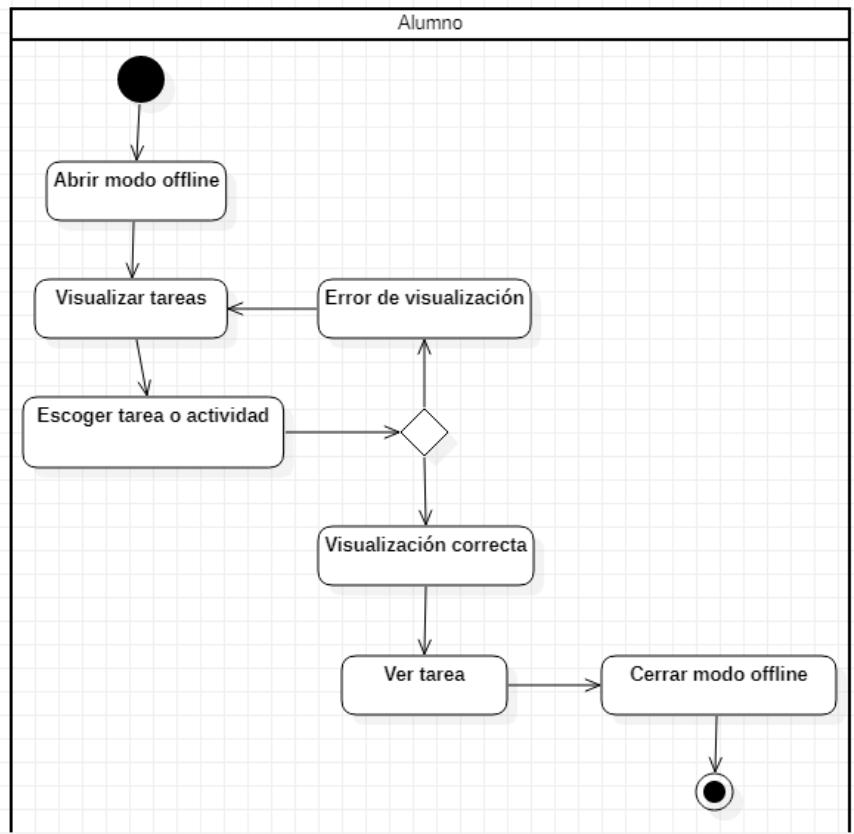
18. Acceder a videollamada activa



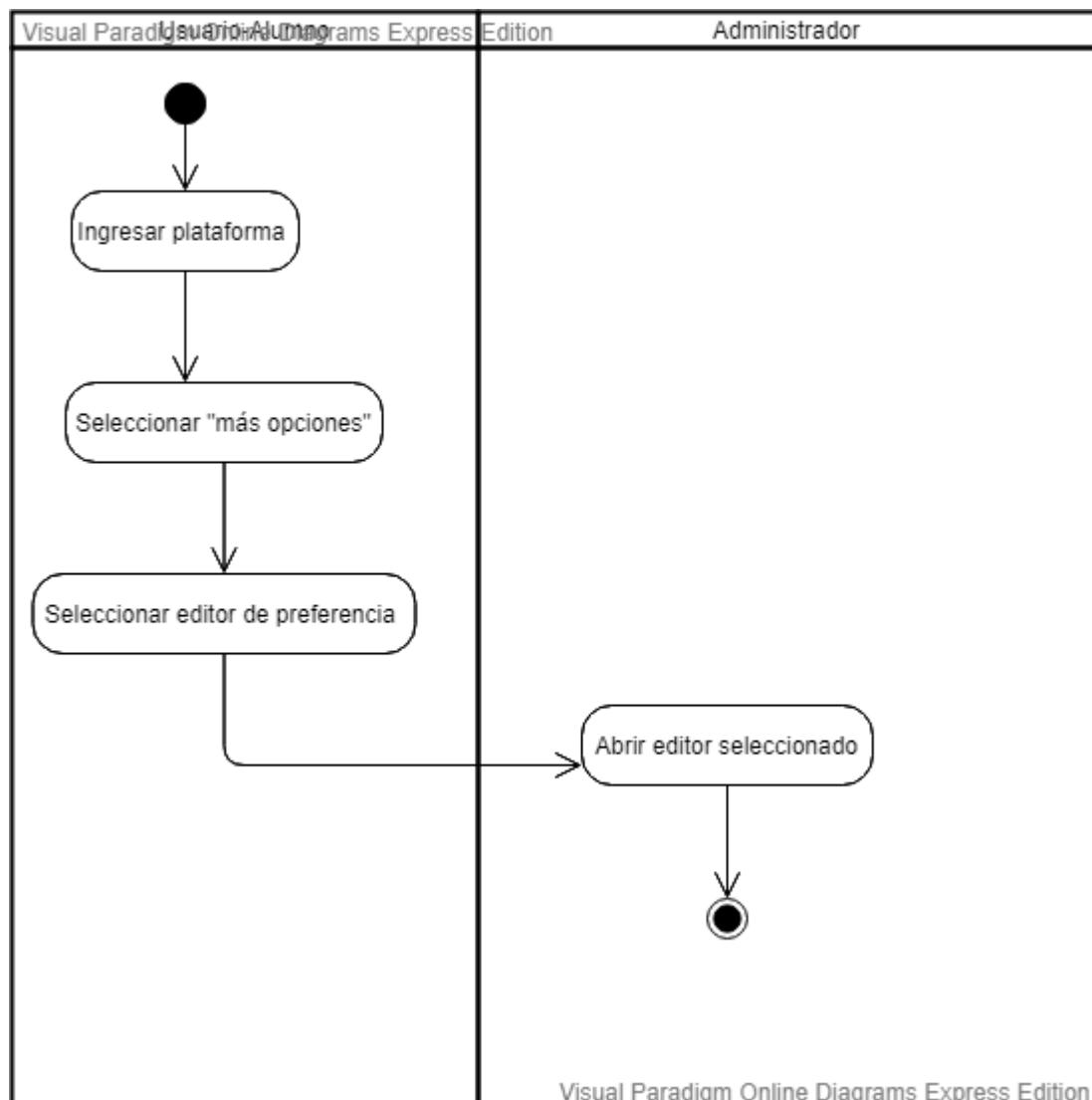
19. Bloquear mensajes



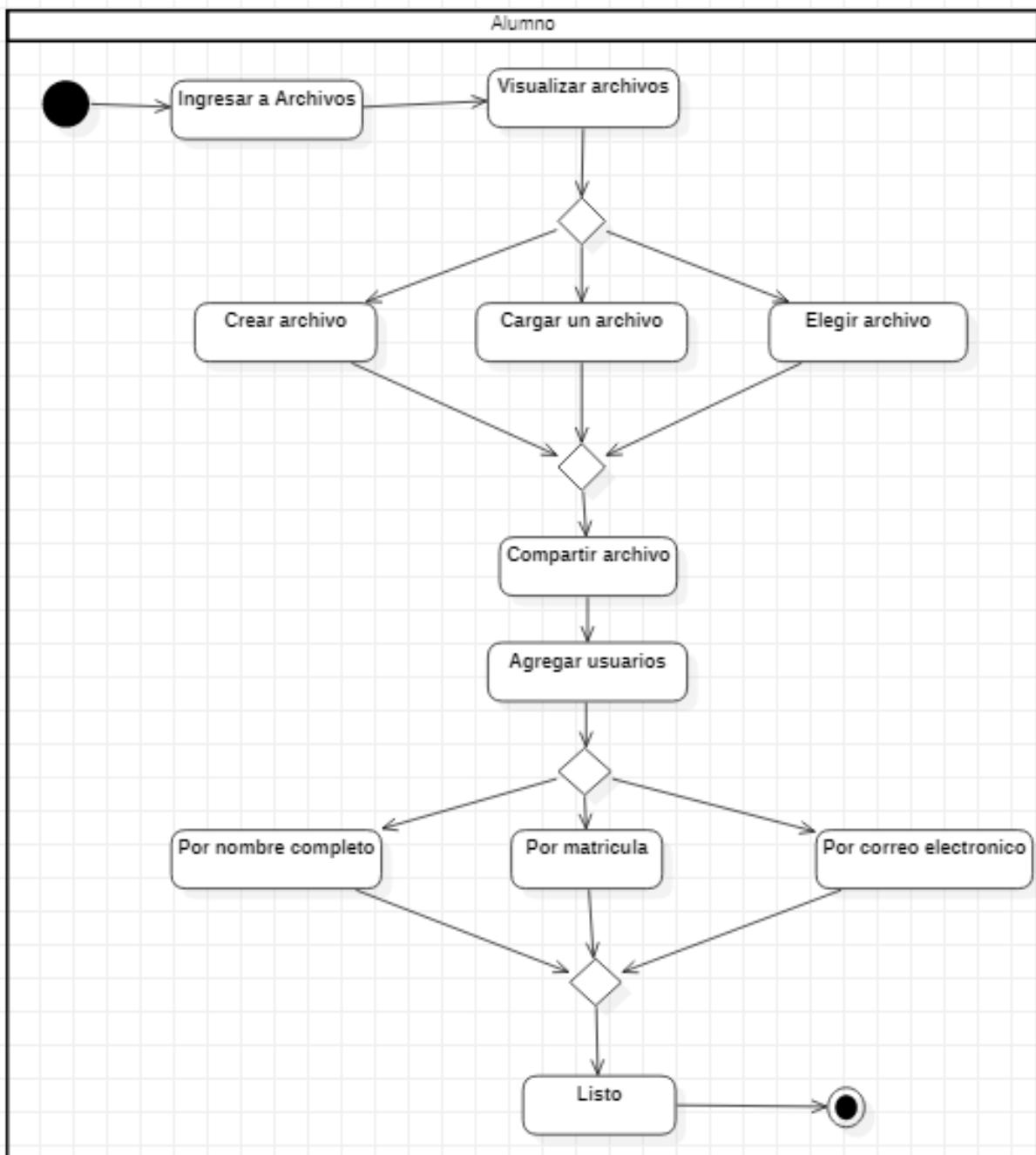
20. Uso offline para visualizar tareas



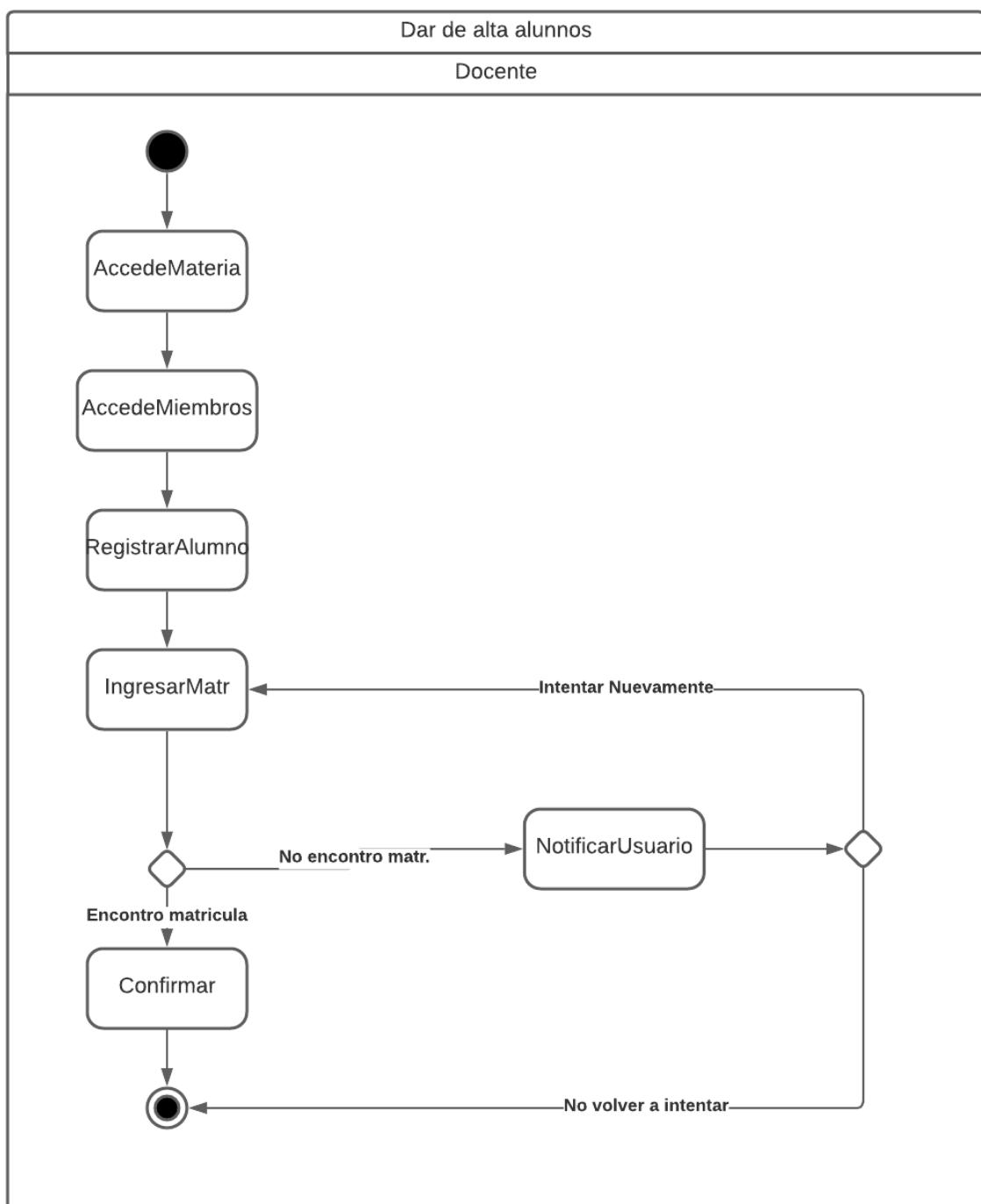
21. Acceder a editores externos



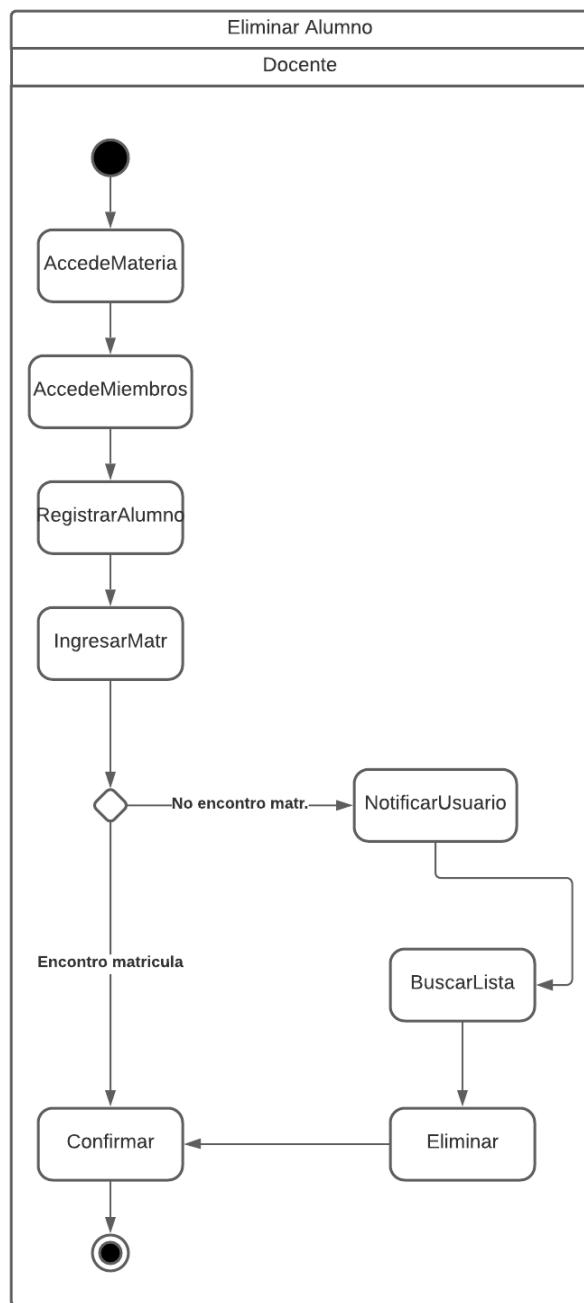
22. Trabajar documentos de manera colaborativa



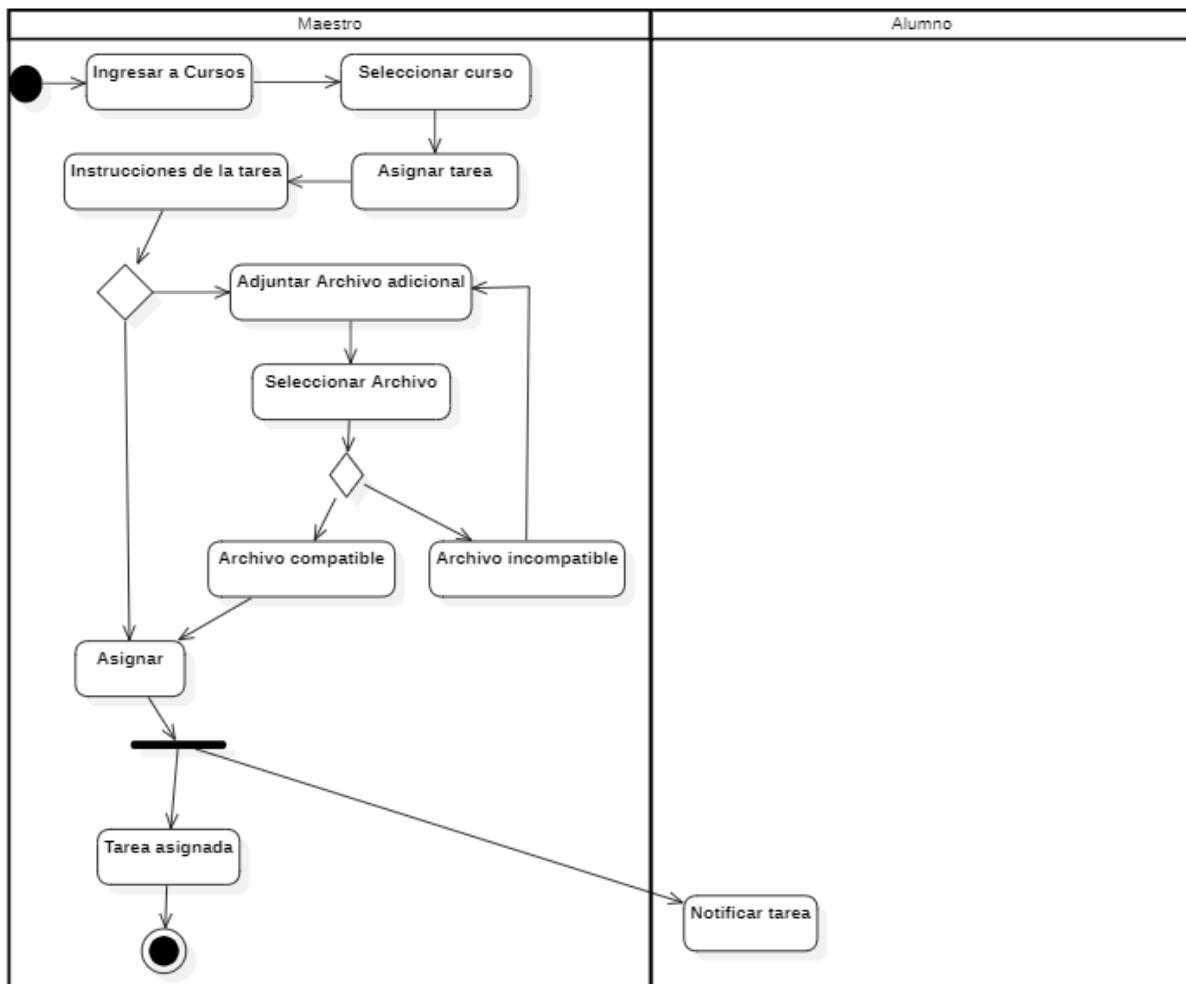
23. Dar de alta alumno del curso



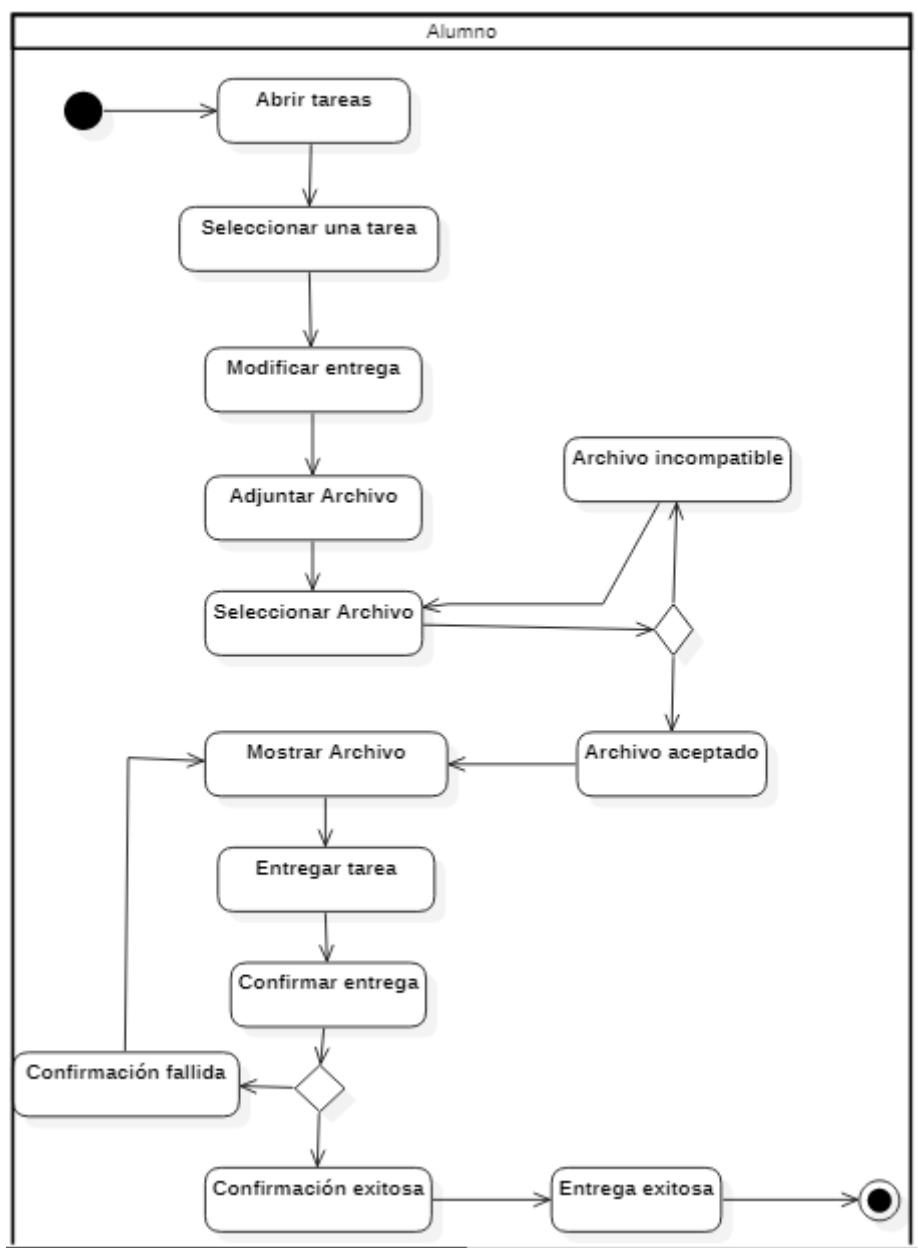
24. Dar de baja alumno de un curso



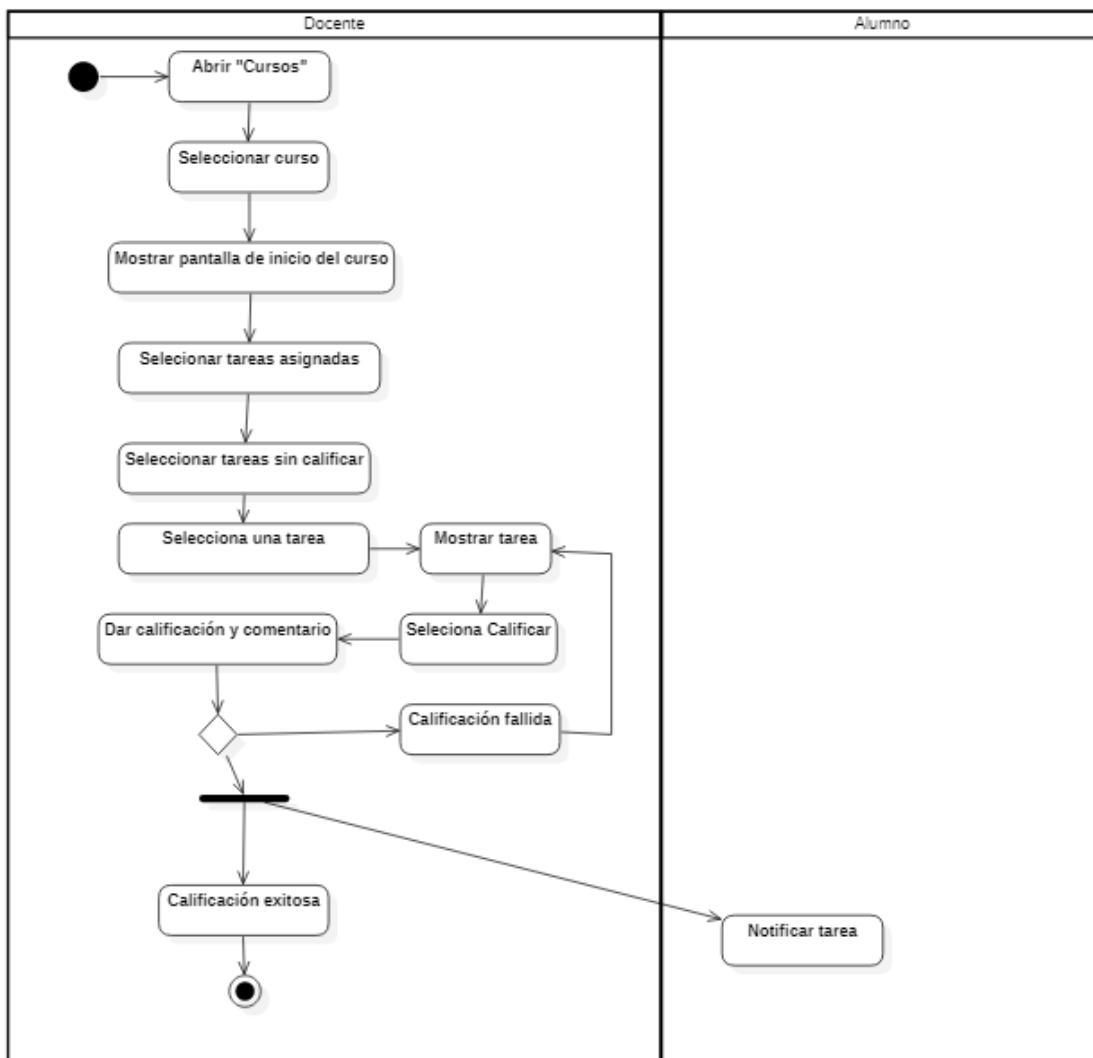
25. Asignar tareas



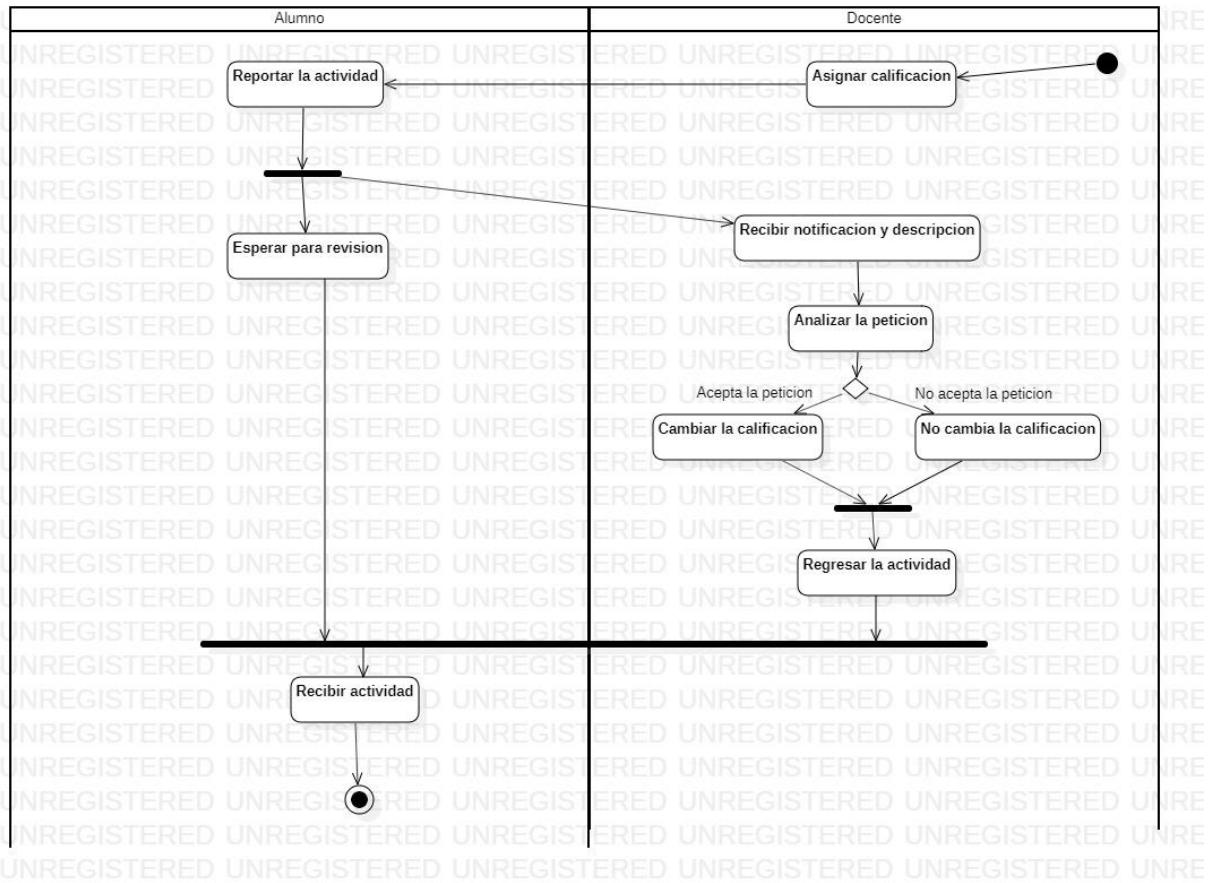
26. Modificar entrega de una tarea



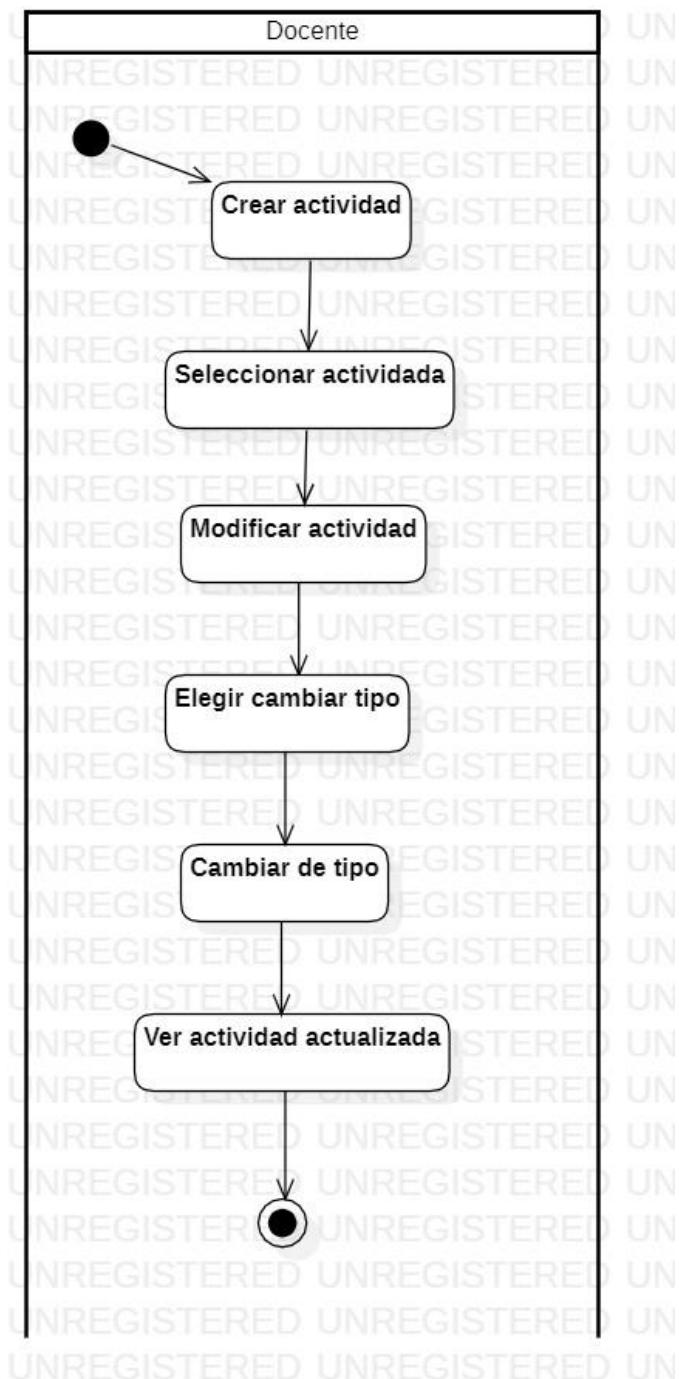
27. Asignar calificación de una tarea



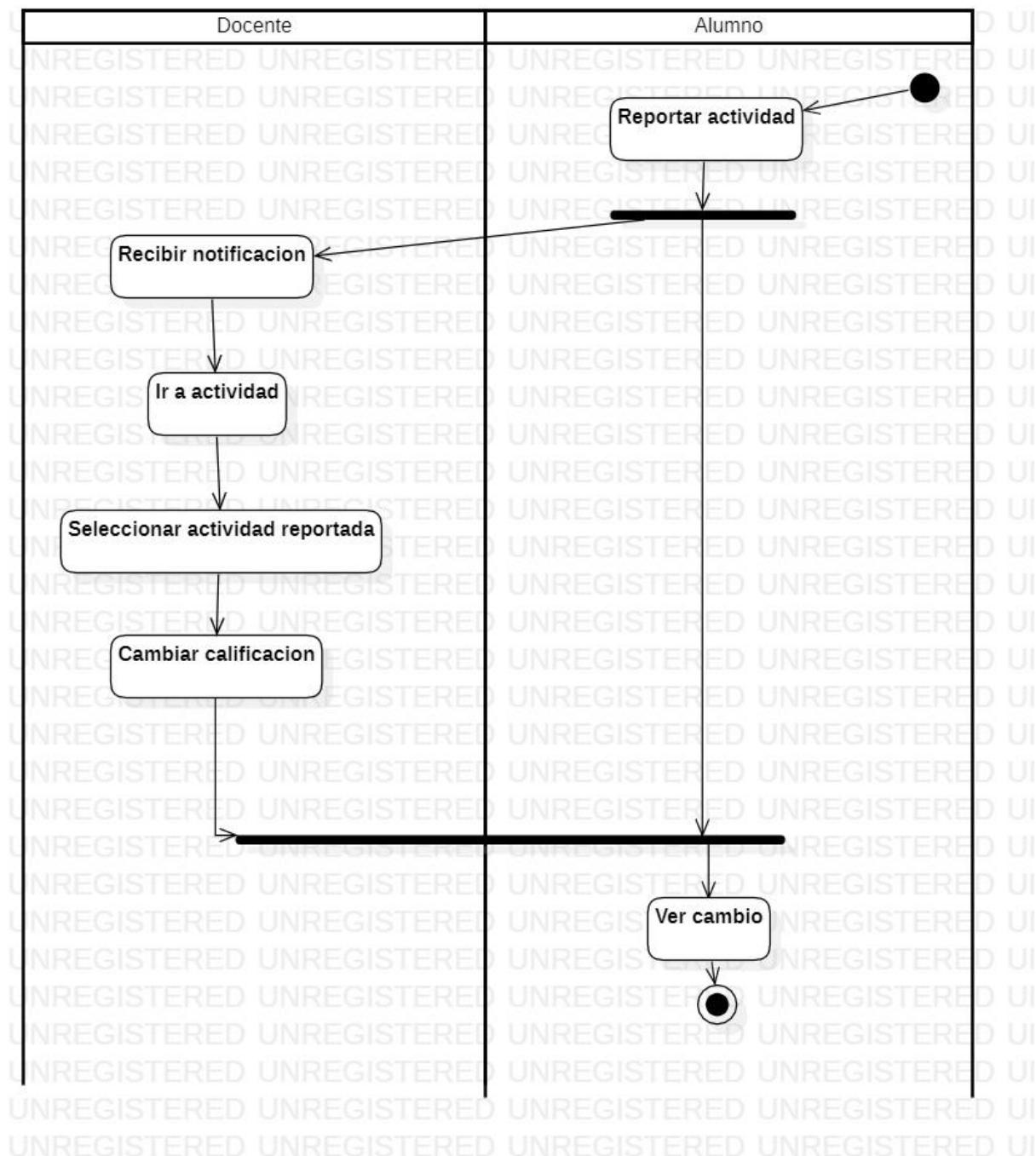
28. Notificar sobre error en la calificación



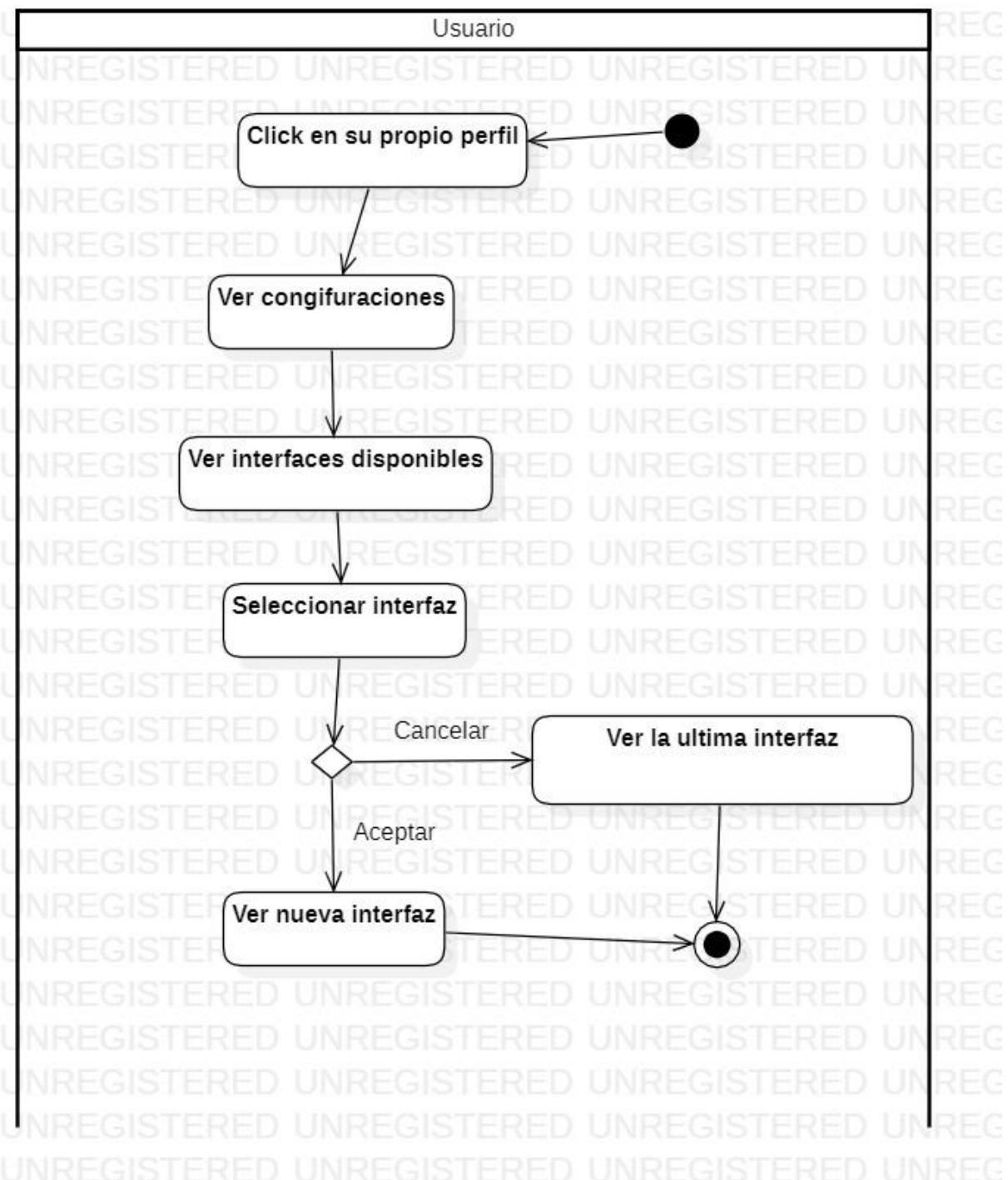
29. Personalización de actividades



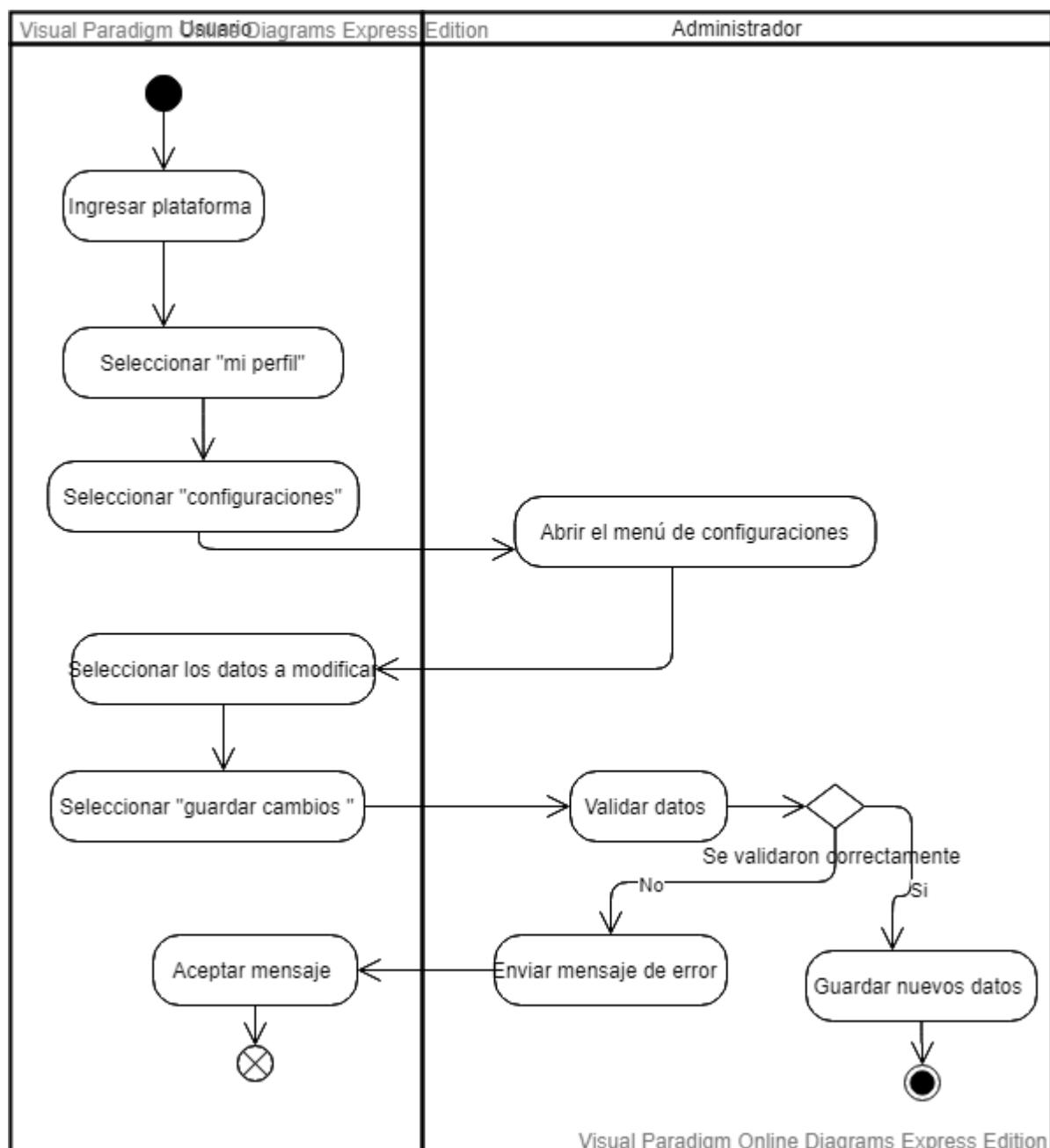
30. Modificar calificación



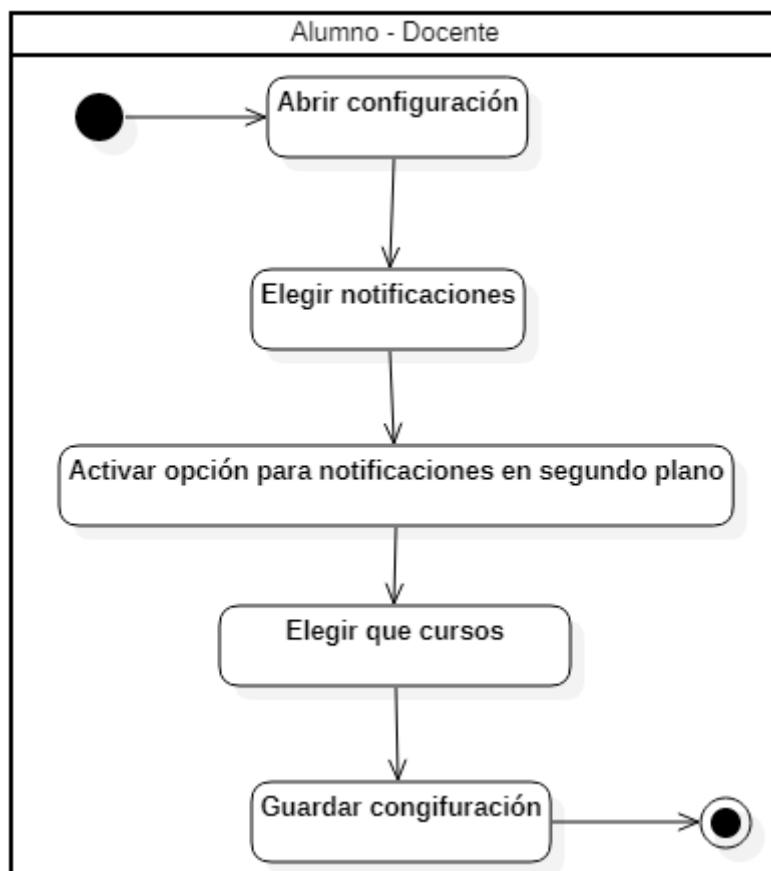
31. Personalizar perfil



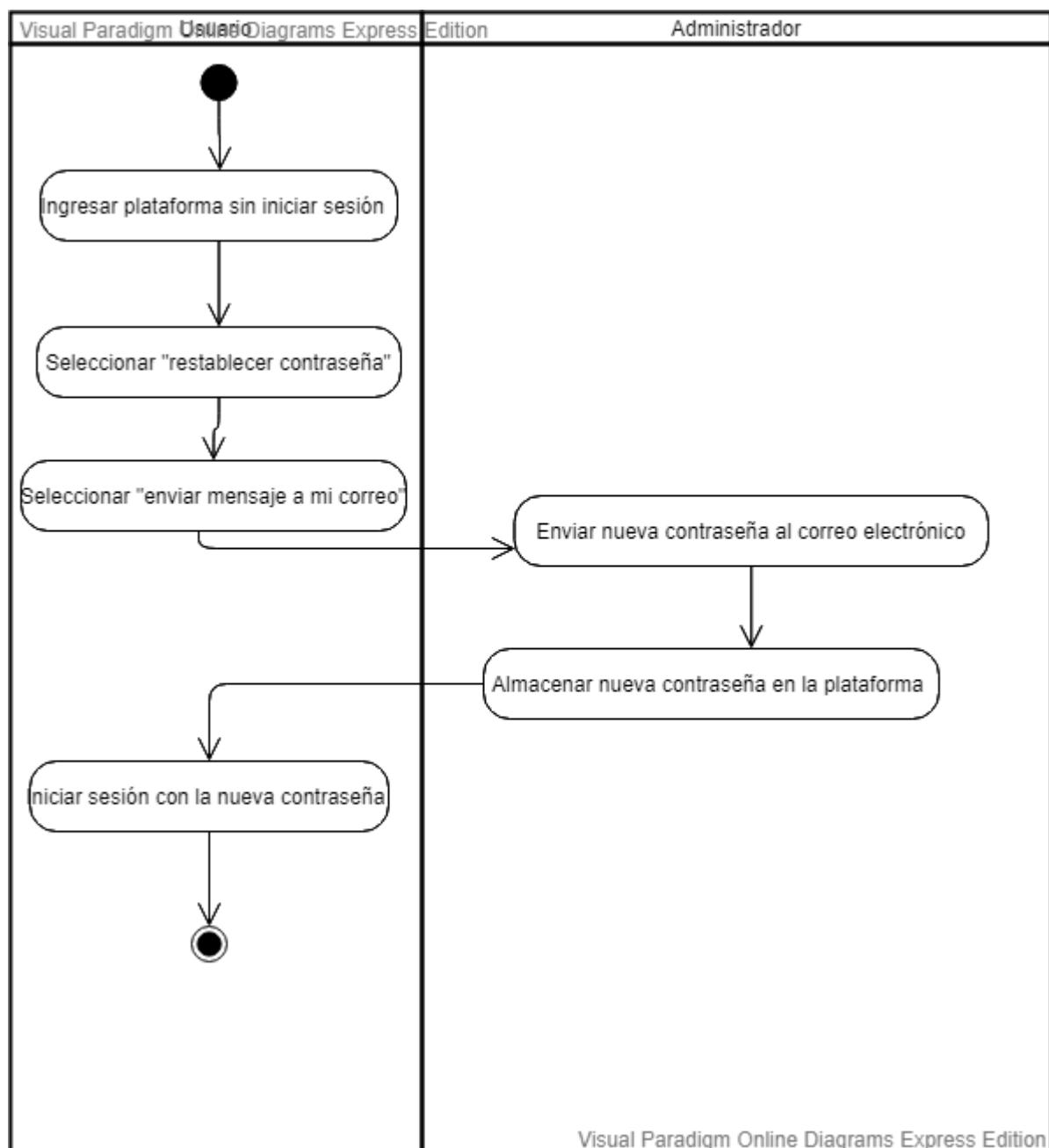
32. Actualizar datos



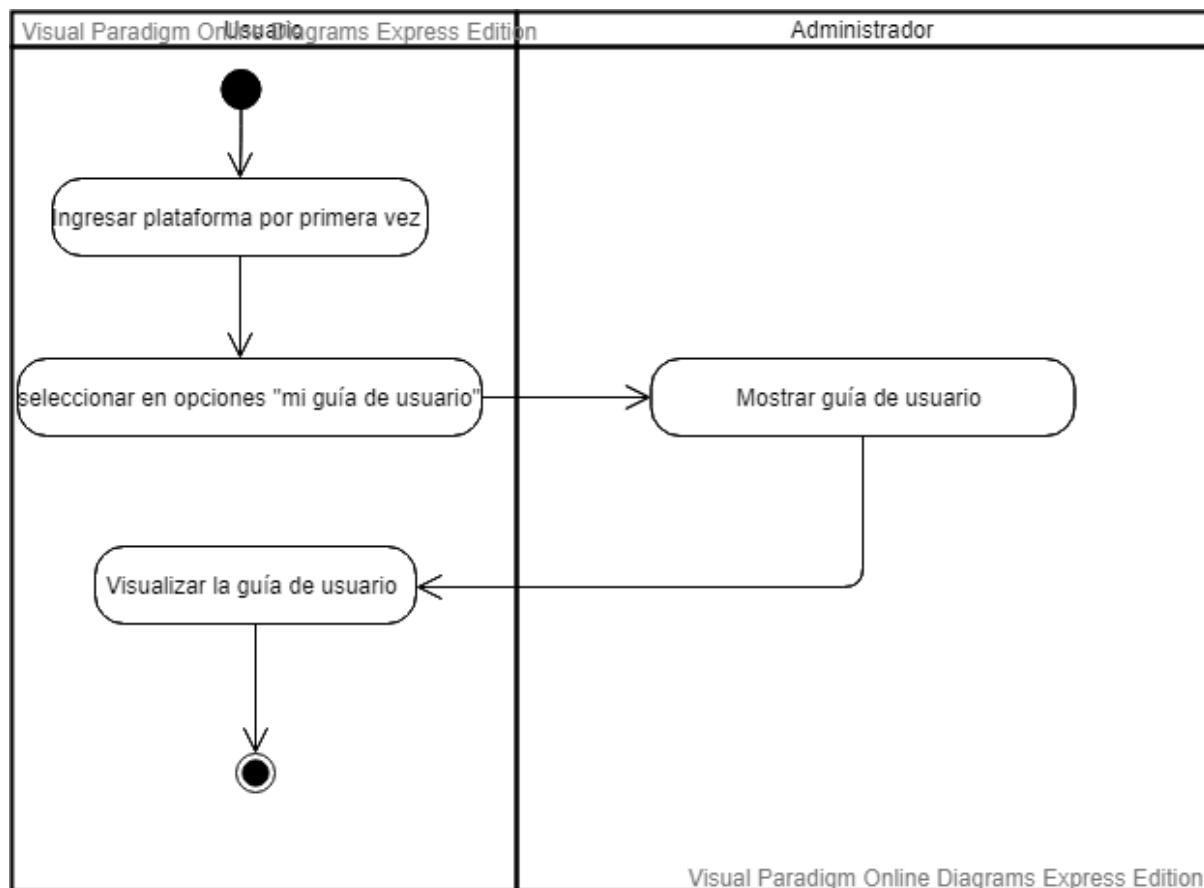
33. Configuración de notificaciones



34. Restablecer contraseña

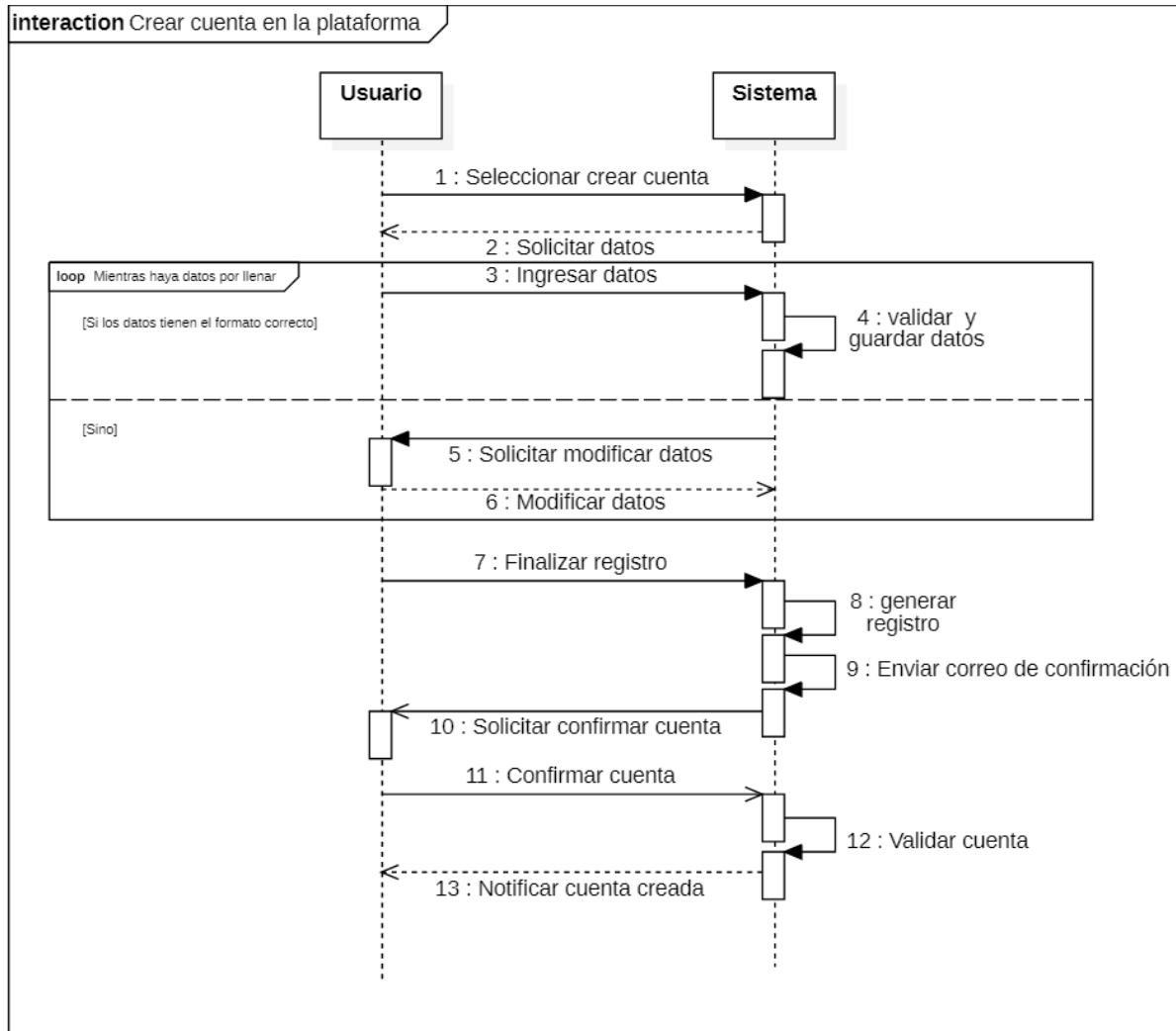


35. Mostrar guía de usuario

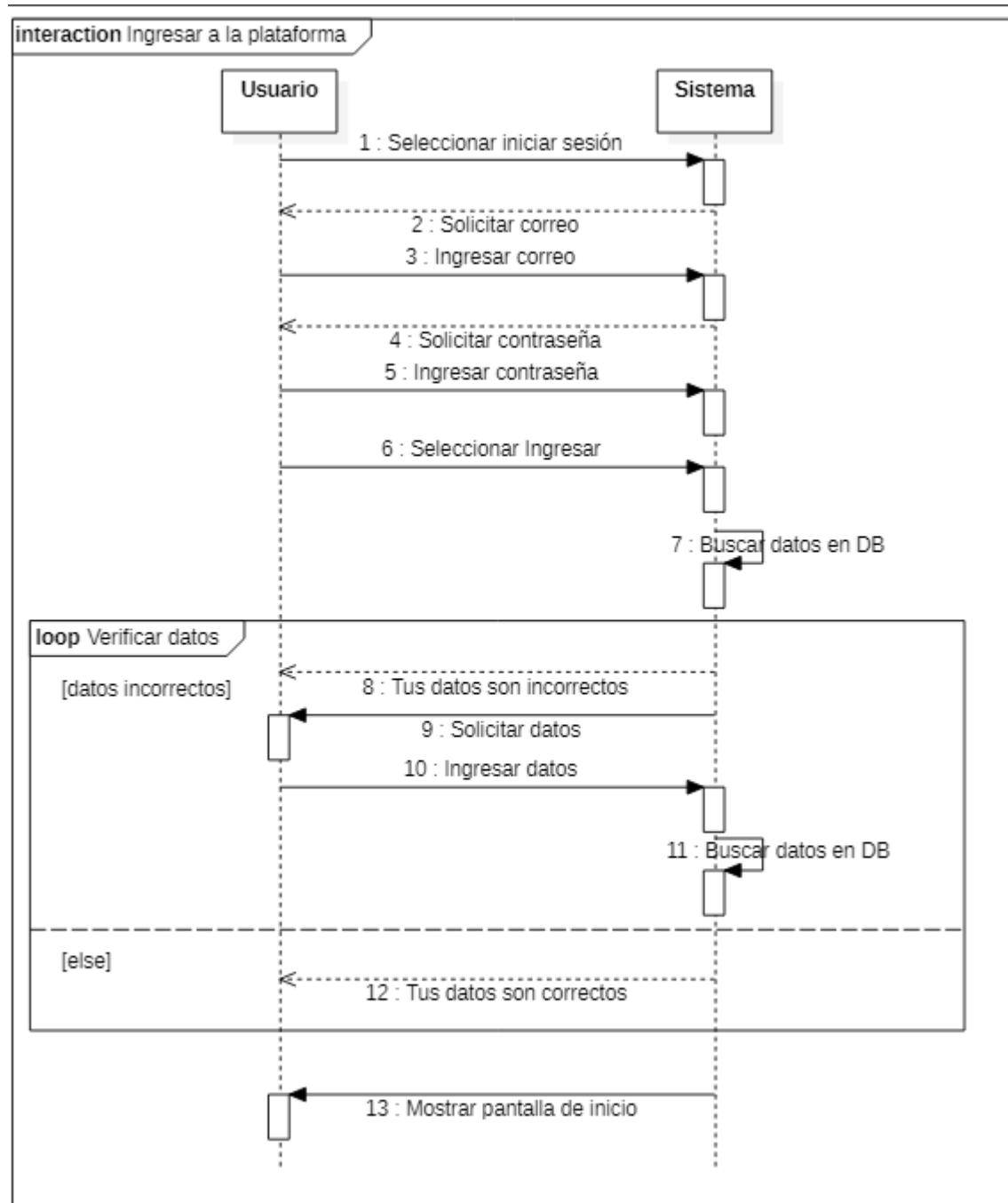


DIAGRAMAS DE SECUENCIA

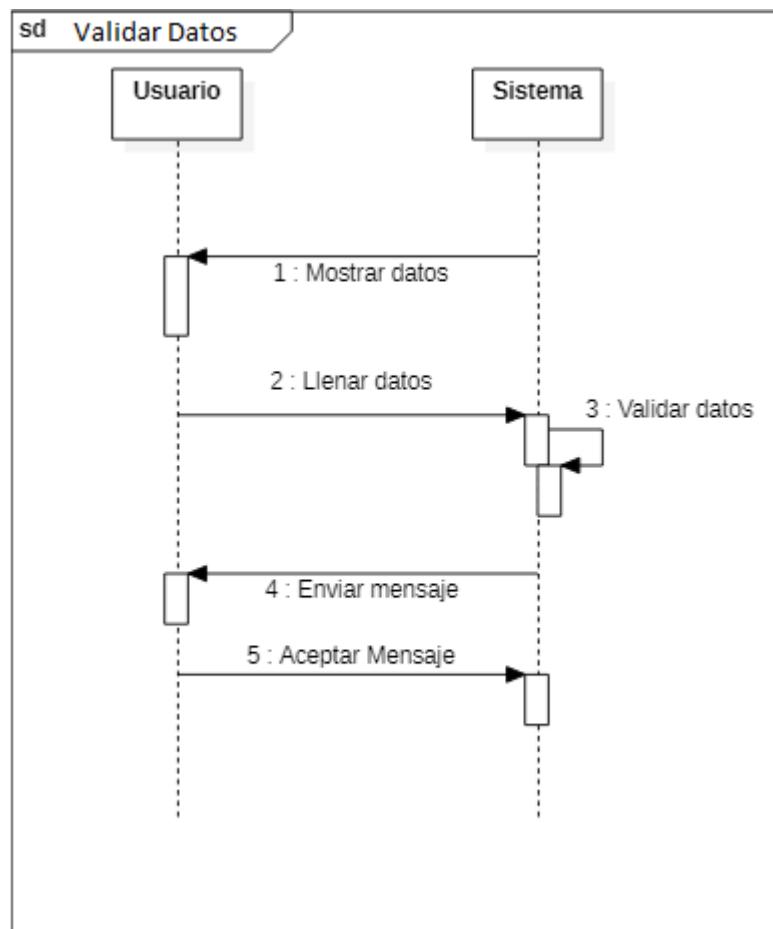
1. Crear cuenta en plataforma



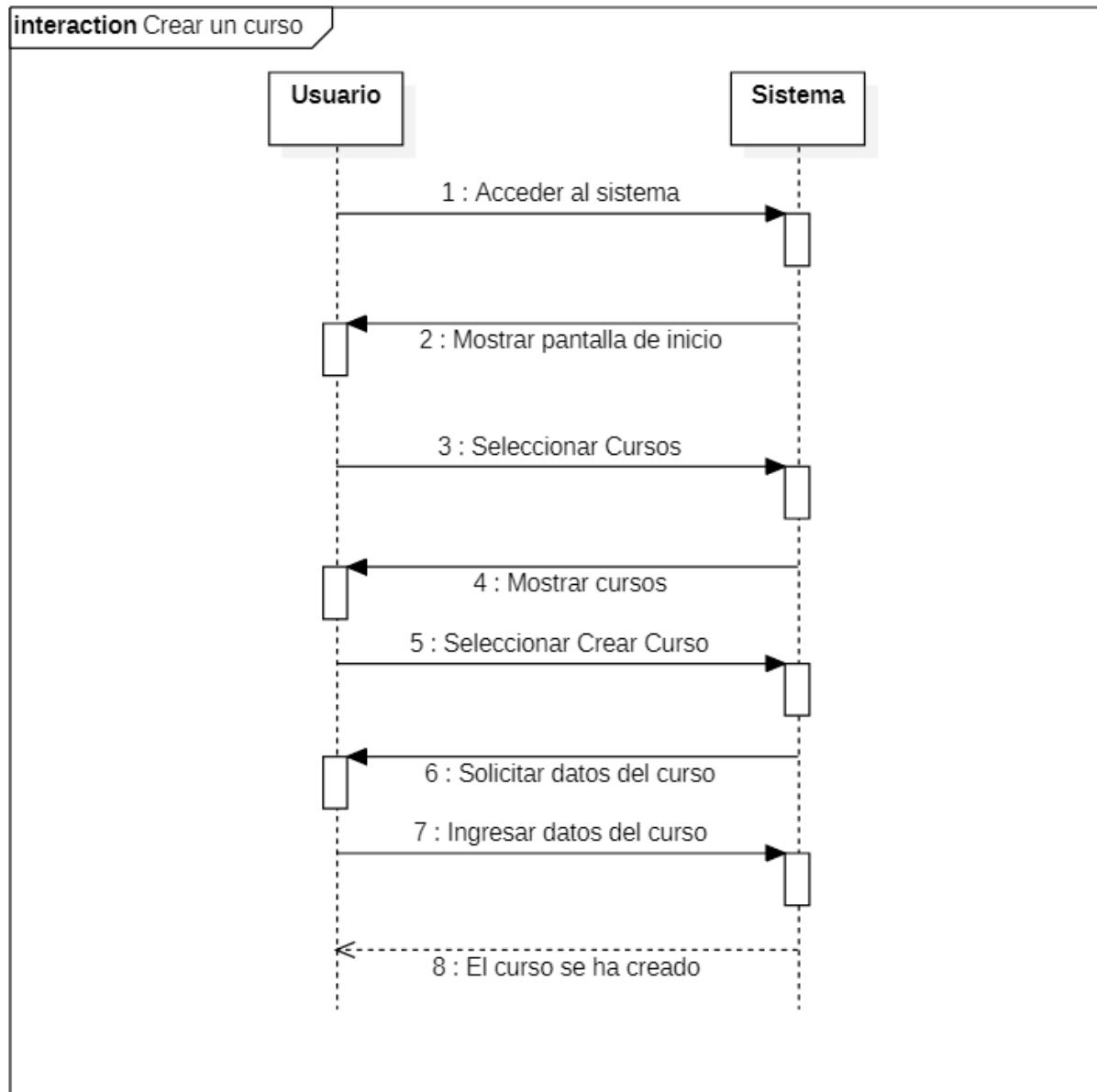
2. Ingresar a la plataforma



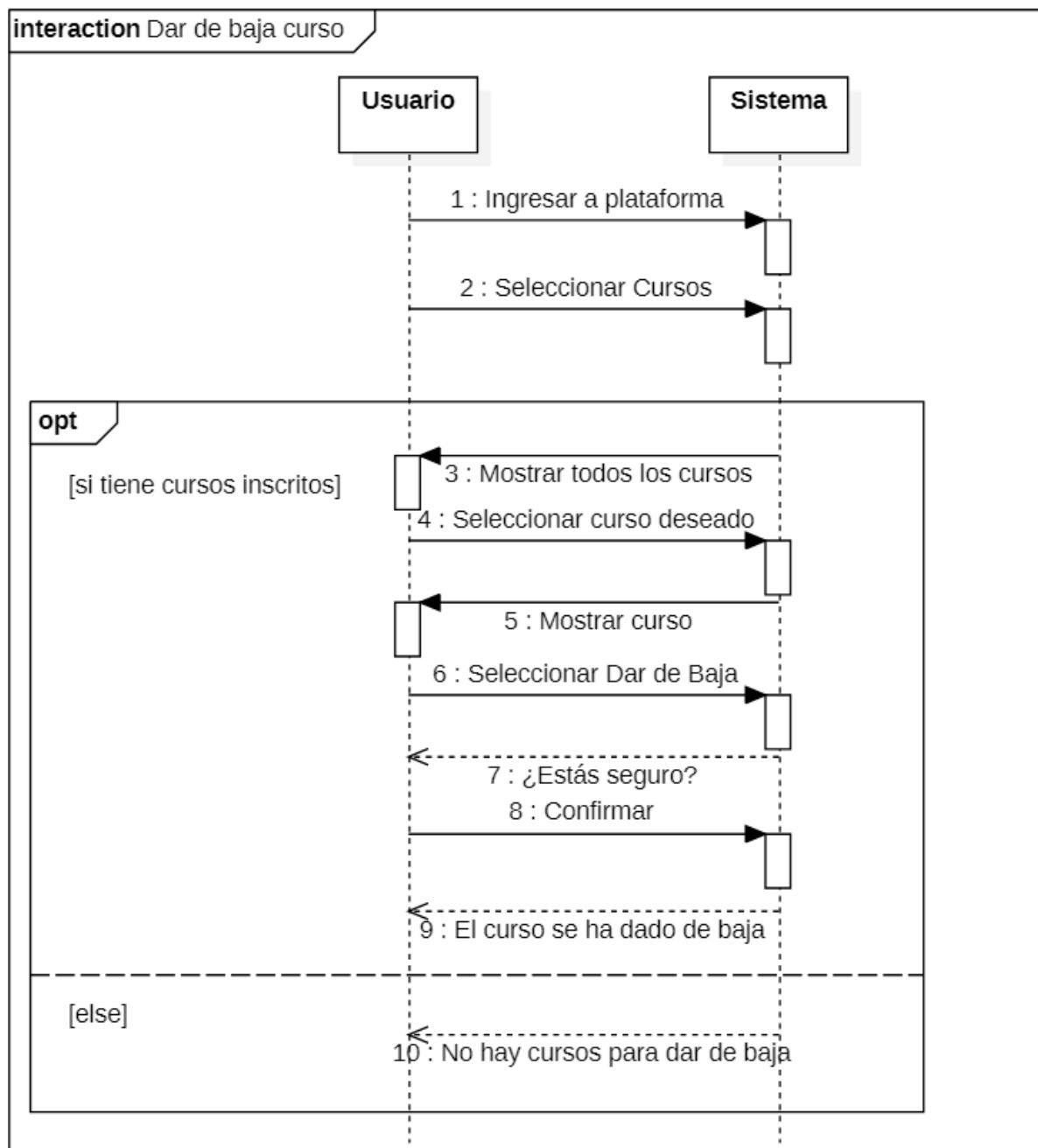
3. Validar Usuario



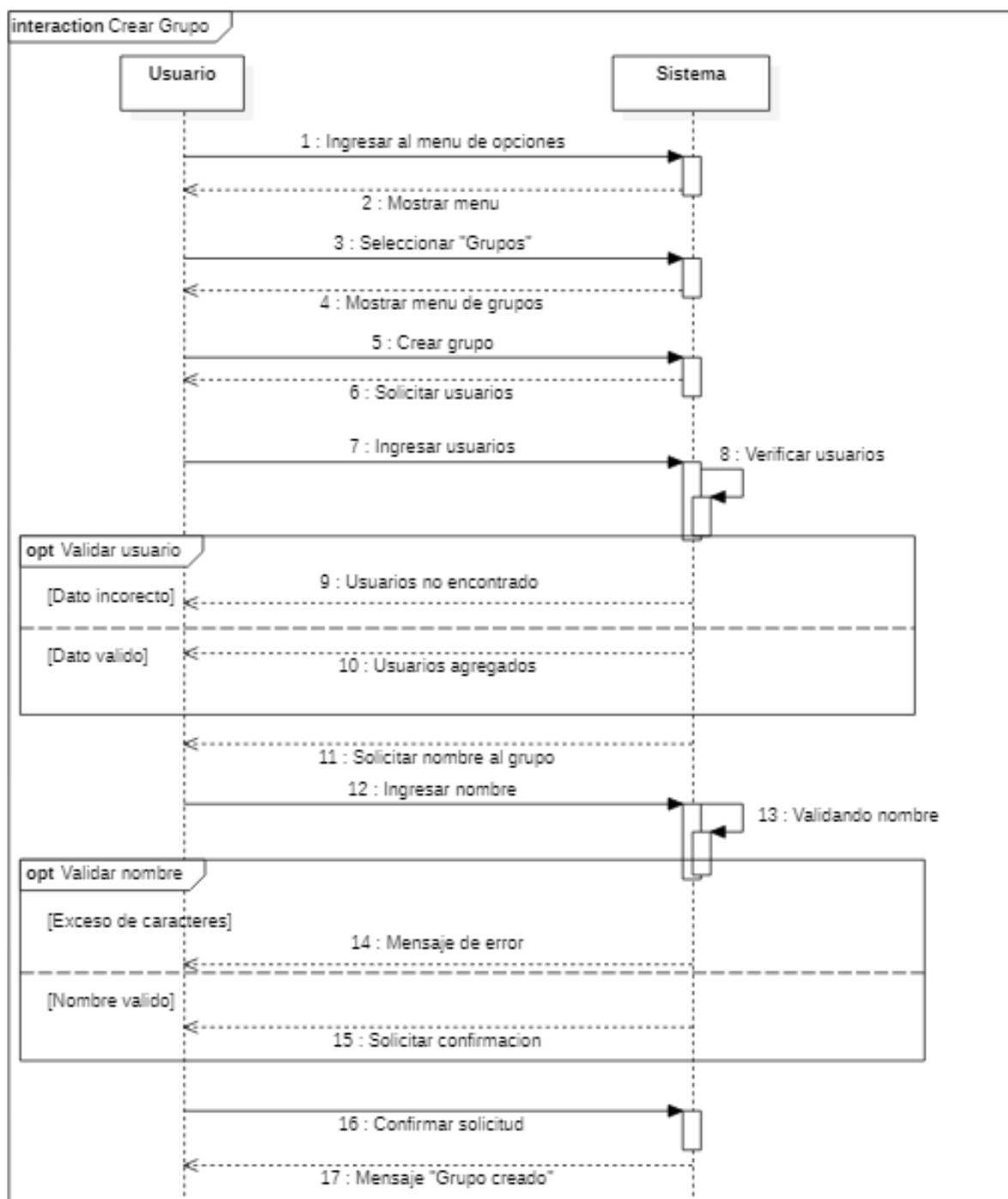
4. Crear un curso



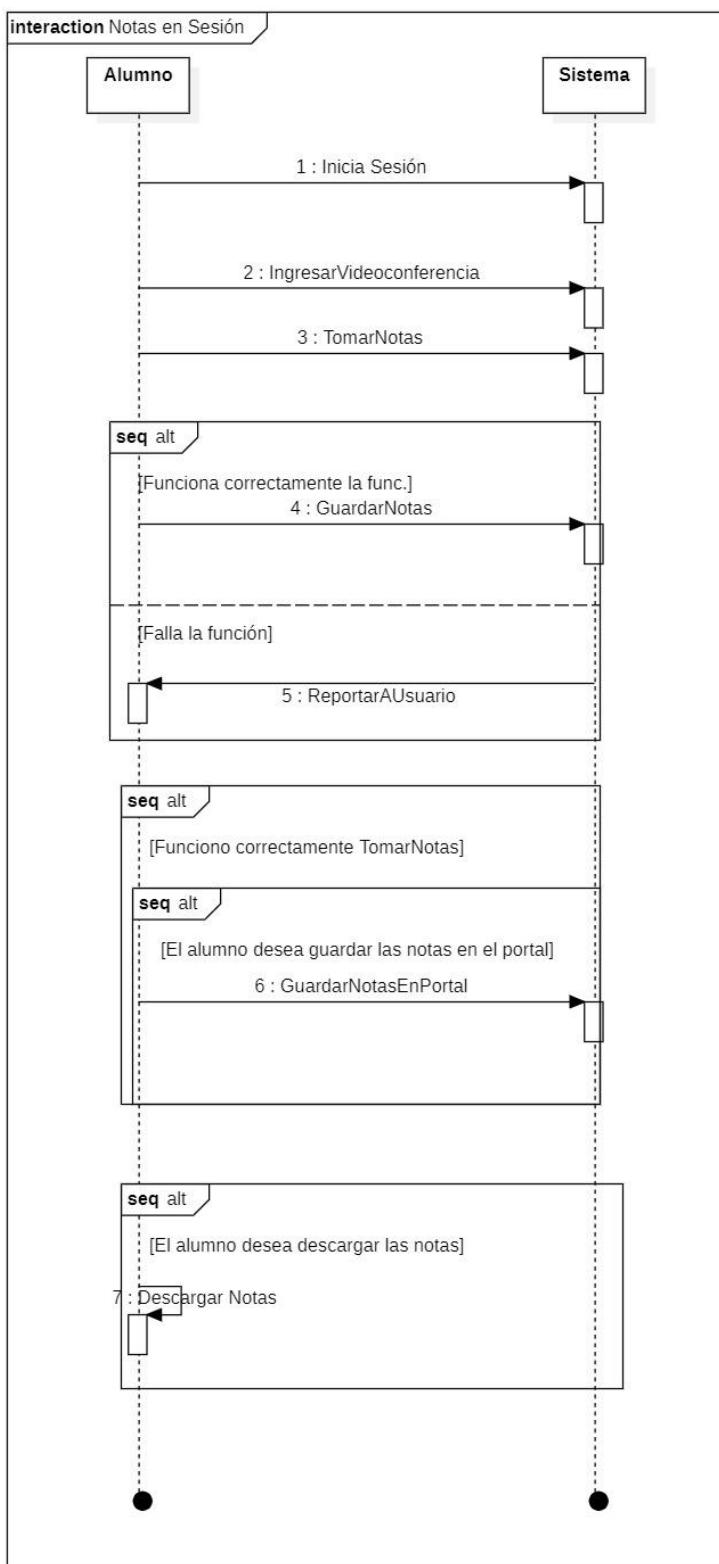
5. Dar de baja un curso



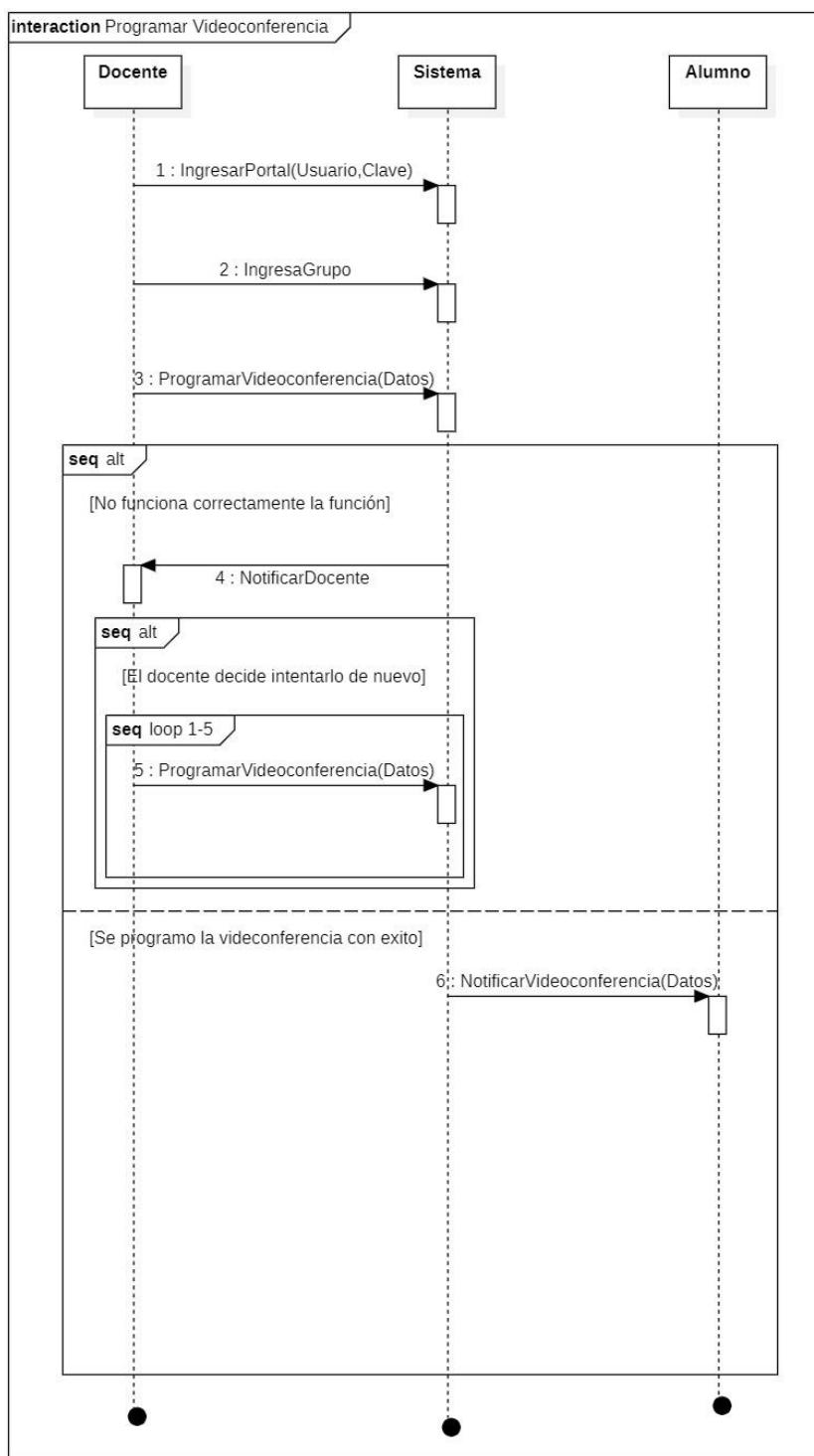
6. Crear grupos entre usuarios



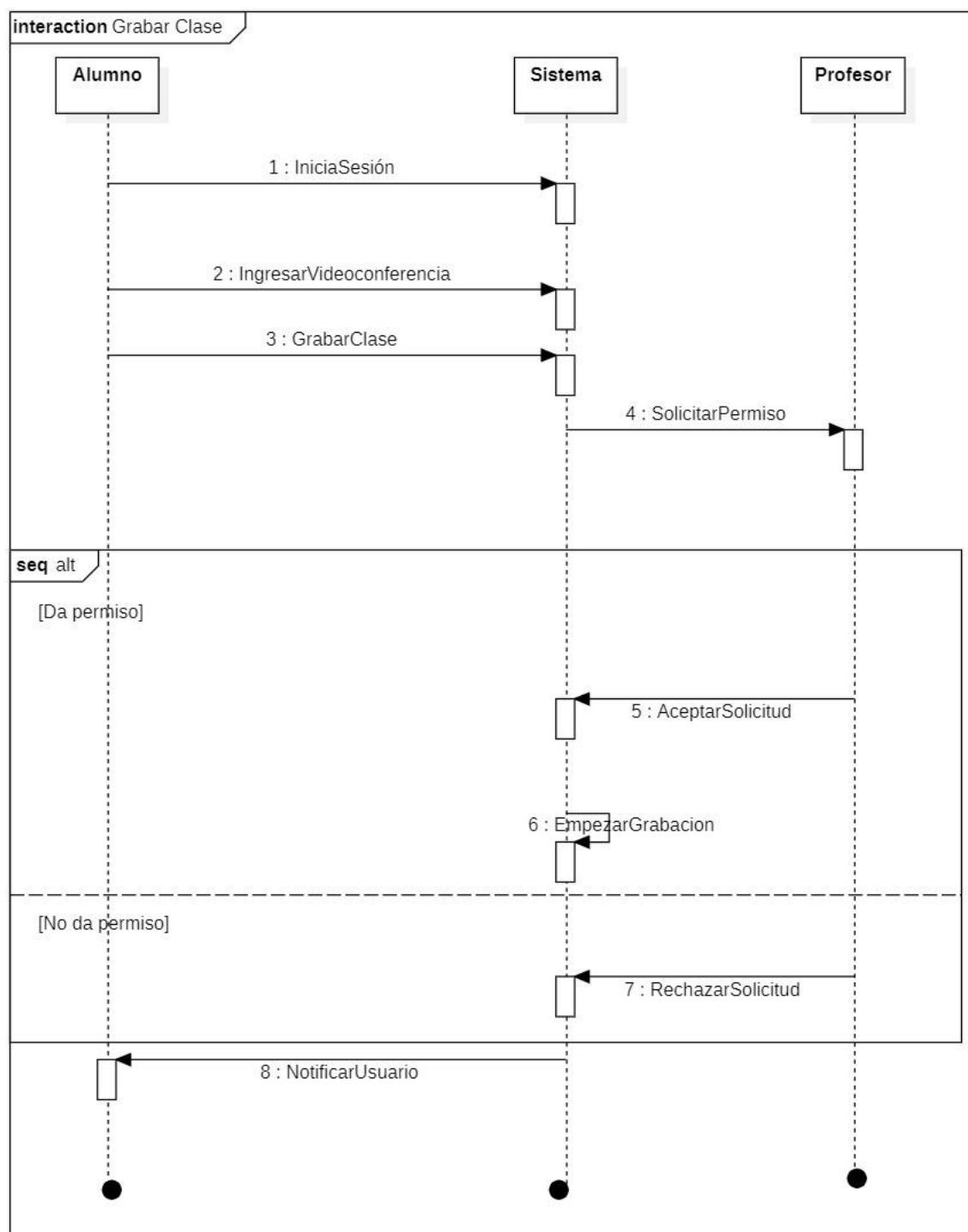
7.Creación de notas en sesión



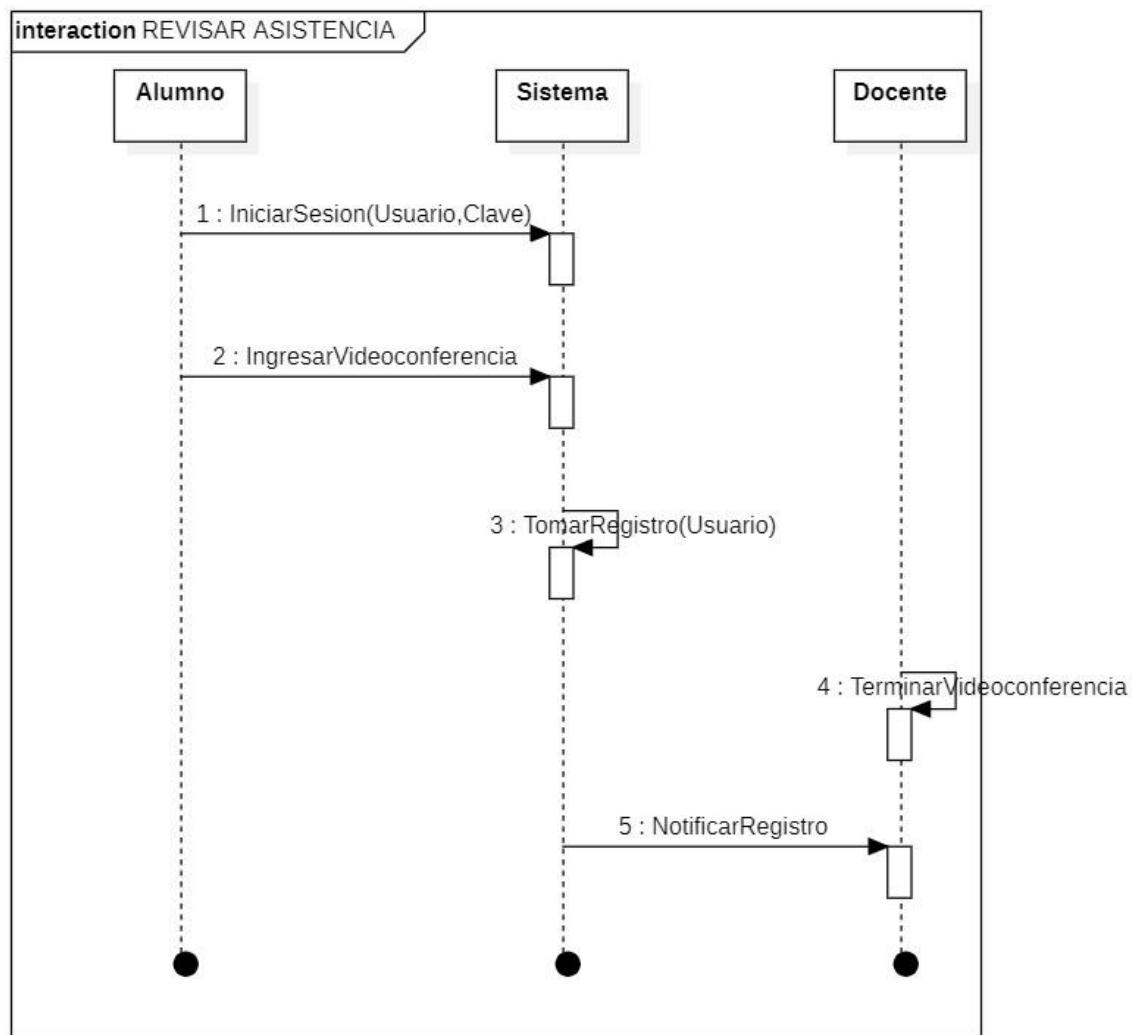
8. Programar Videoconferencia



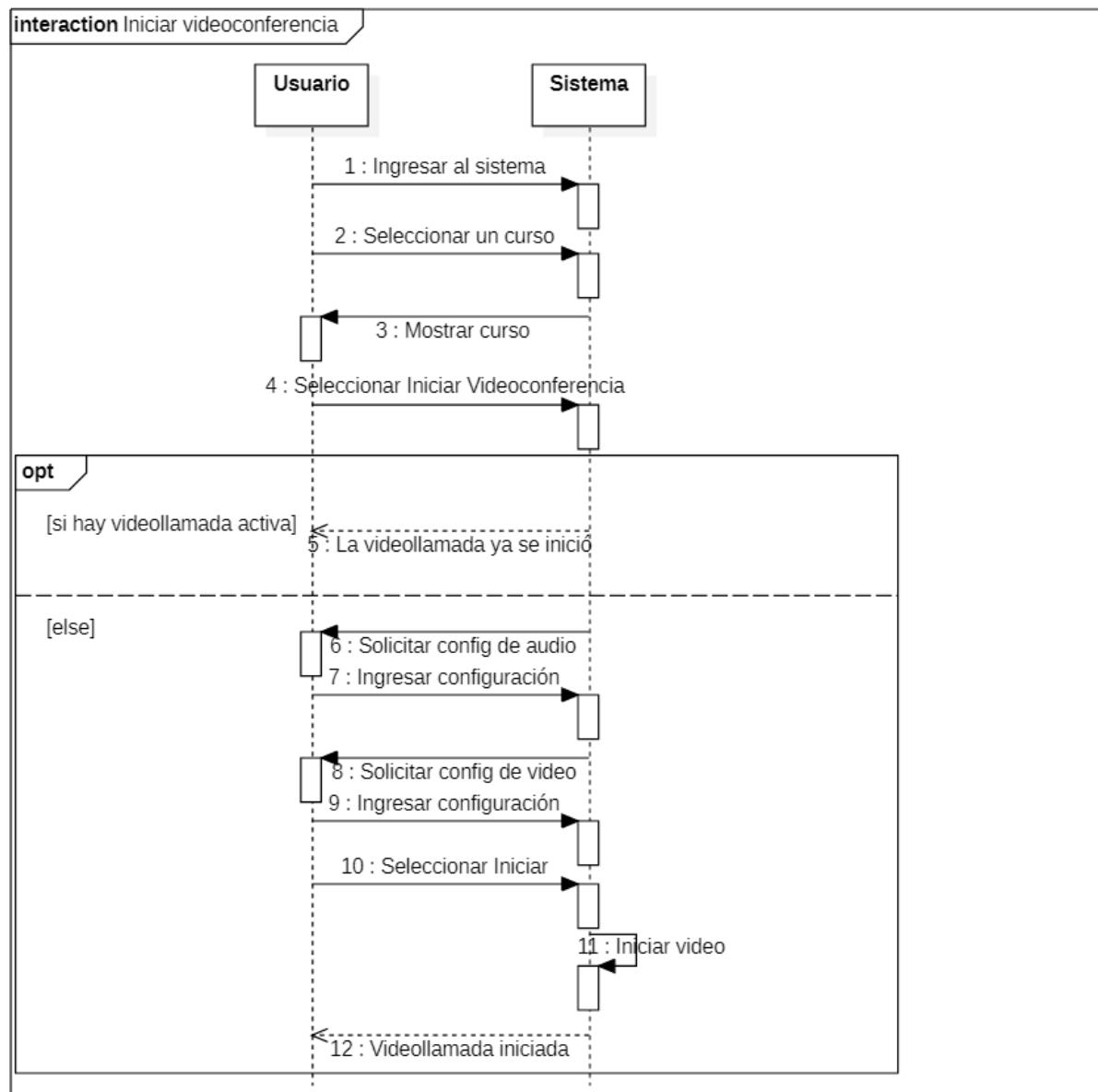
9.Grabar Clase



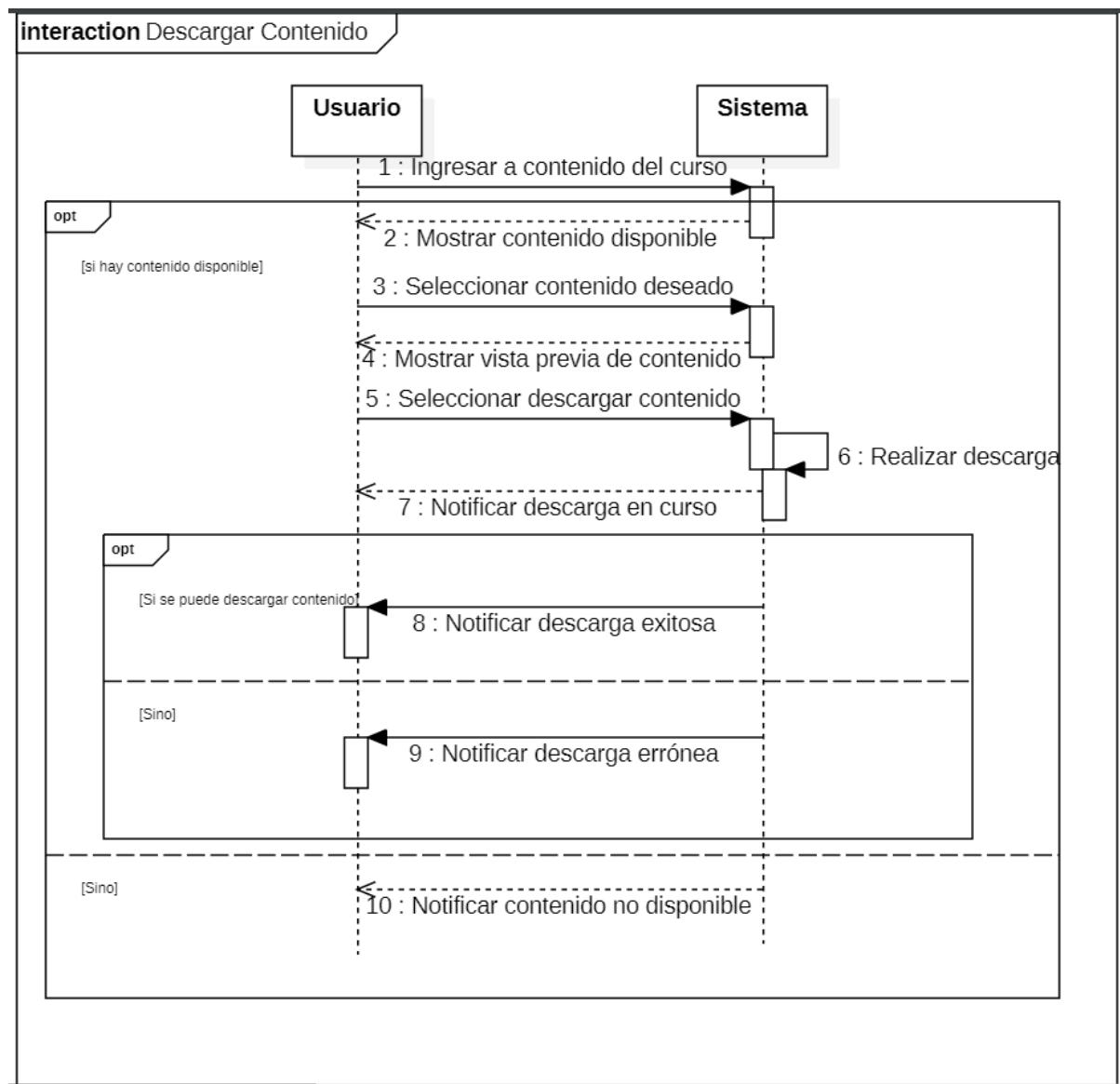
10. Revisar Asistencia



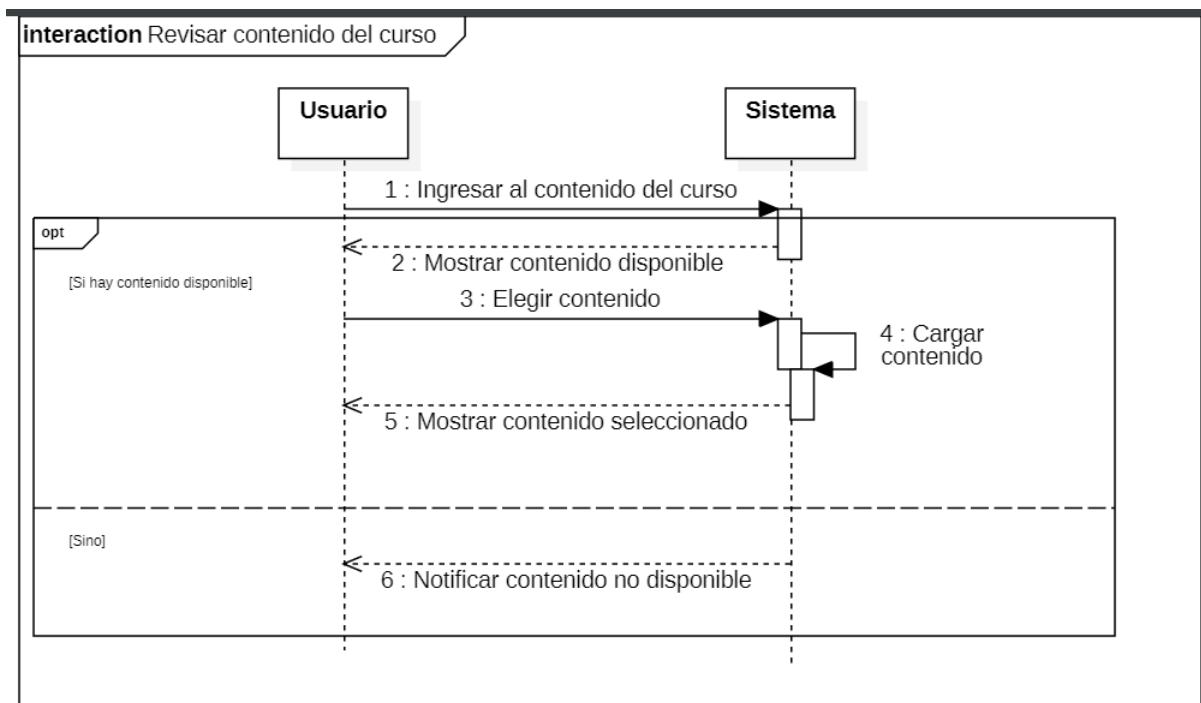
11. Iniciar videoconferencia



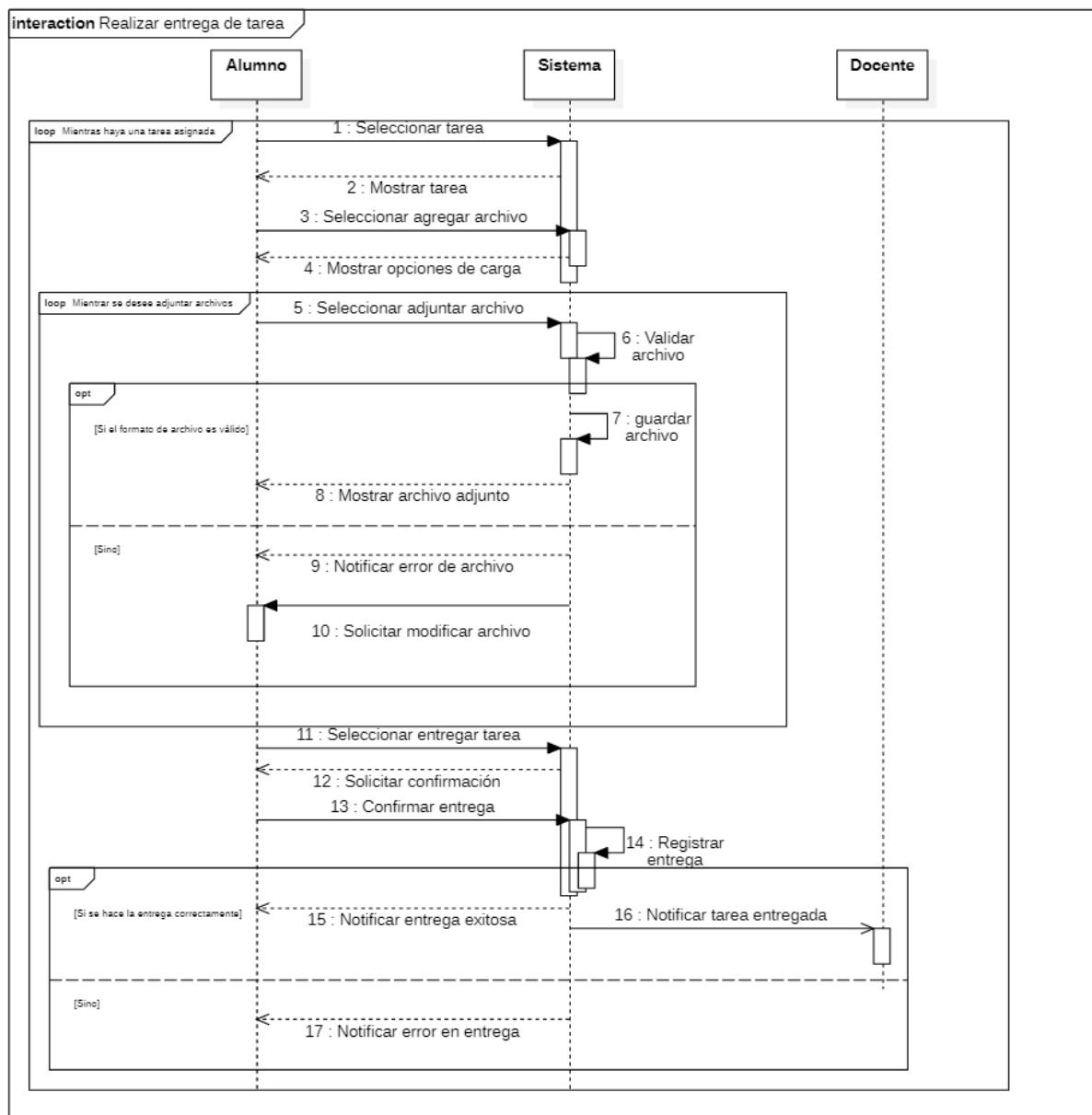
12. Descargar contenido



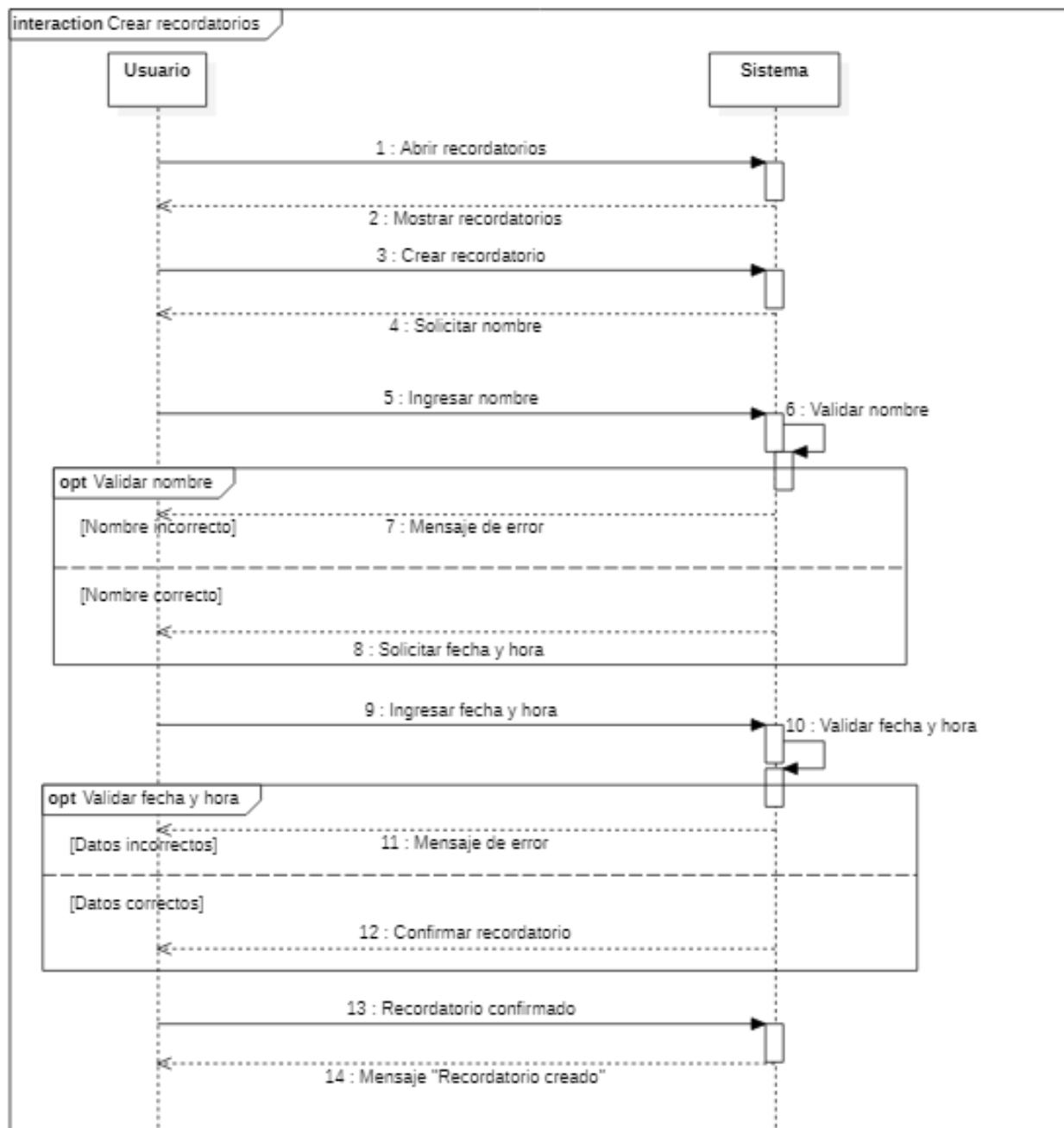
13. Revisar contenido



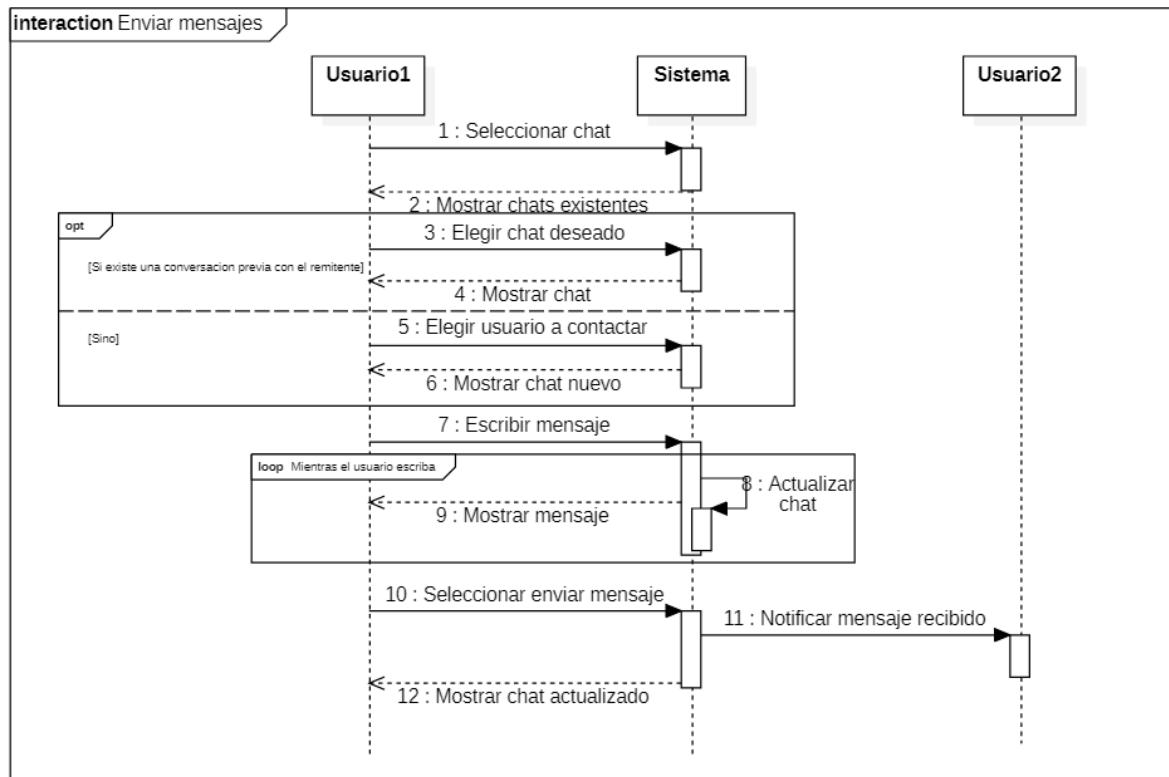
14. Realizar entrega de una tarea



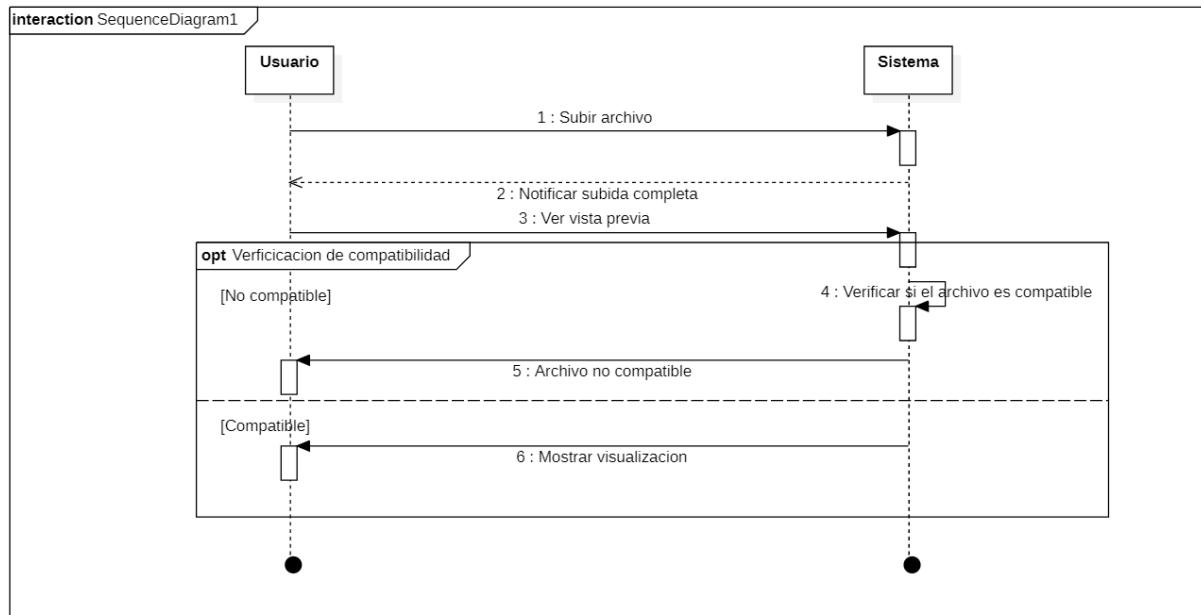
15. Crear recordatorios



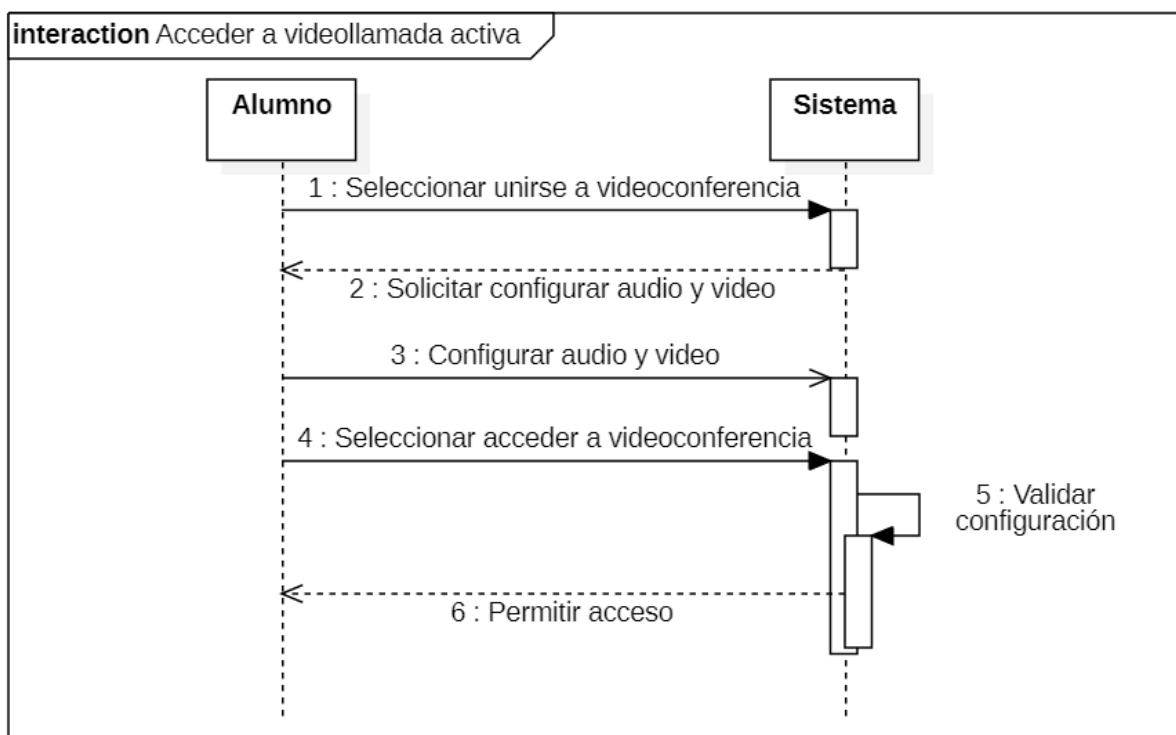
16. Enviar mensajes



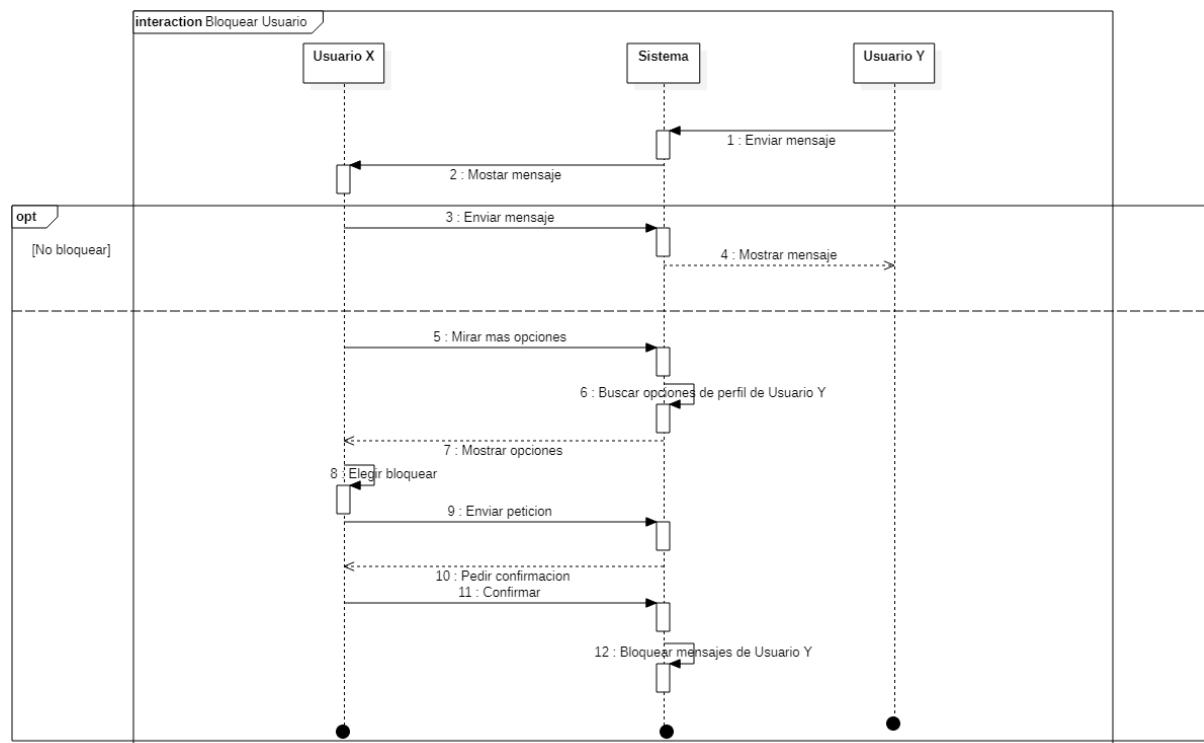
17. Previsualizar archivos



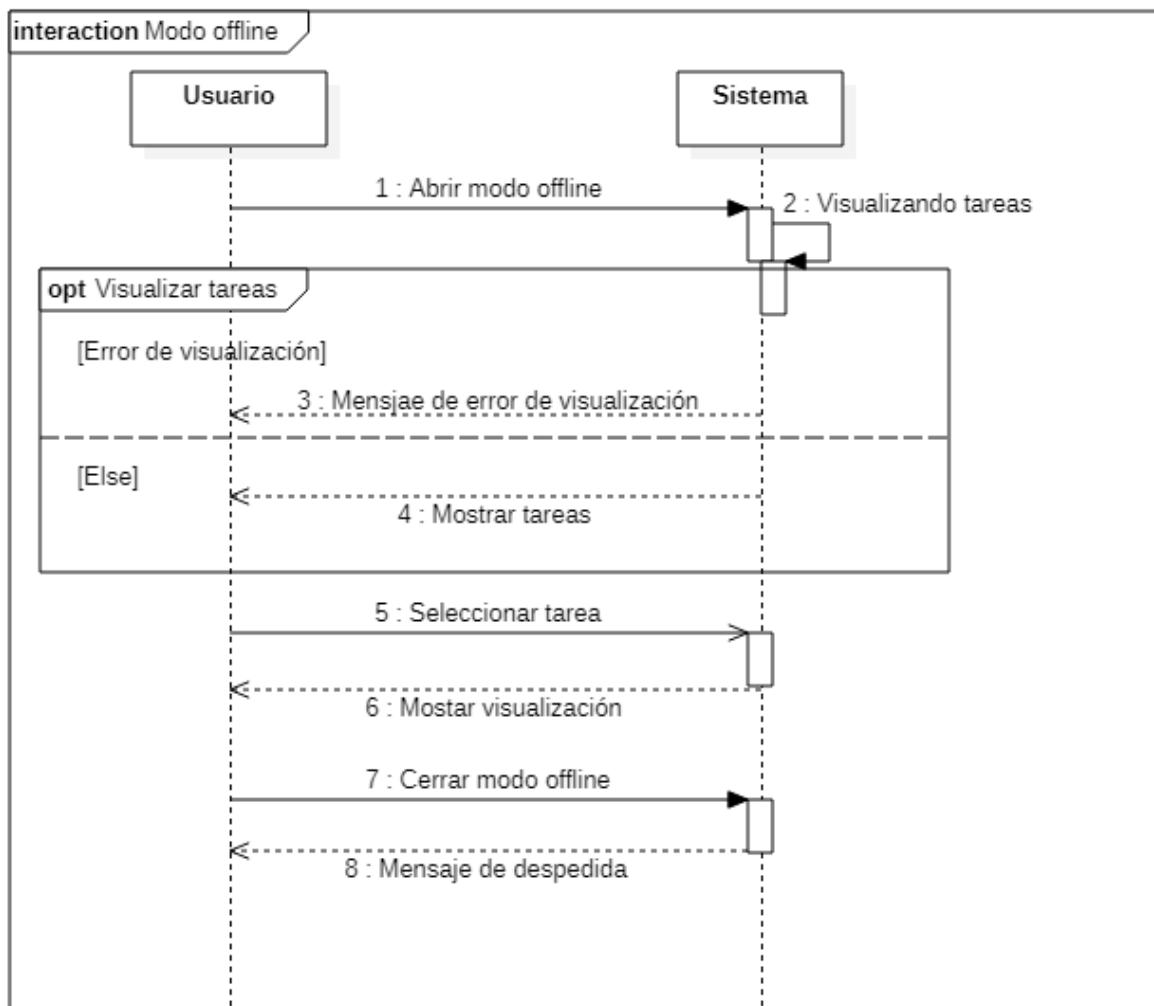
18. Acceder a videollamada activa



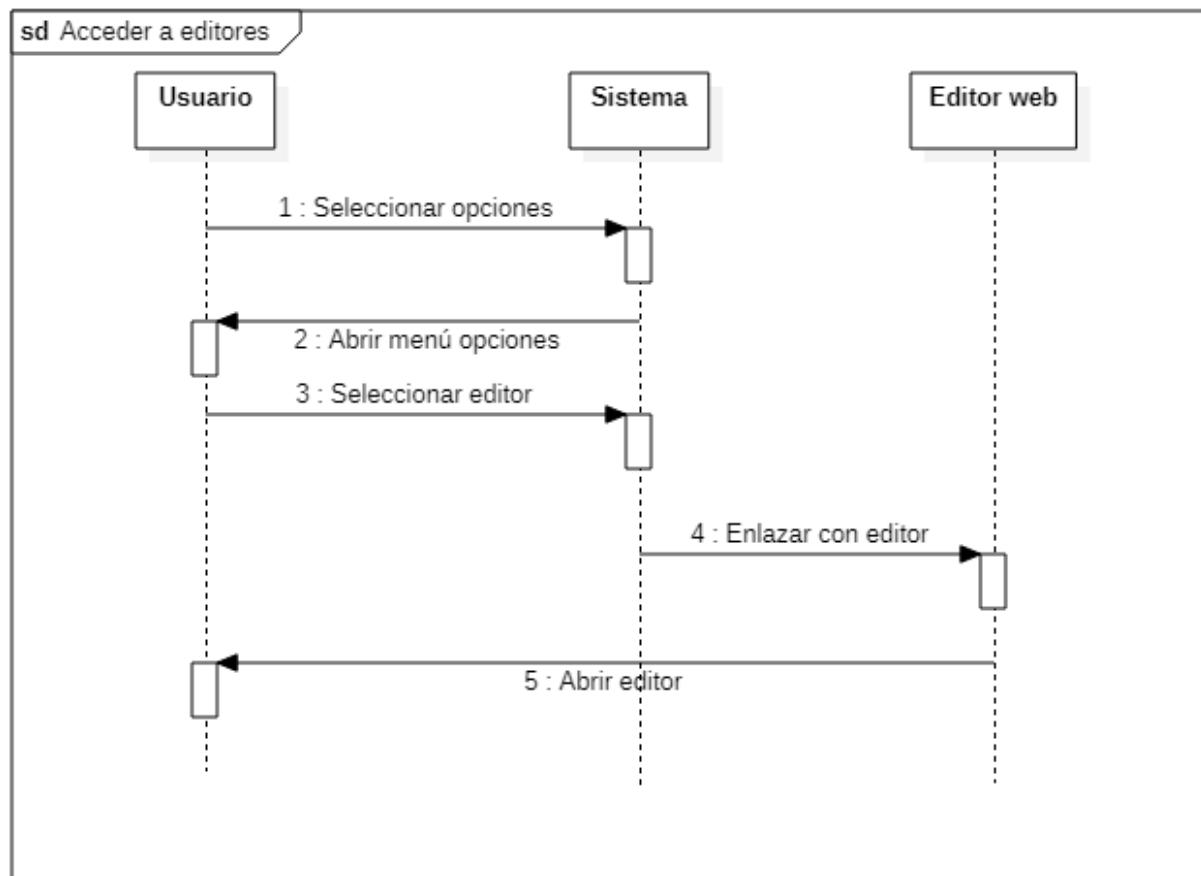
19. Bloquear usuario



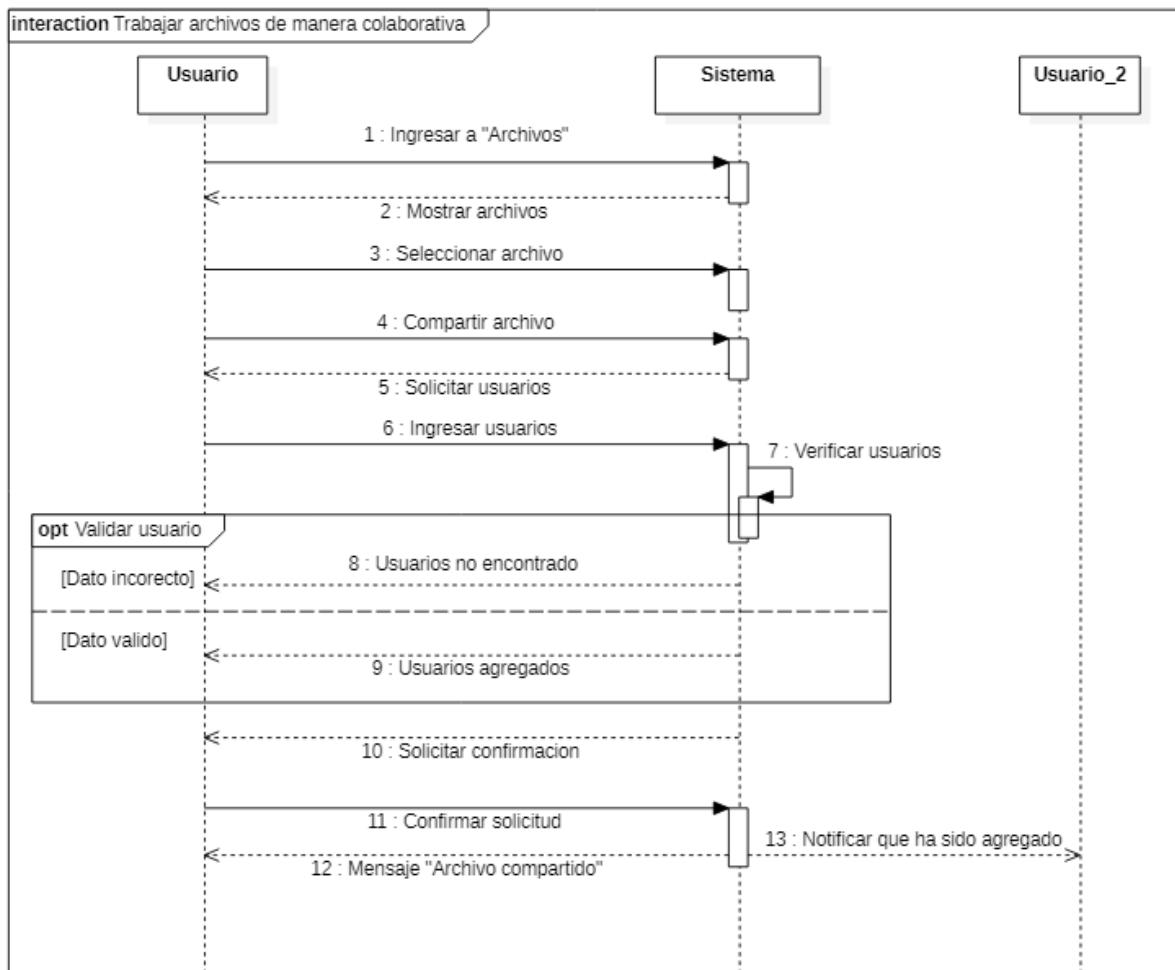
20. Uso offline para visualizar tareas



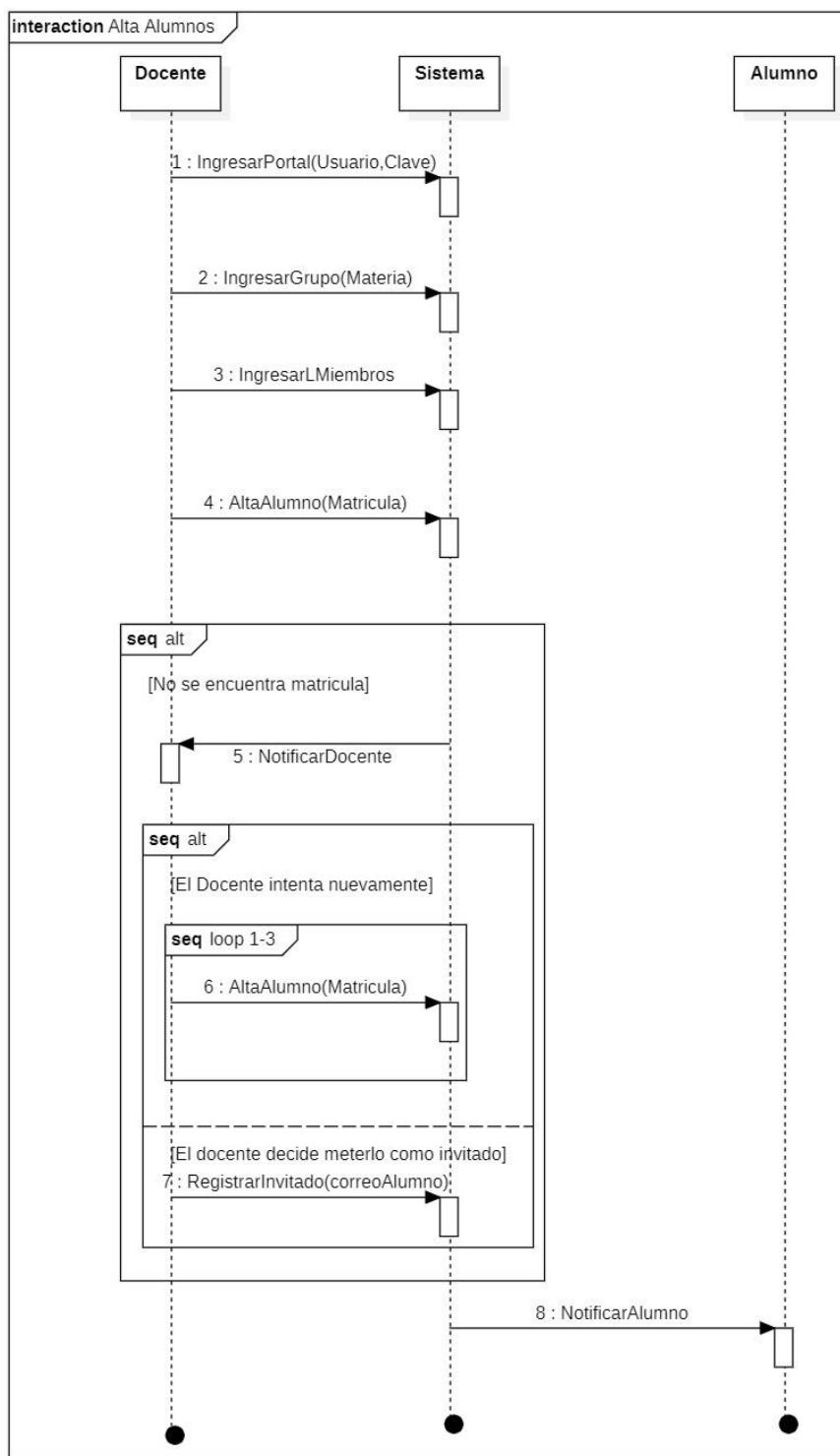
21. Acceder a editores externos



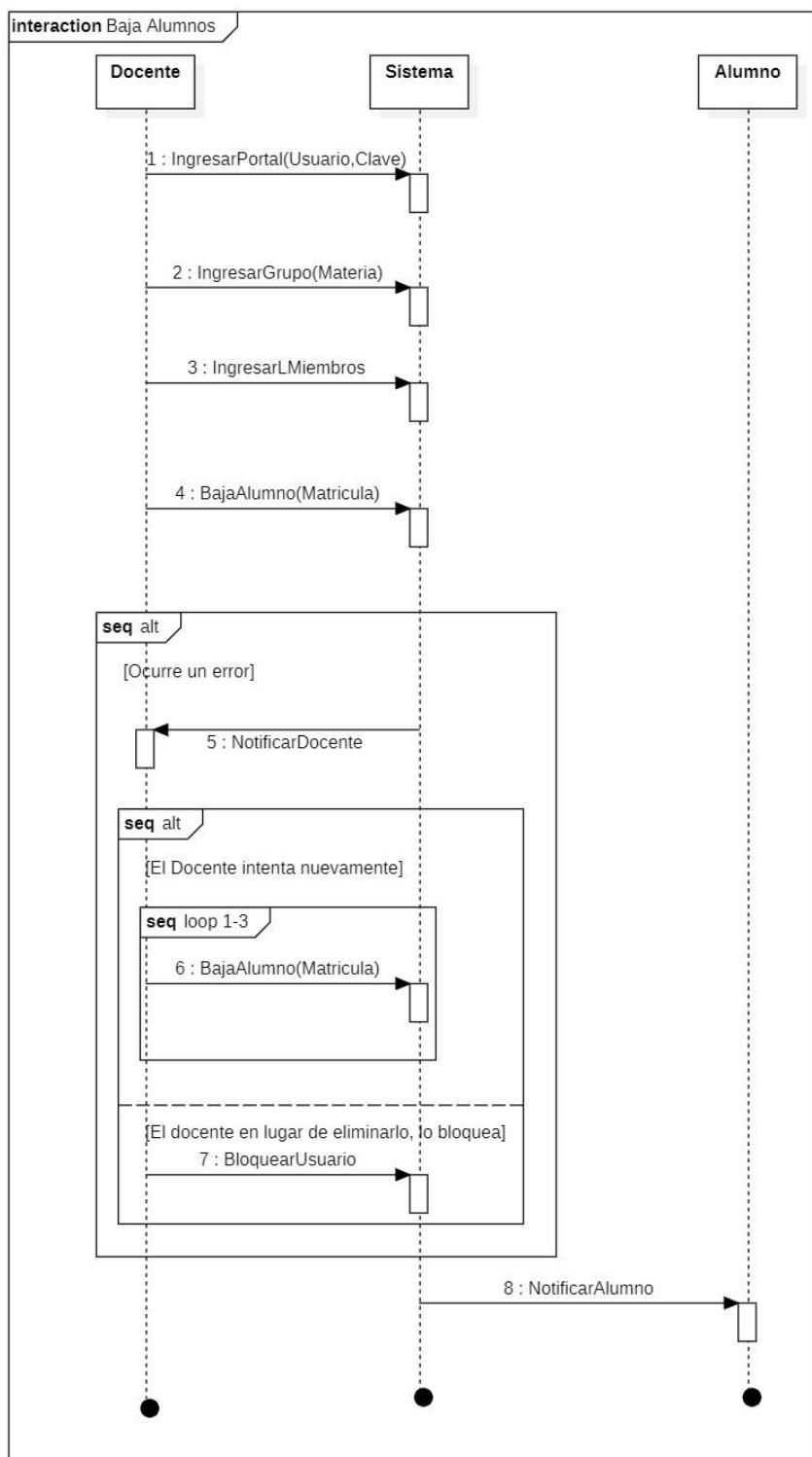
22. Trabajar documentos de manera colaborativa



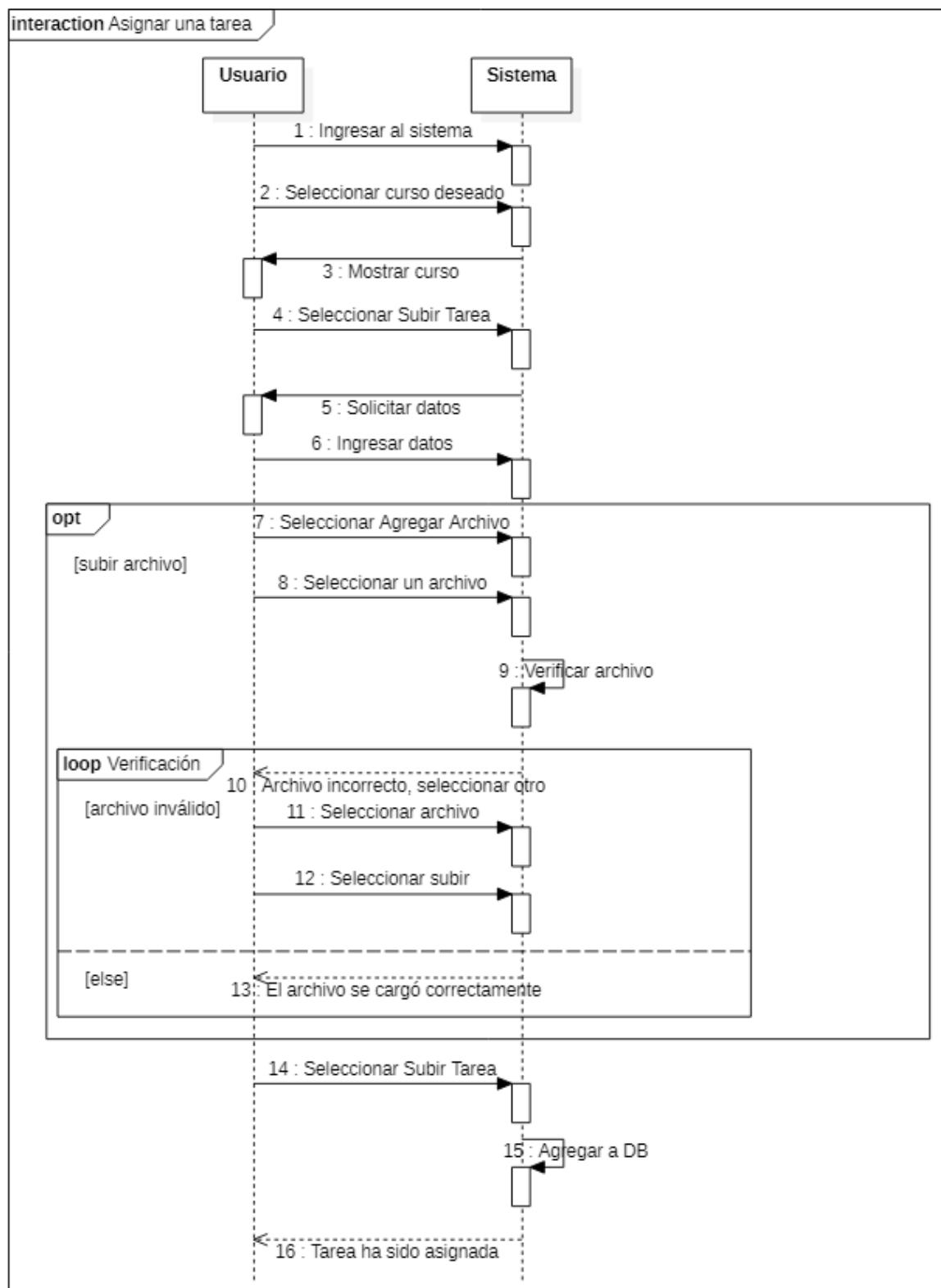
23. Alta Alumnos



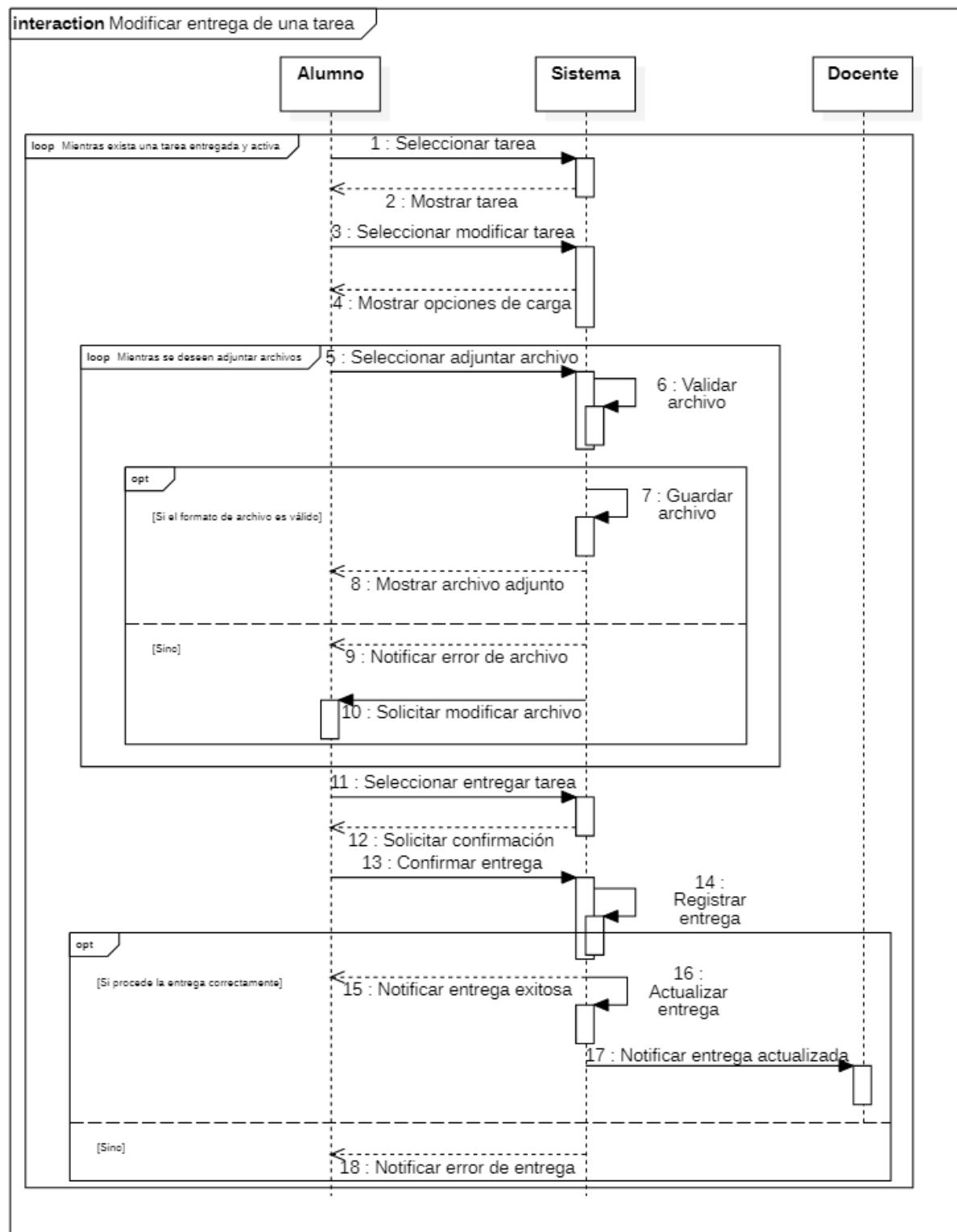
24. Baja Alumnos



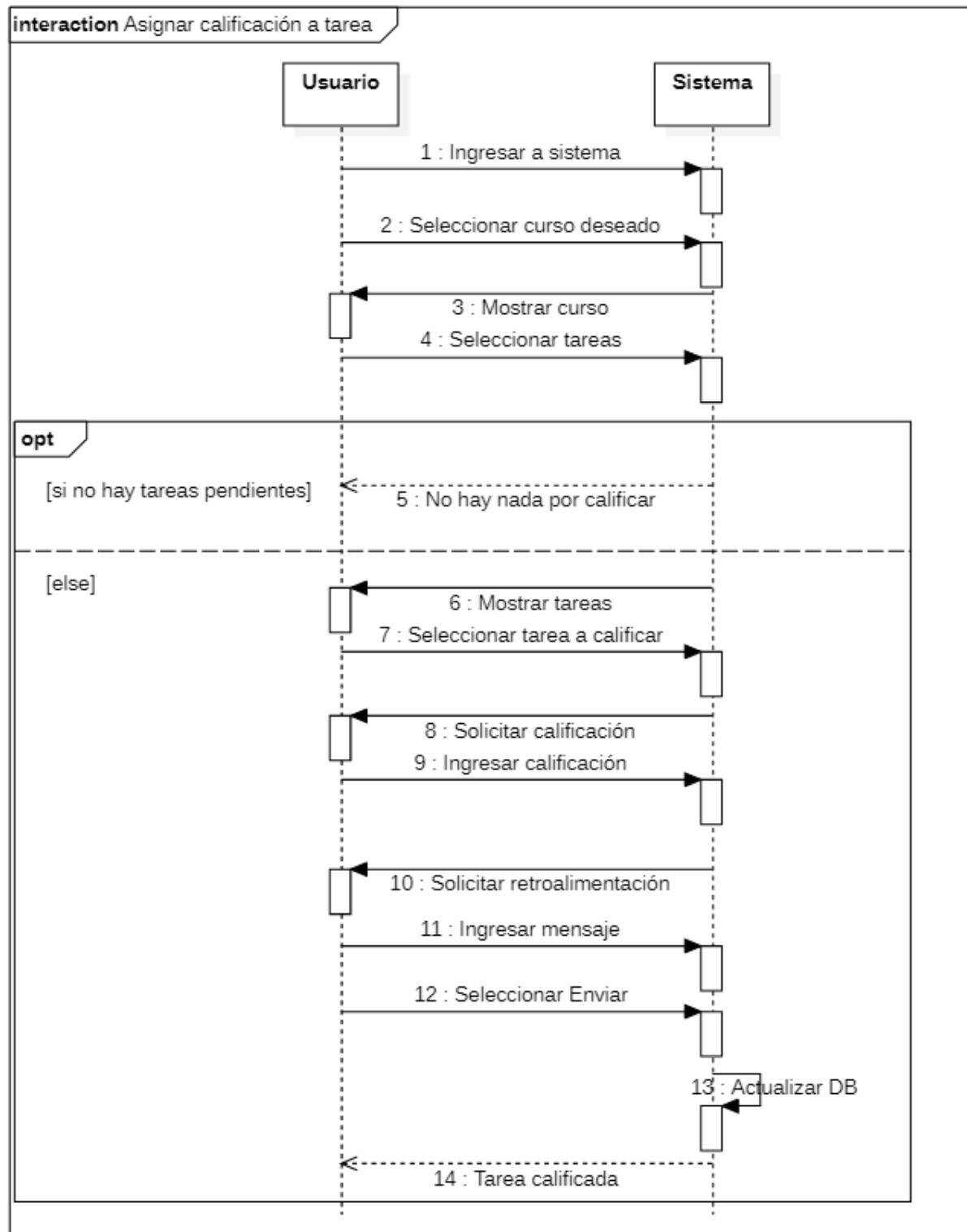
25. Asignar una tarea



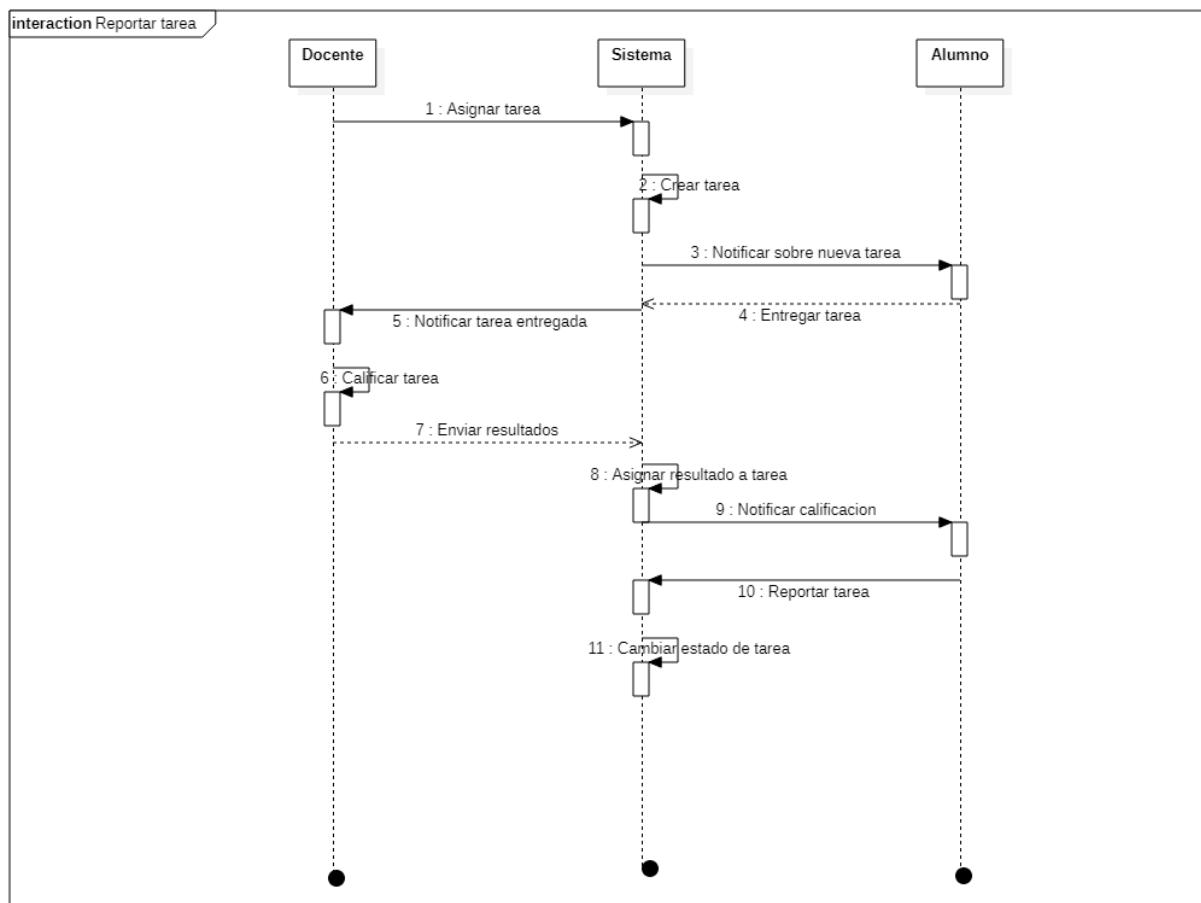
26. Modificar entrega de una tarea



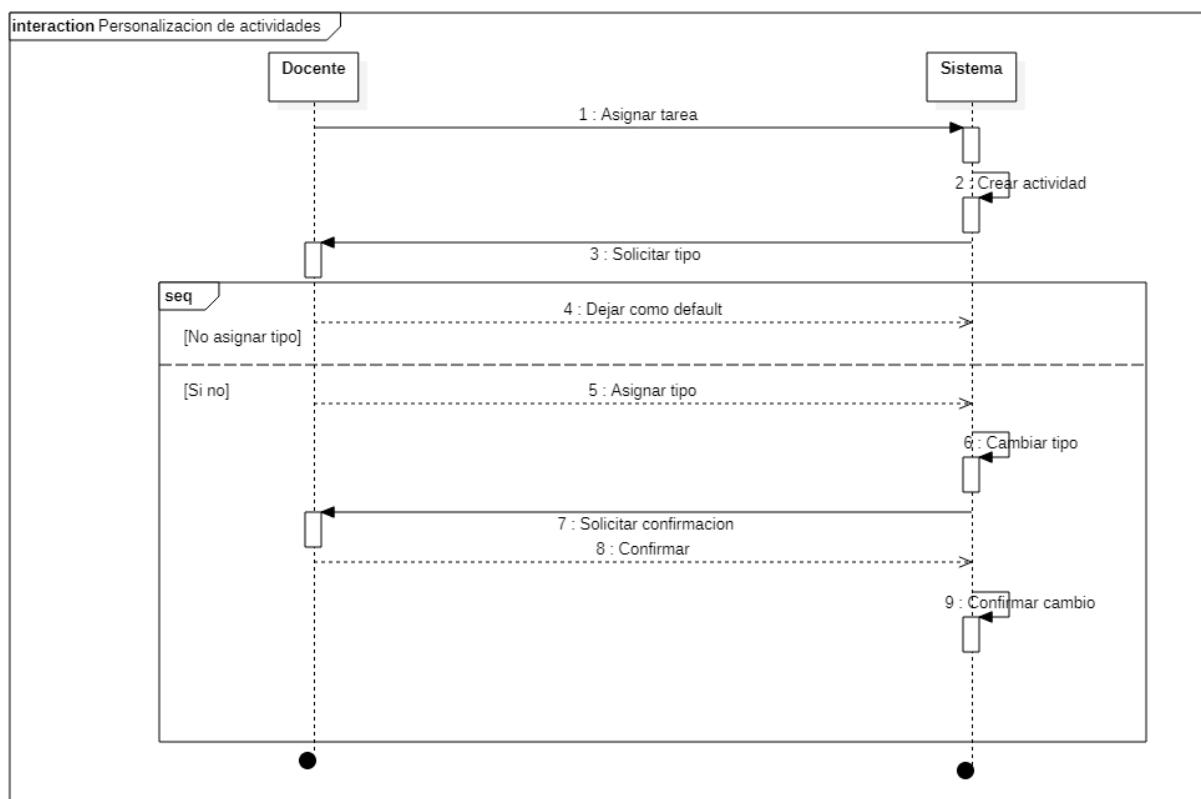
27. Asignar calificación a una tarea



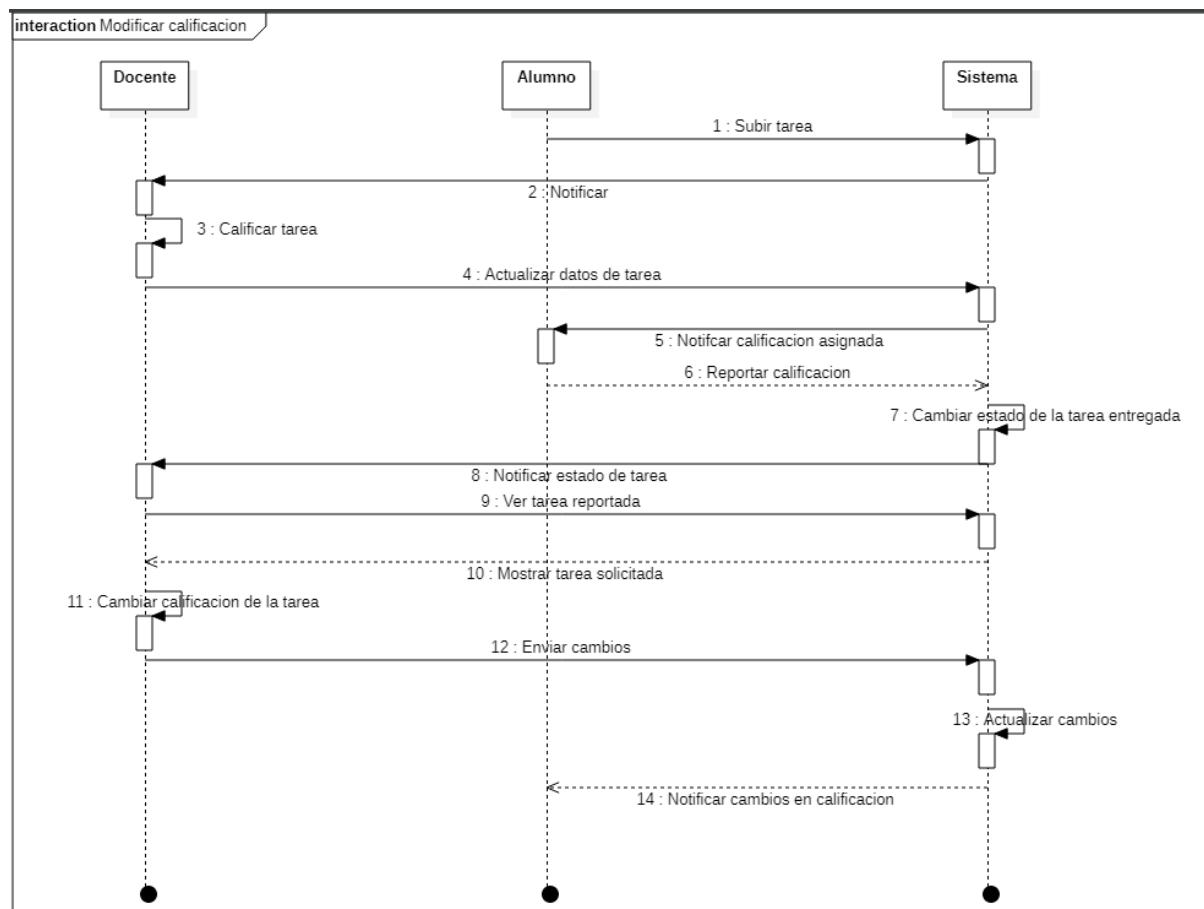
28. Reportar tarea



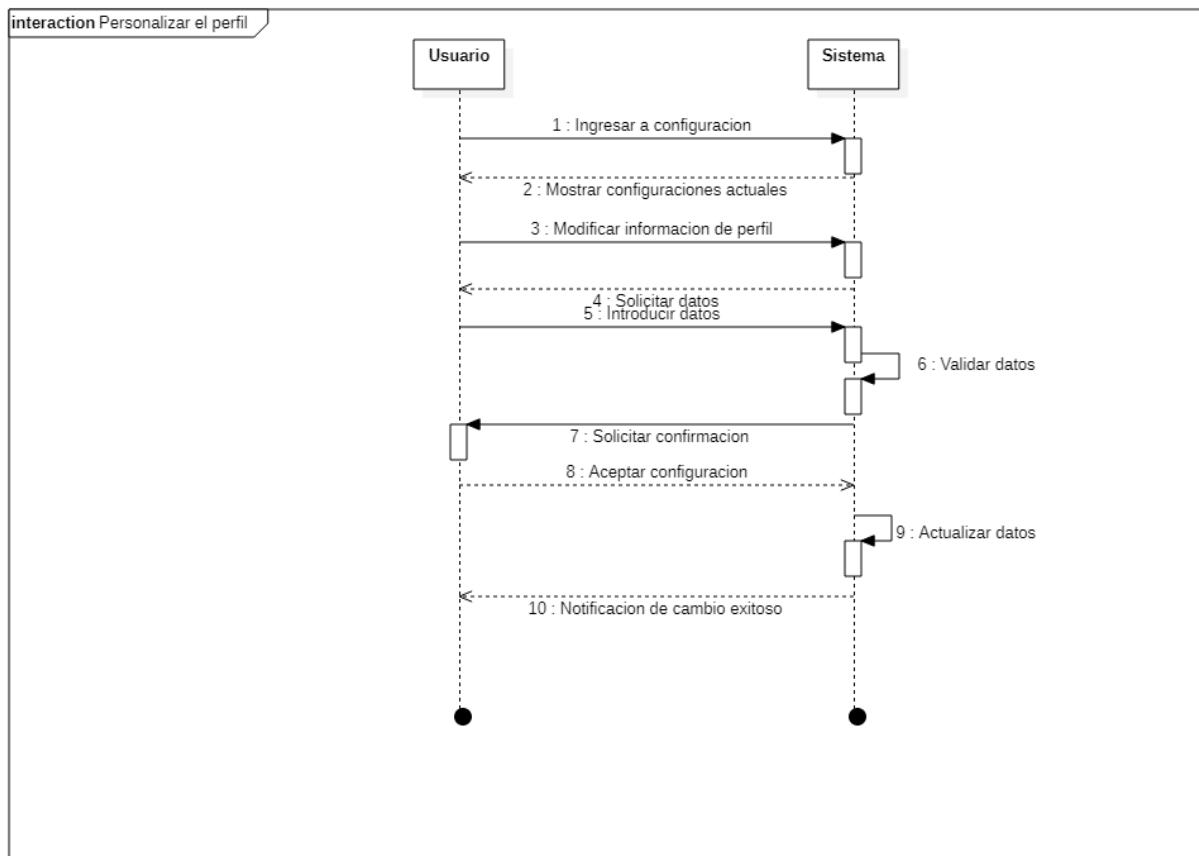
29. Personalización de actividades



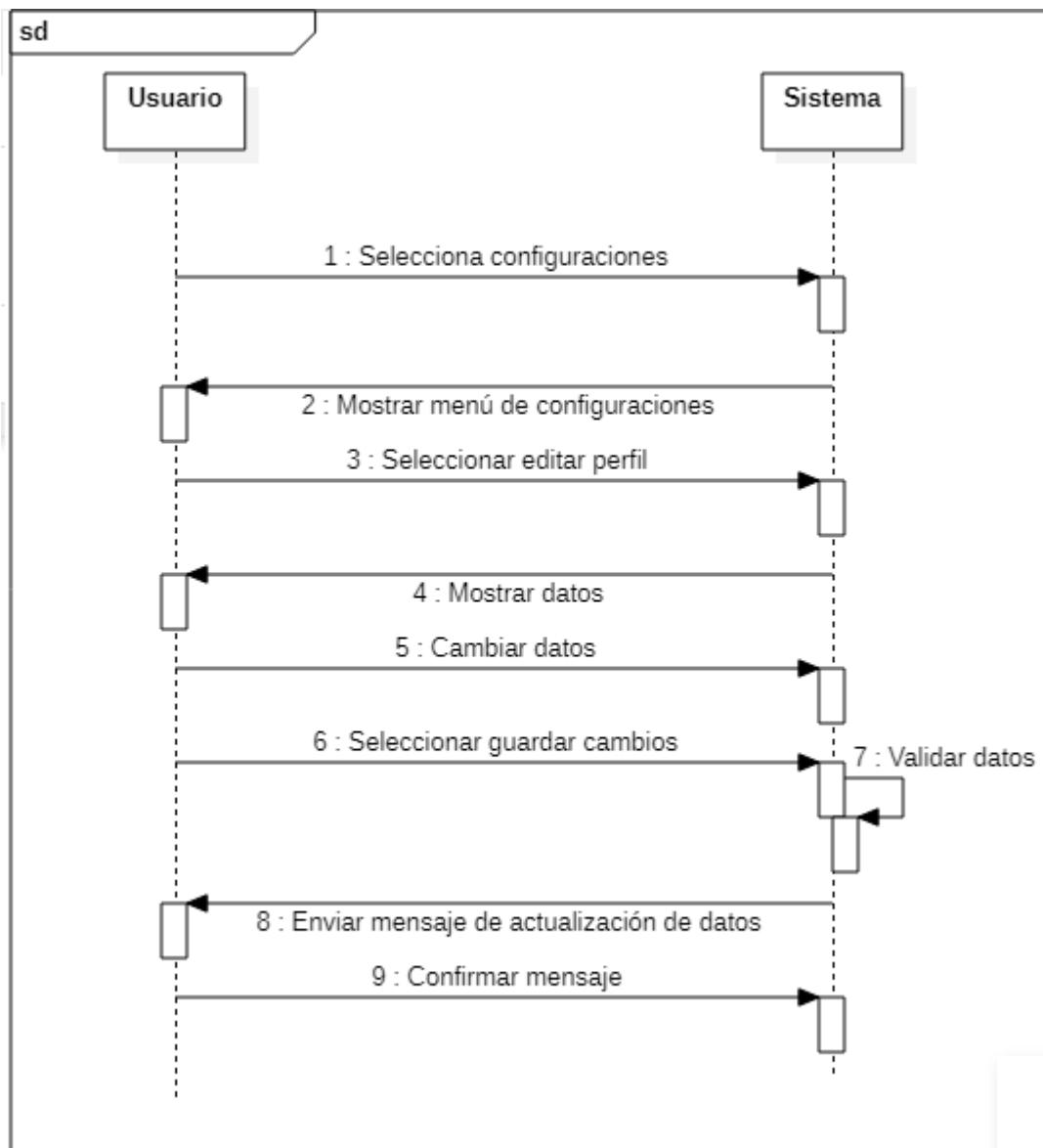
30. Modificar calificación



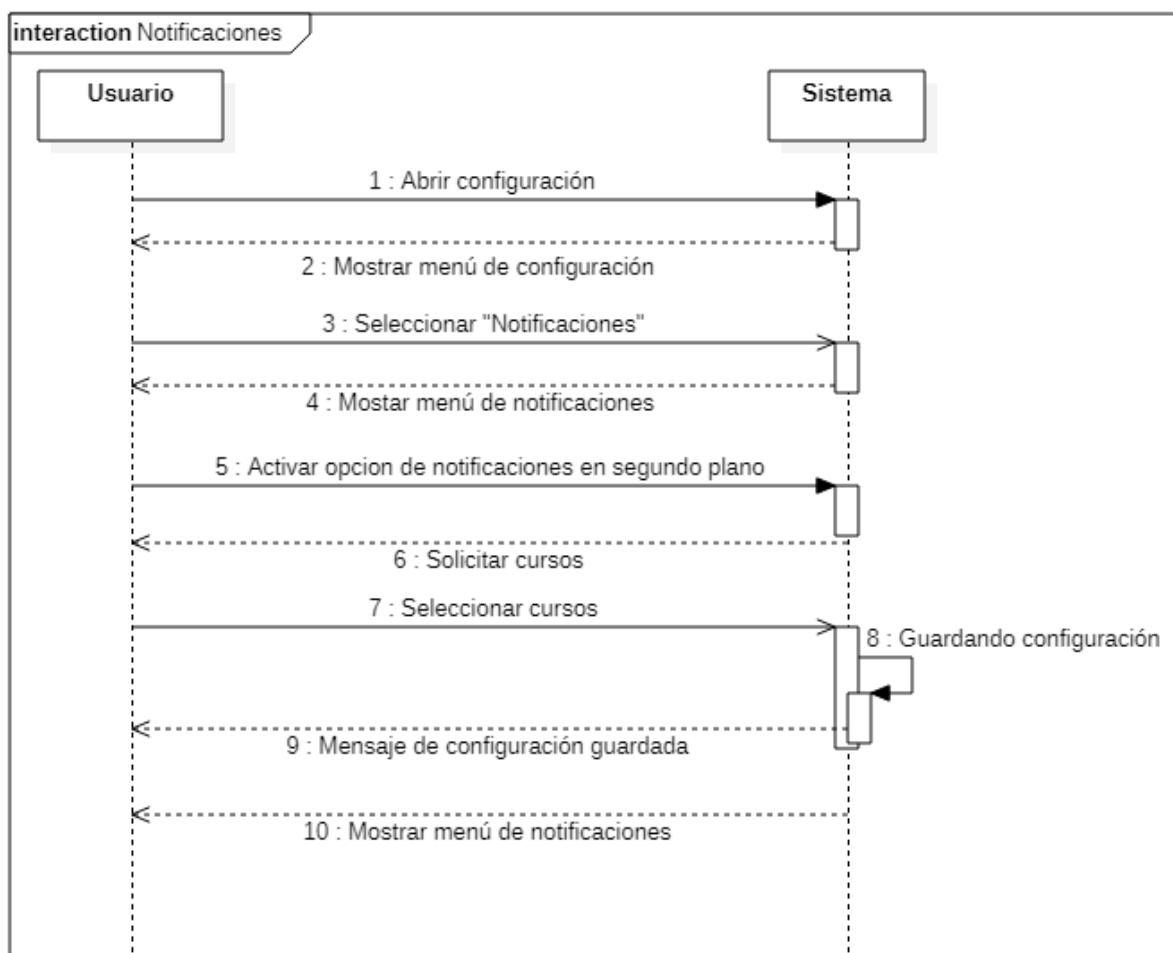
31. Personalización de Perfil



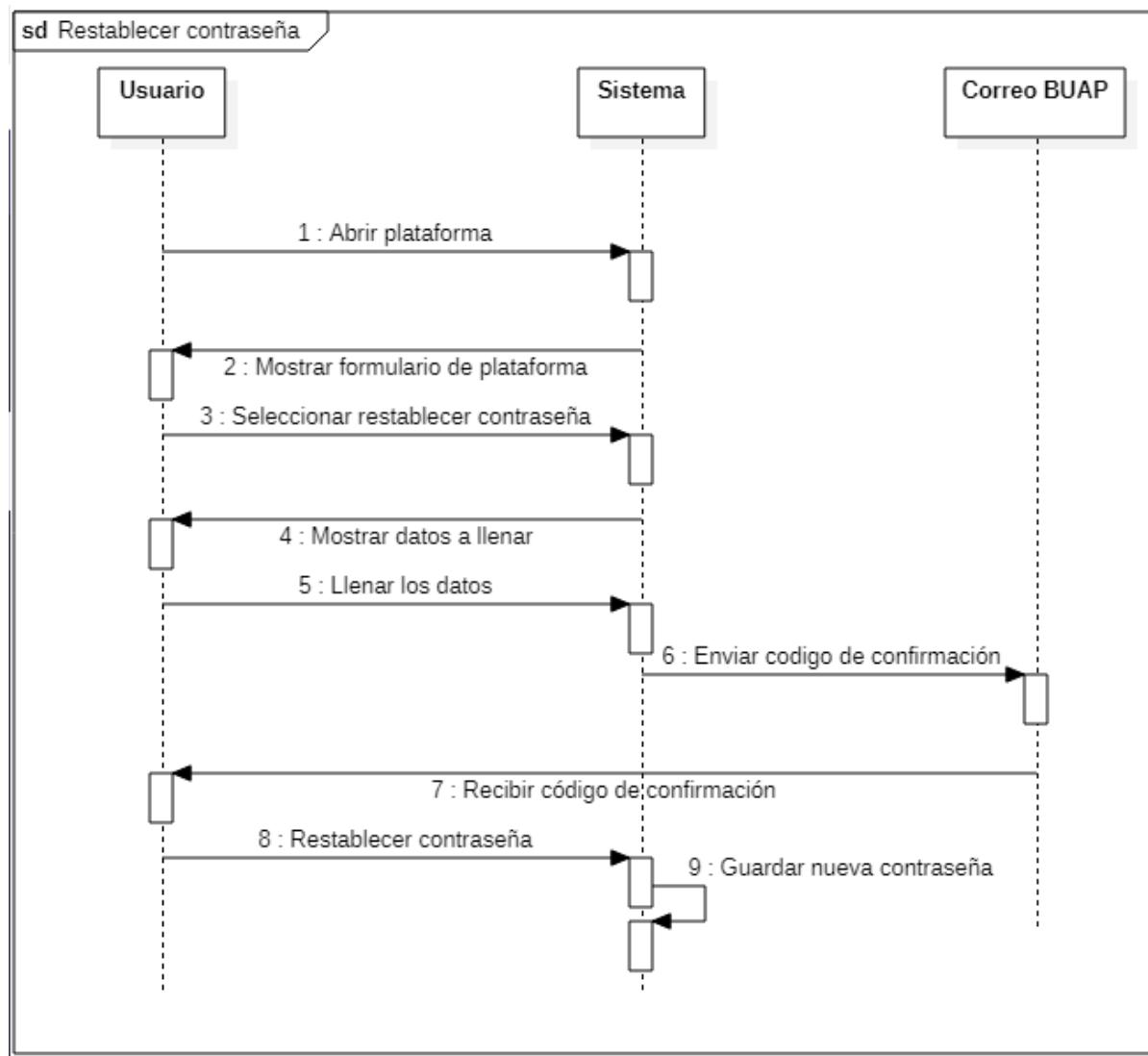
32. Actualizar datos



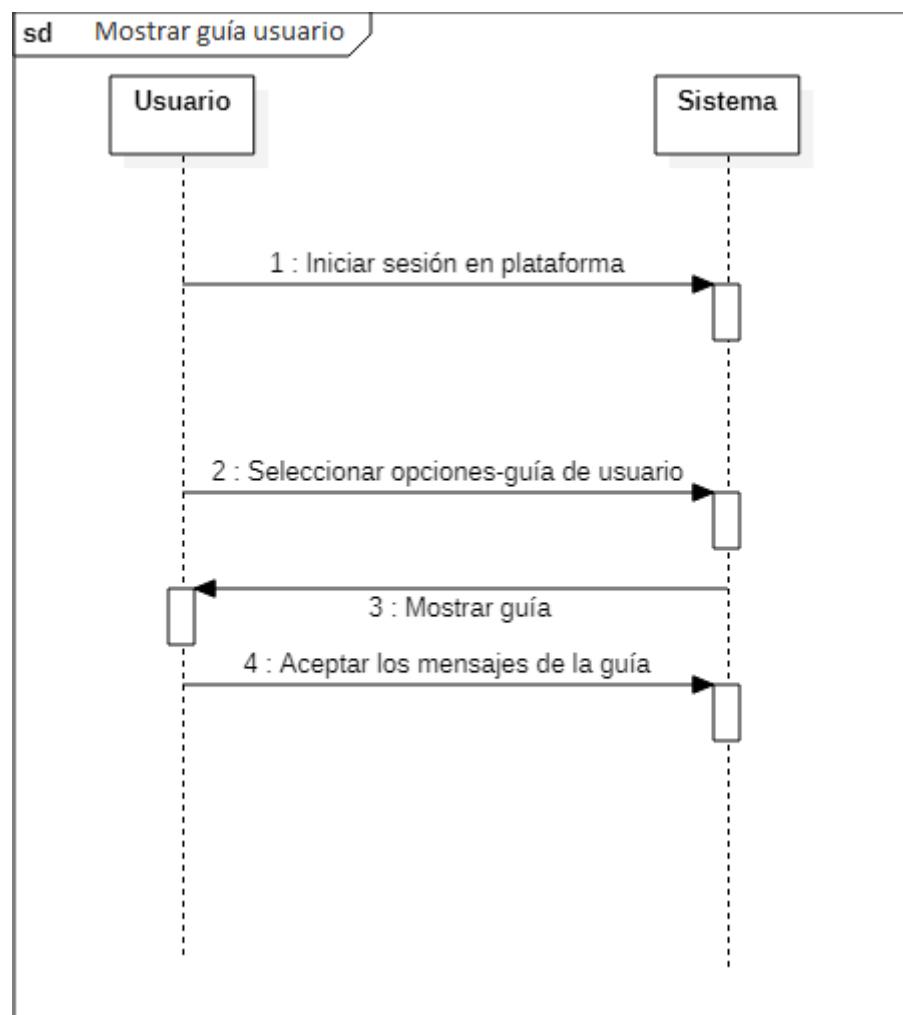
33. Configuración de notificaciones



34. Restablecer contraseña

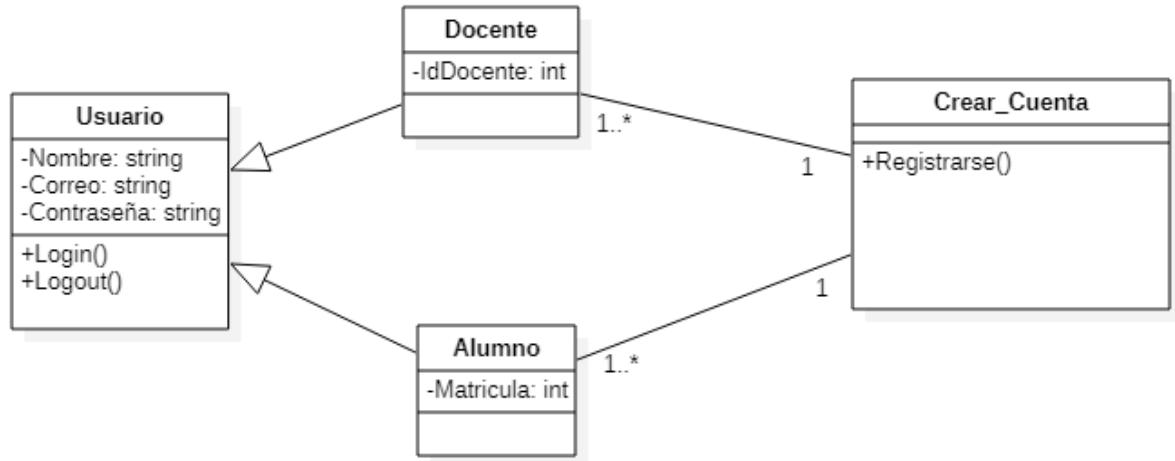


35. Mostrar guía de usuario

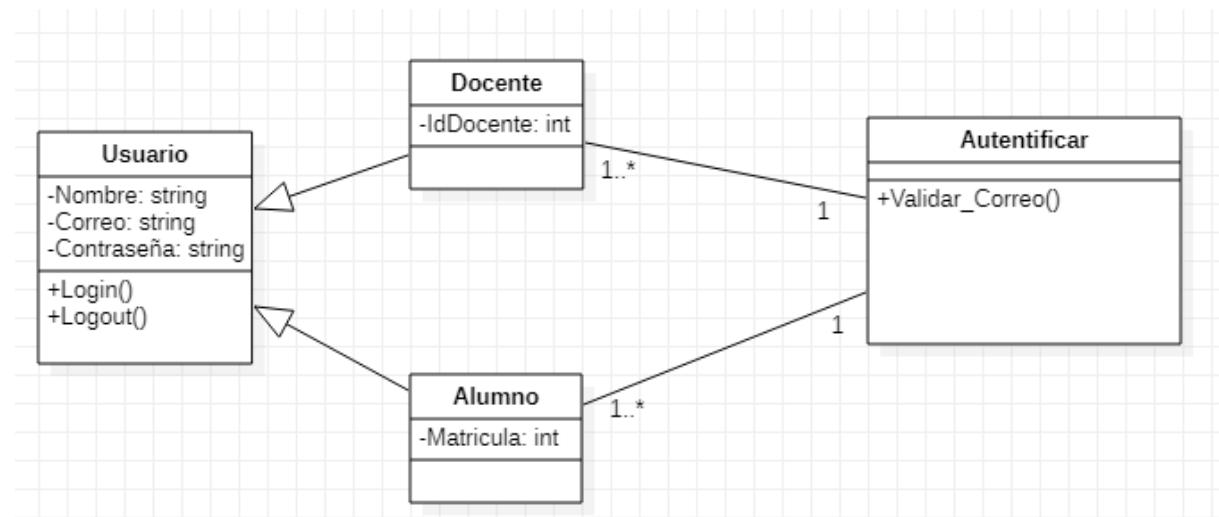


DIAGRAMAS DE CLASES

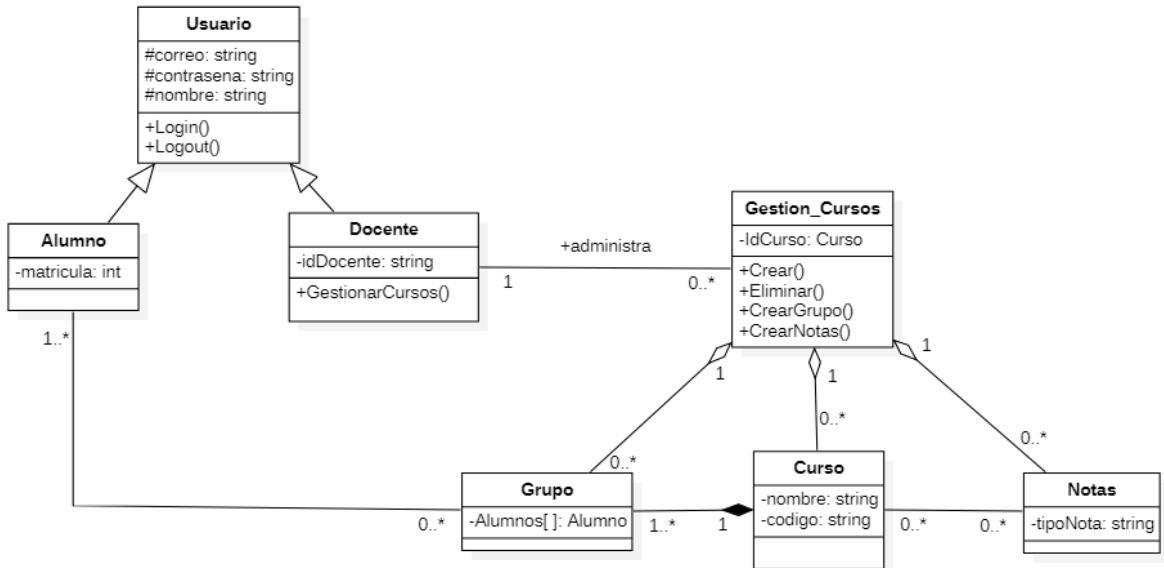
RF01. Crear cuenta de usuario



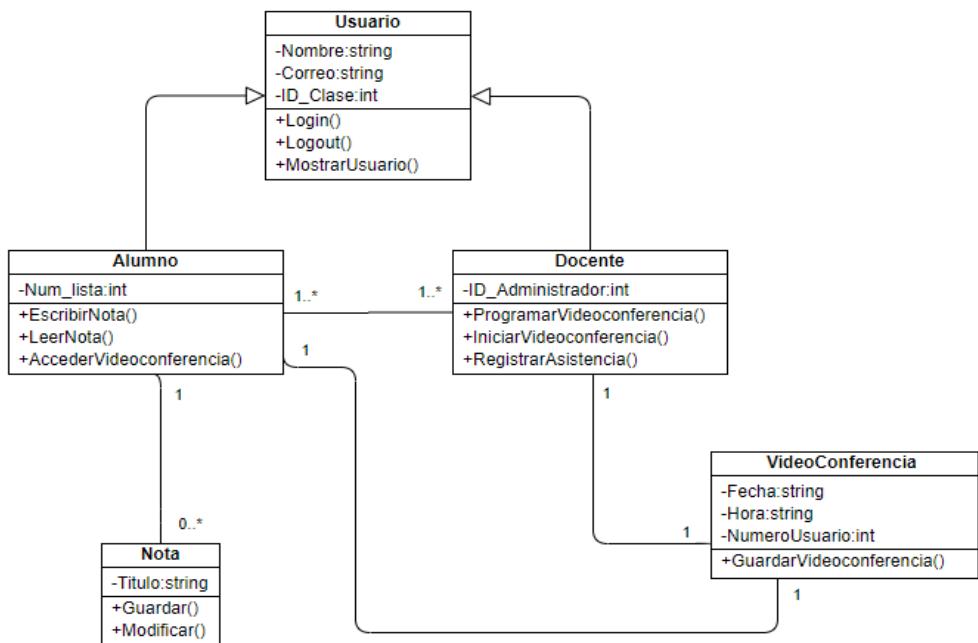
RF02. Autentificación de usuario



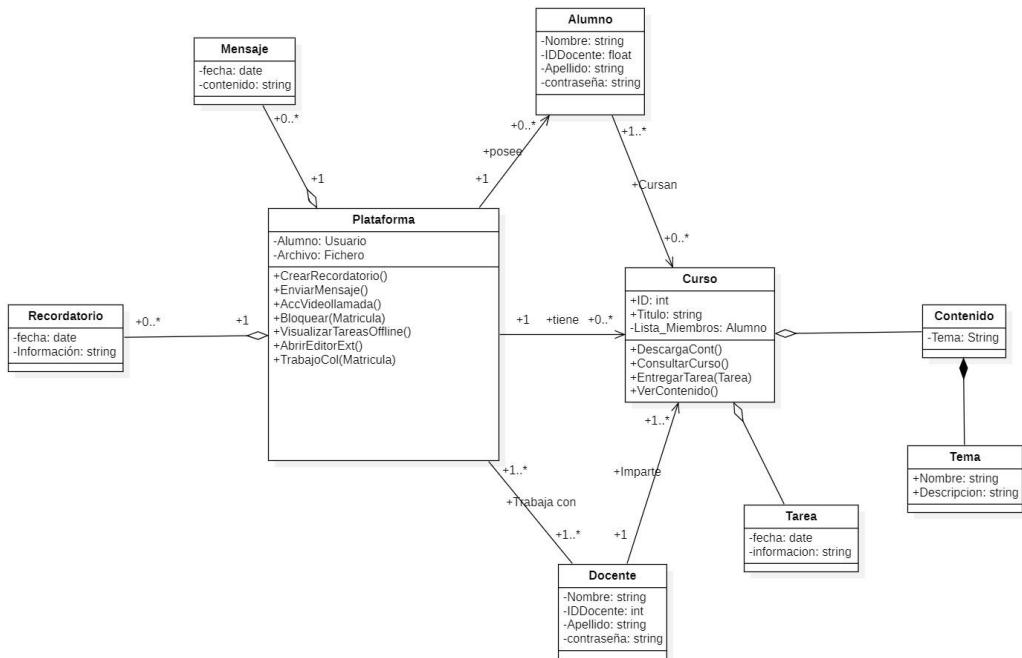
RF03. Gestión de cursos



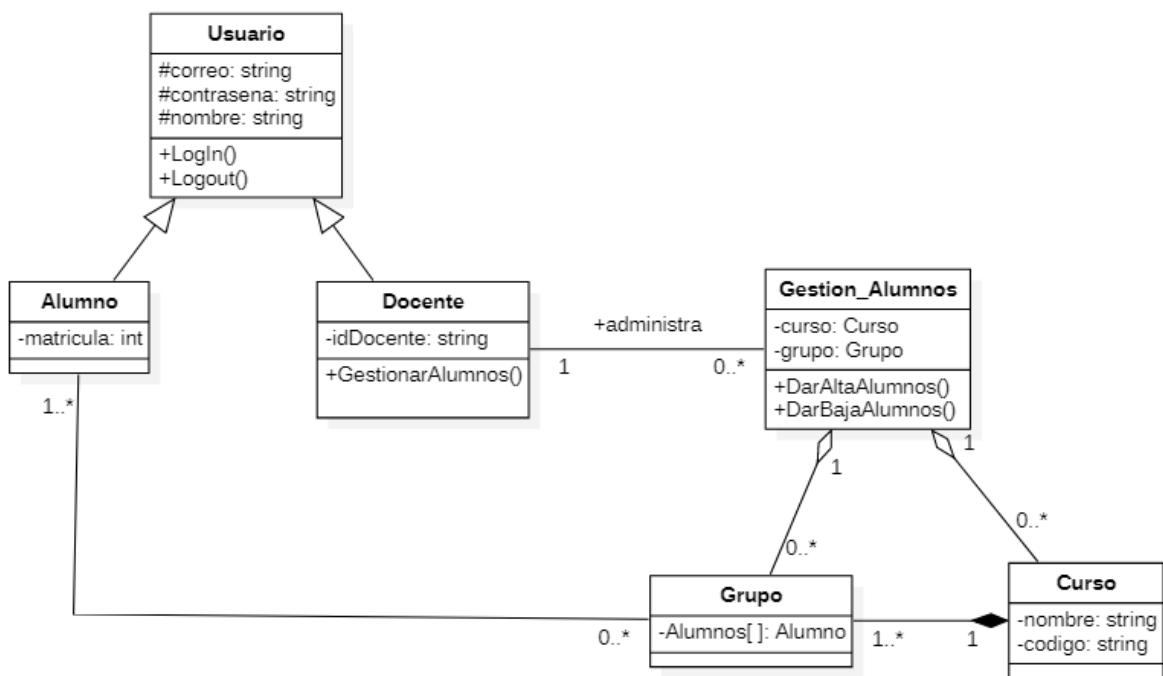
RF04. Gestión de clases



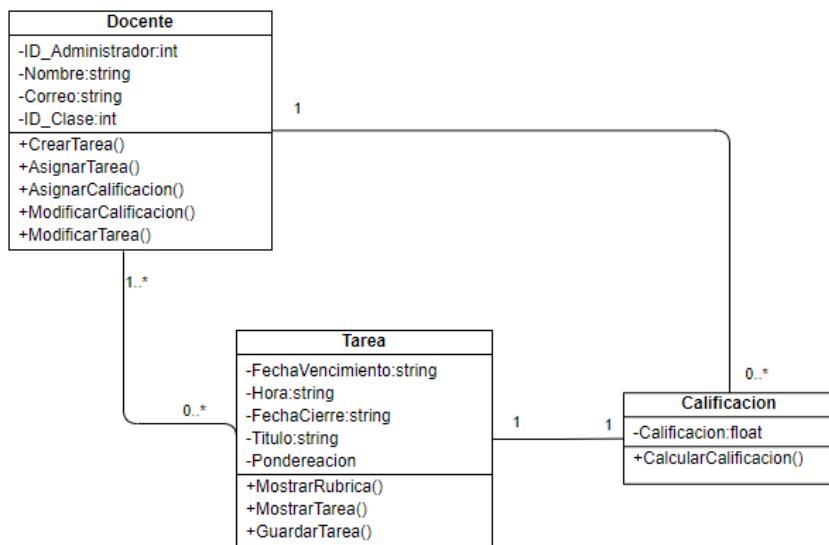
RF05. Gestión de aula virtual



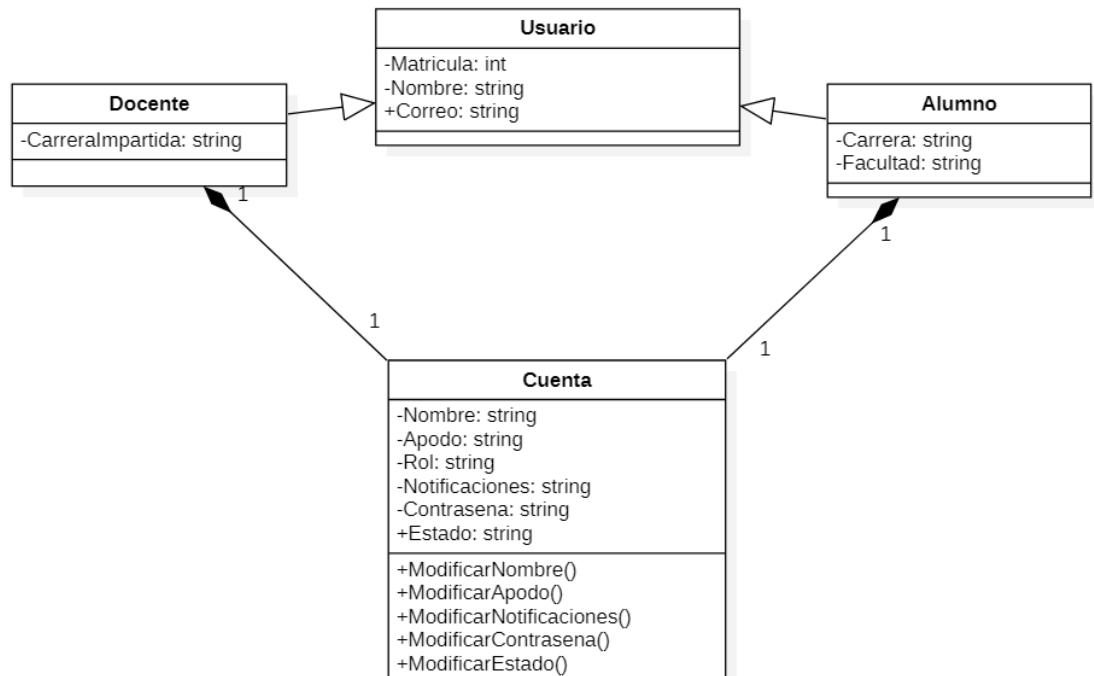
RF06. Gestión de alumnos



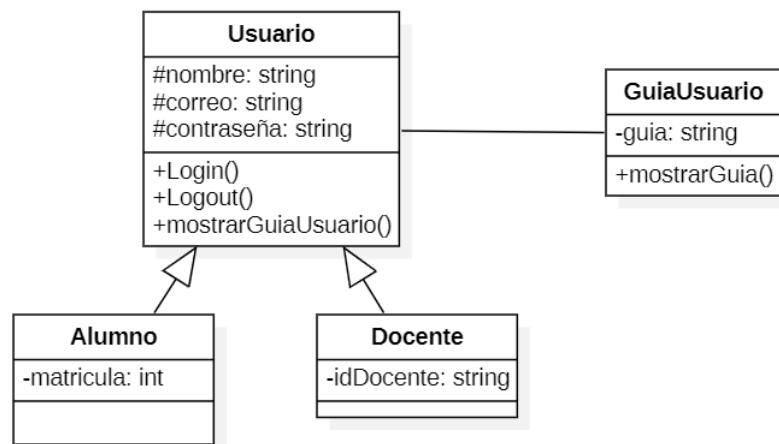
RF07. Gestión de tareas



RF08. Modificar cuenta



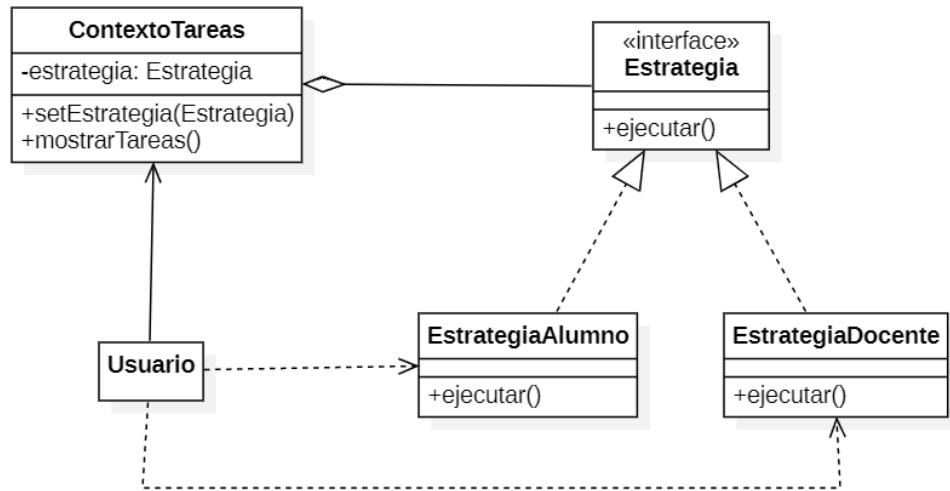
RF09. Mostrar guía de usuario



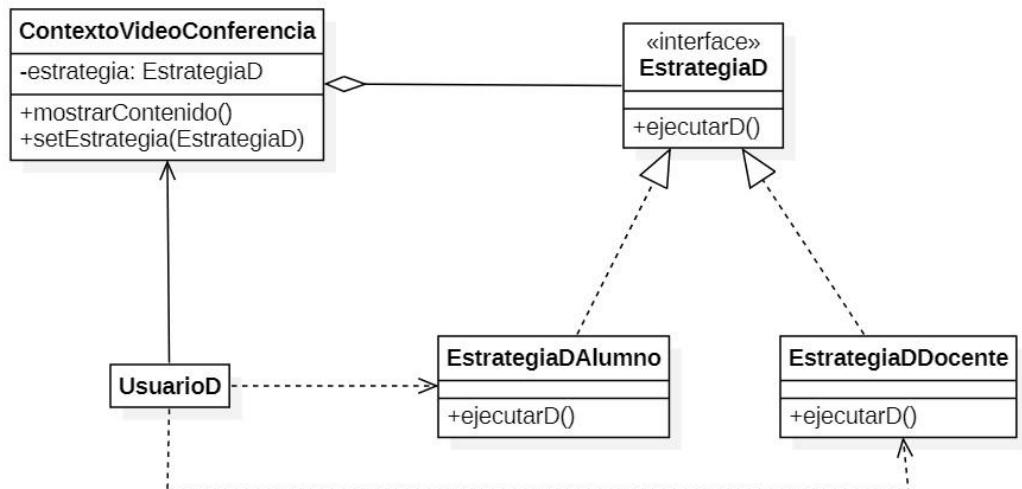
ANEXO 3. Patrones de diseño

STRATEGY

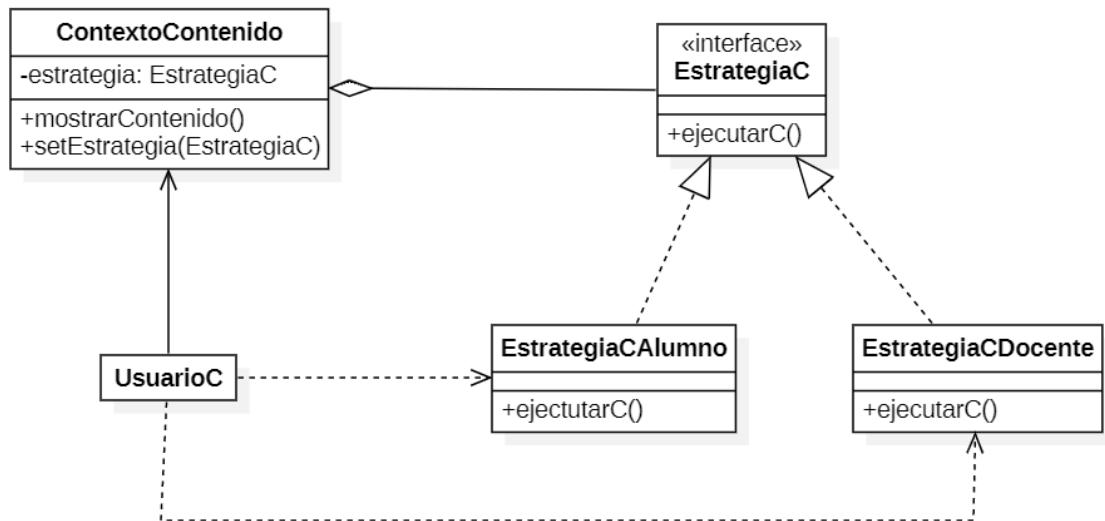
Estrategia Tareas



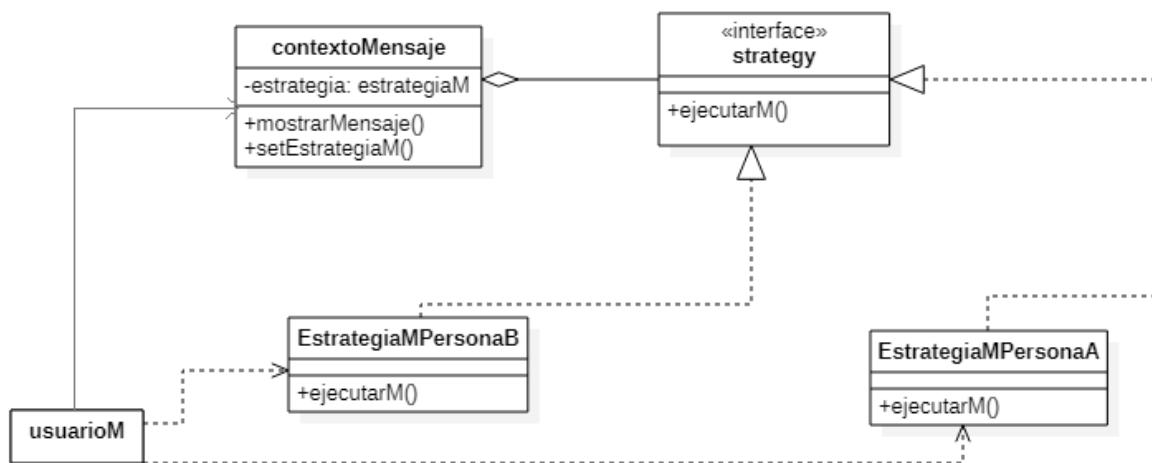
Estrategia Videoconferencia



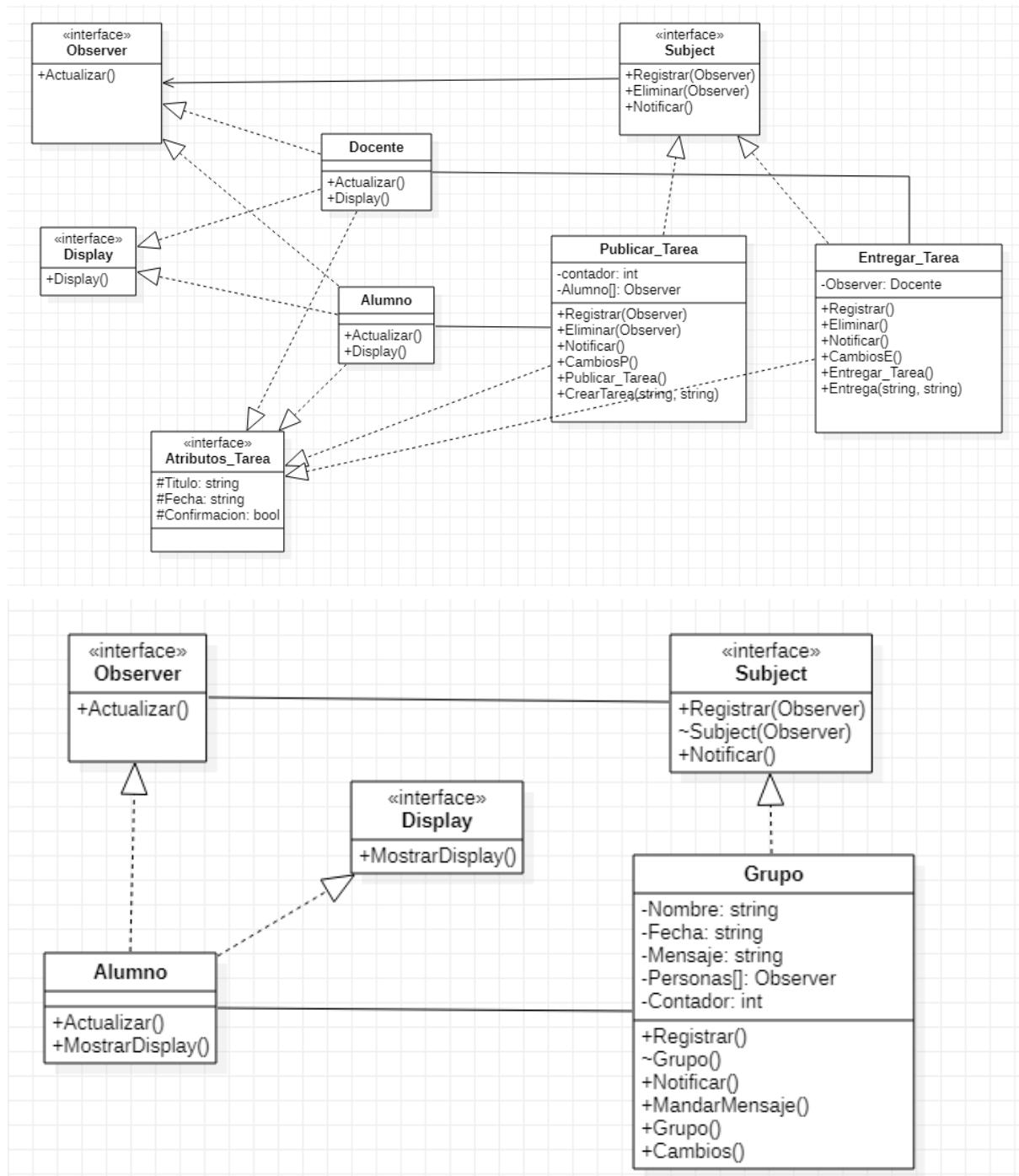
Estrategia Contenido

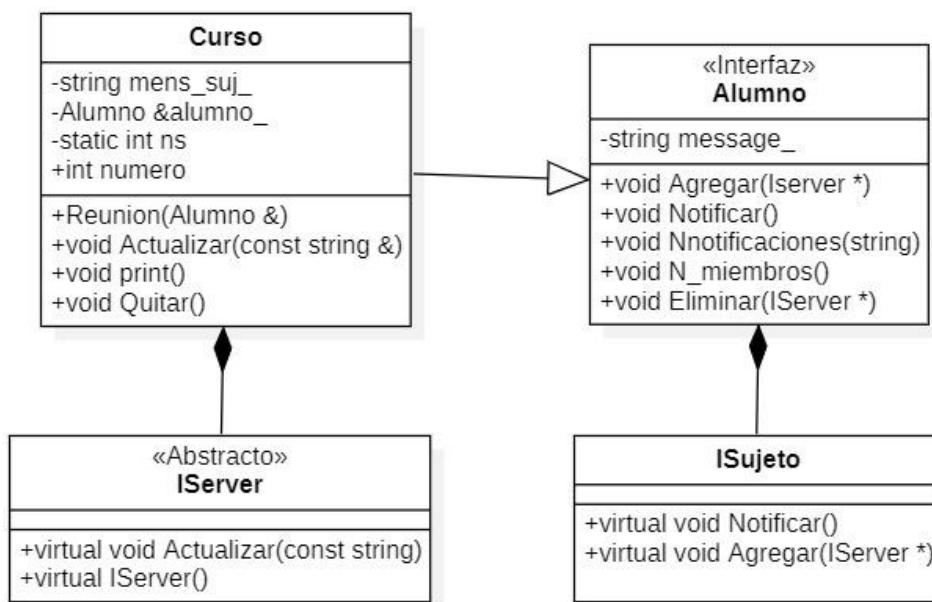
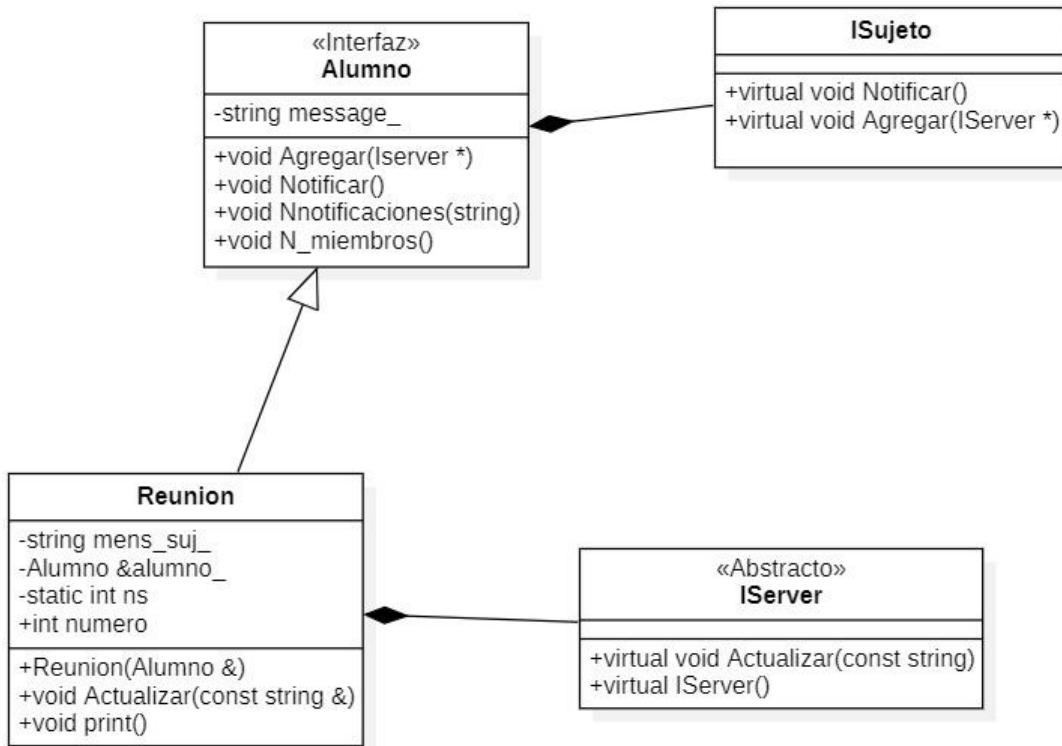


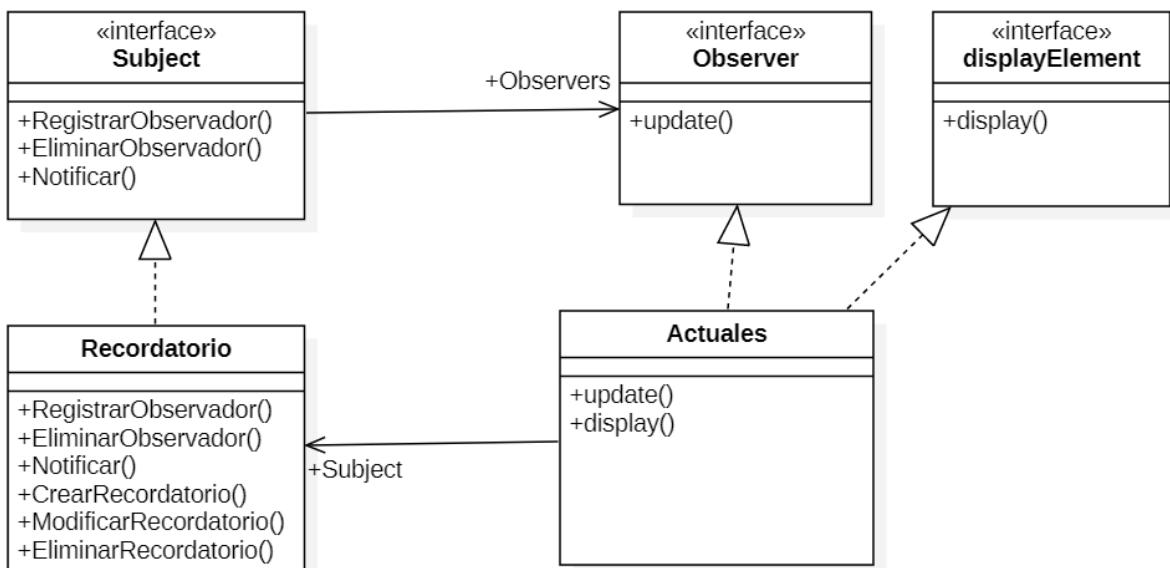
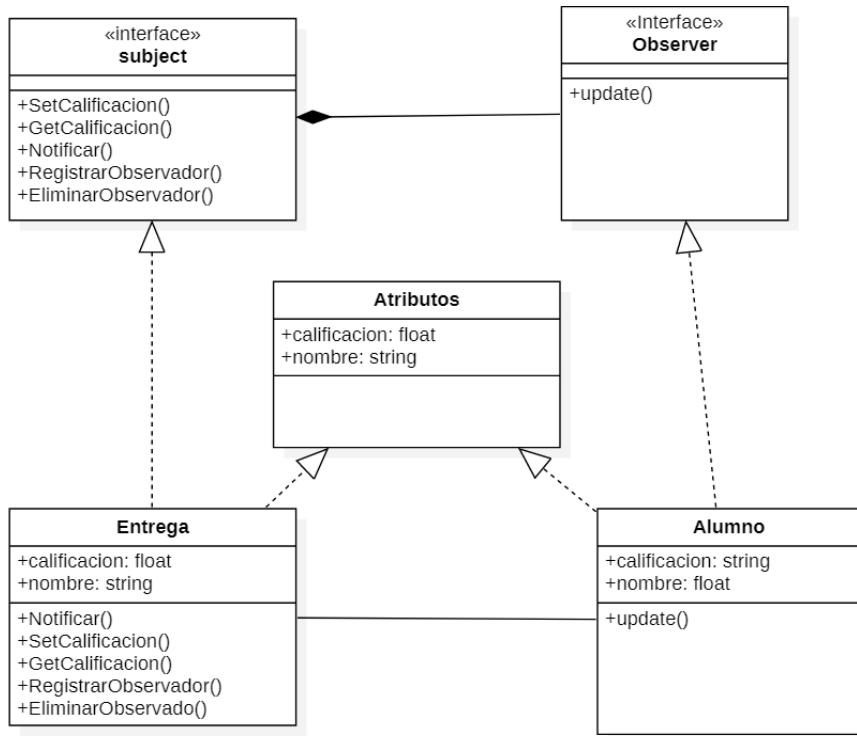
Estrategia Mensajes

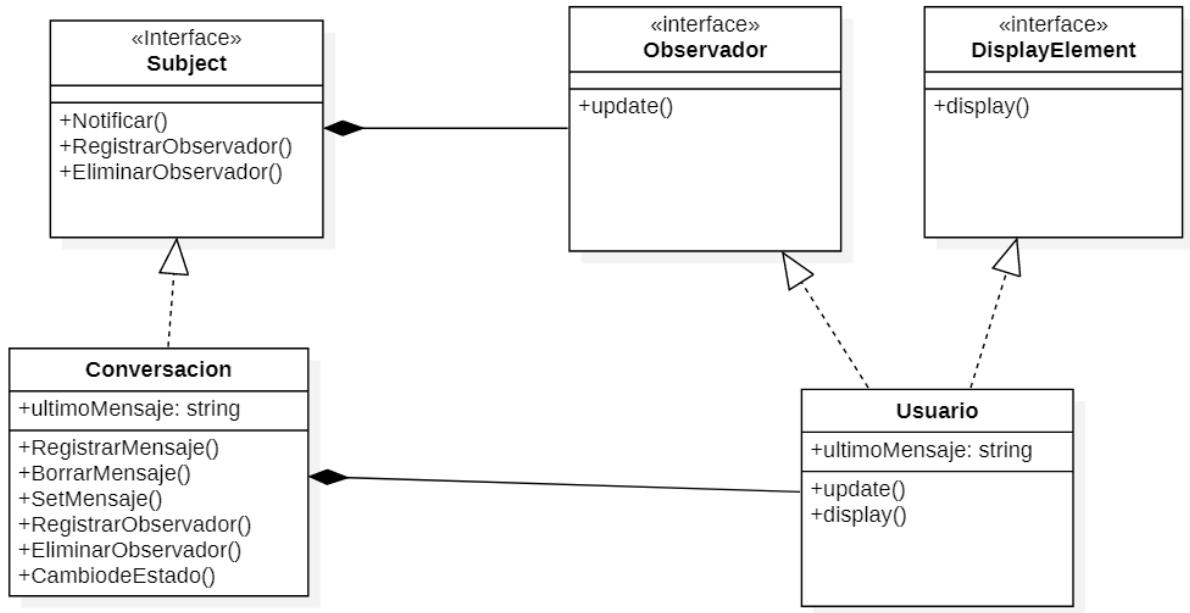


OBSERVER



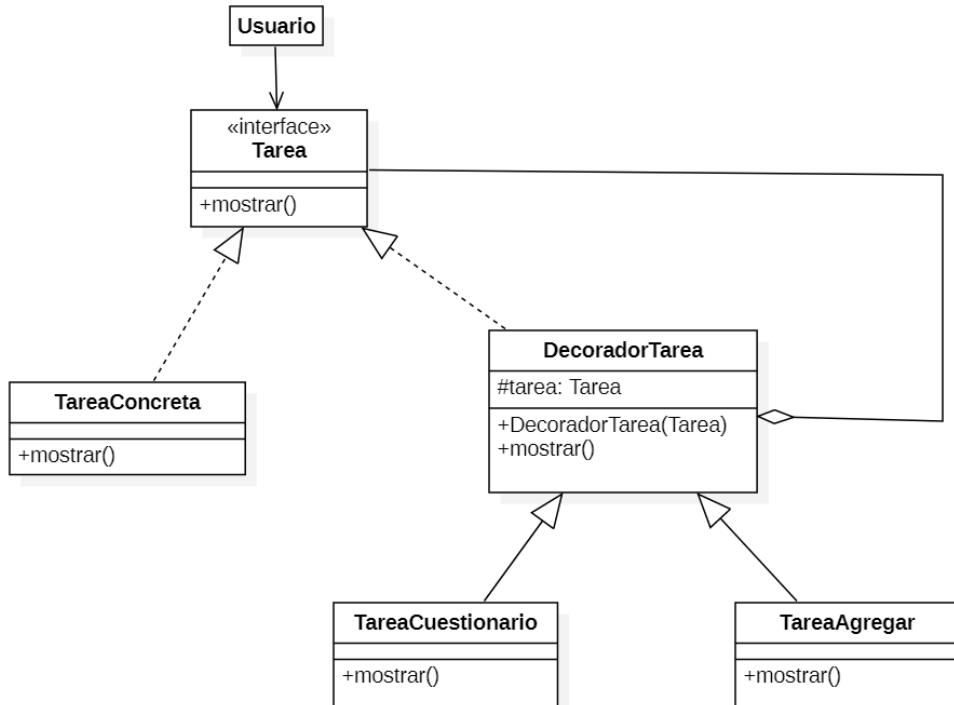




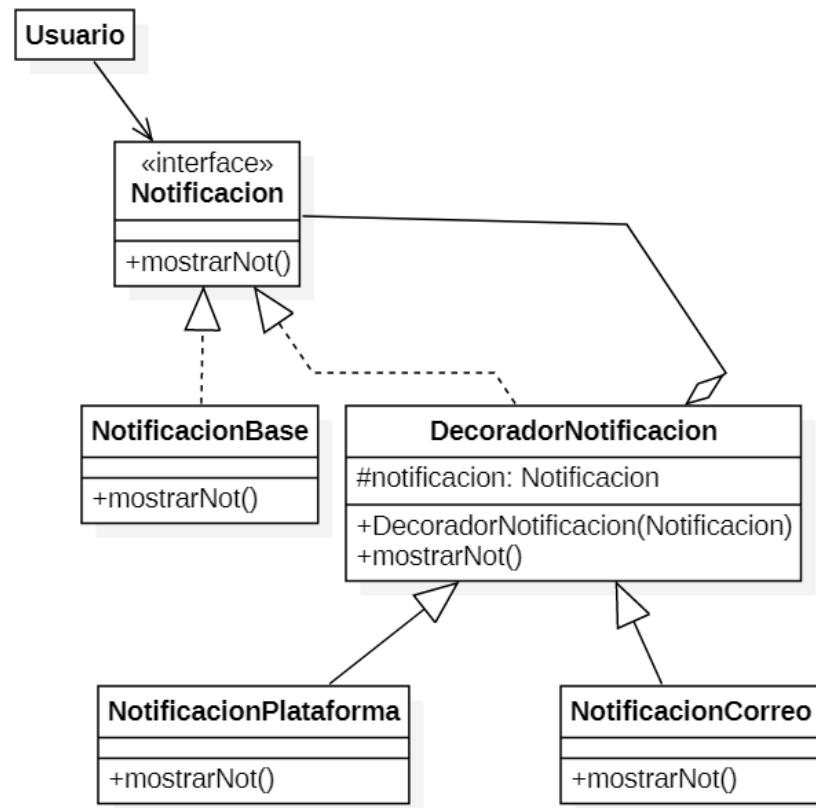


DECORATOR

Decorator Tareas

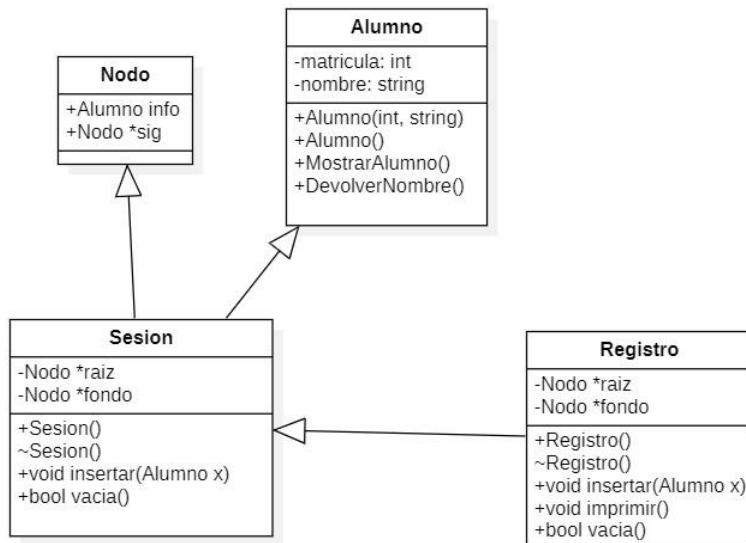


Decorator Notificaciones

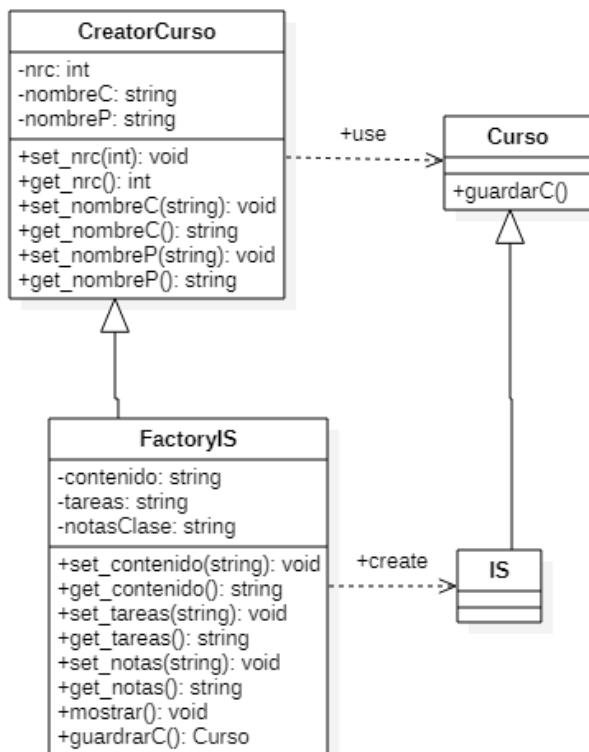


FACTORY

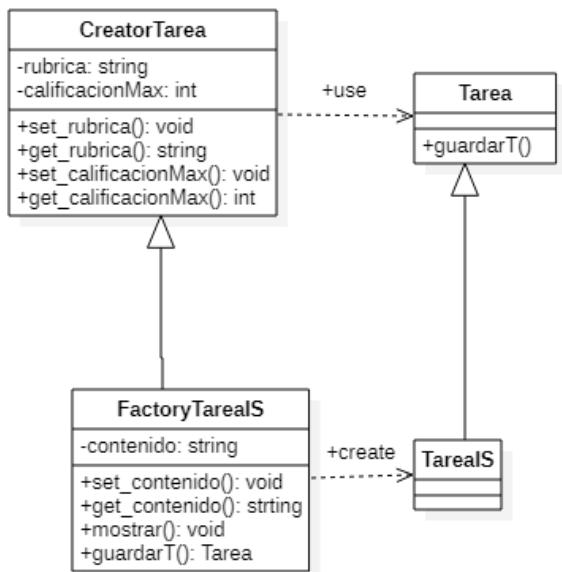
Factory de Toma de asistencia



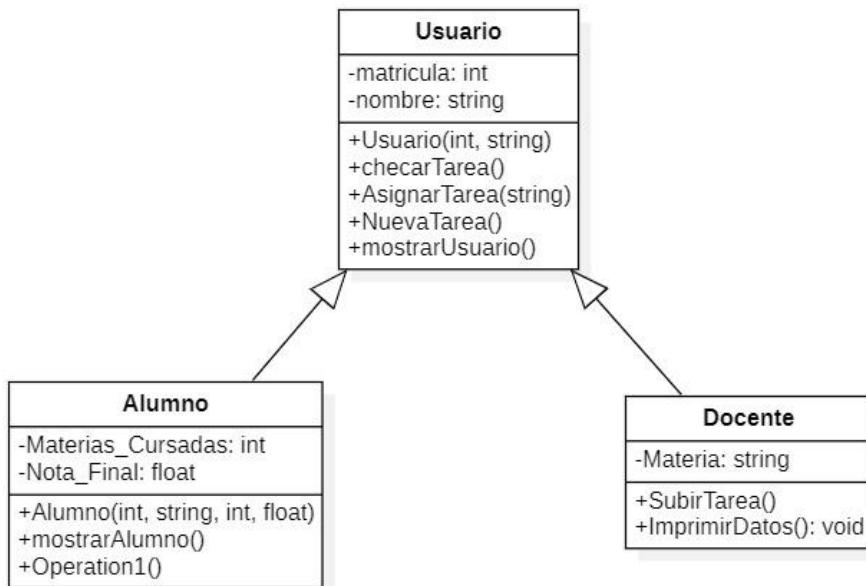
Factory Curso



Factory Tarea

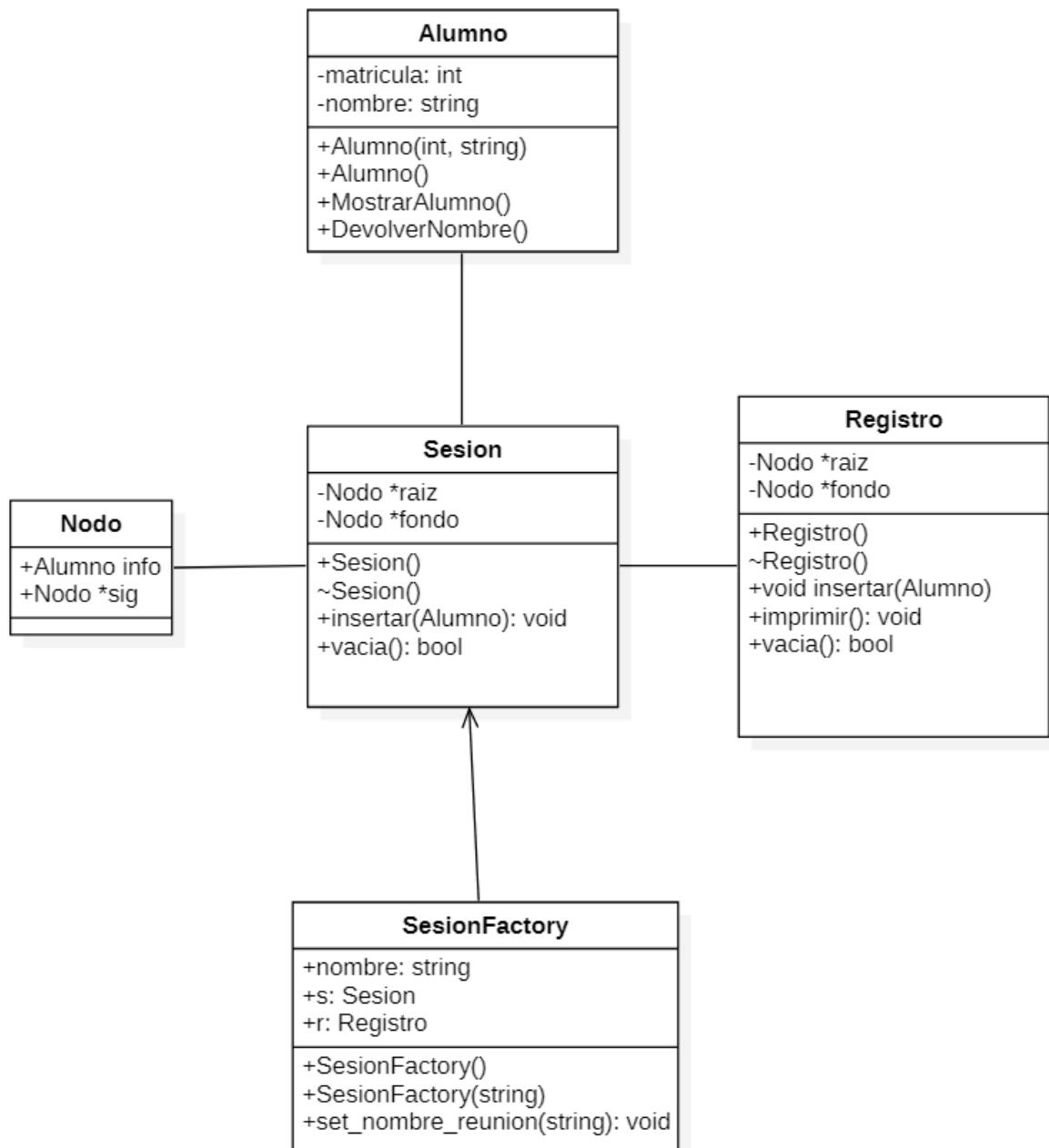


Factory de creación de usuarios

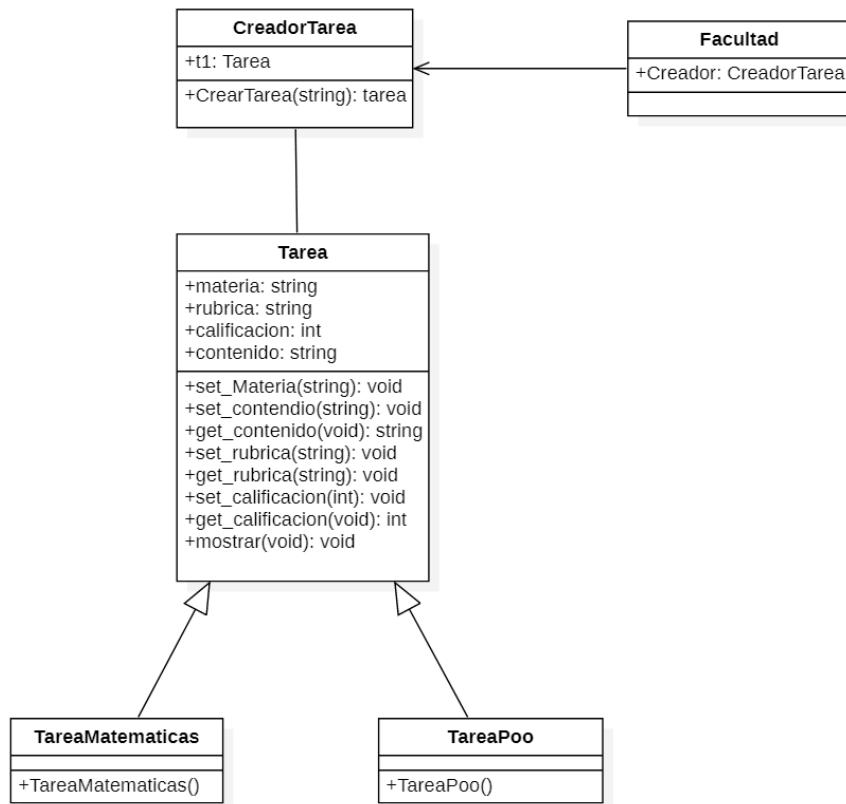


FACTORY METHOD

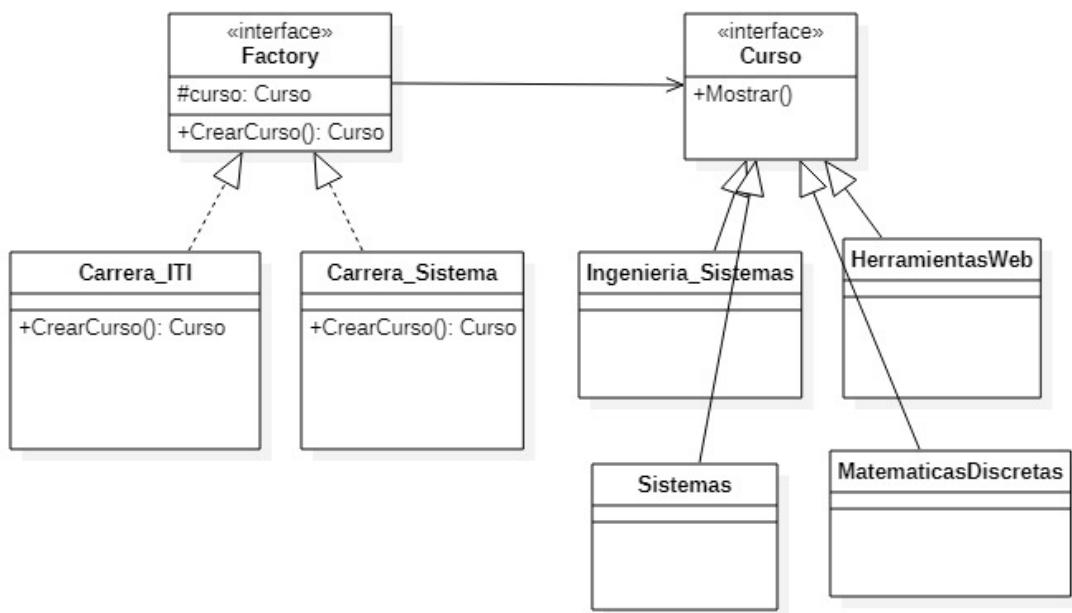
Método fabrica para toma de asistencia



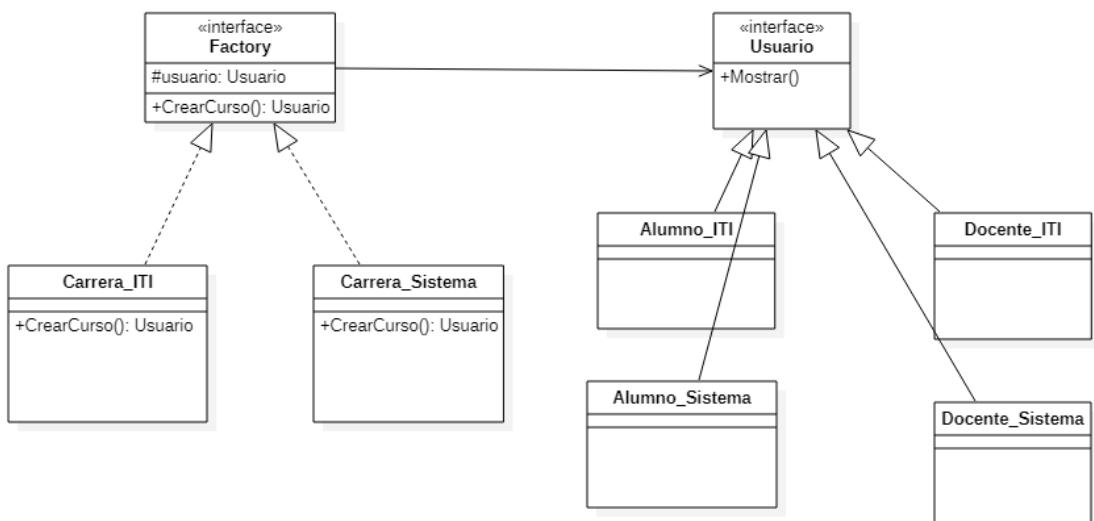
Método fabrica para tarea



Método fabrica Cursos

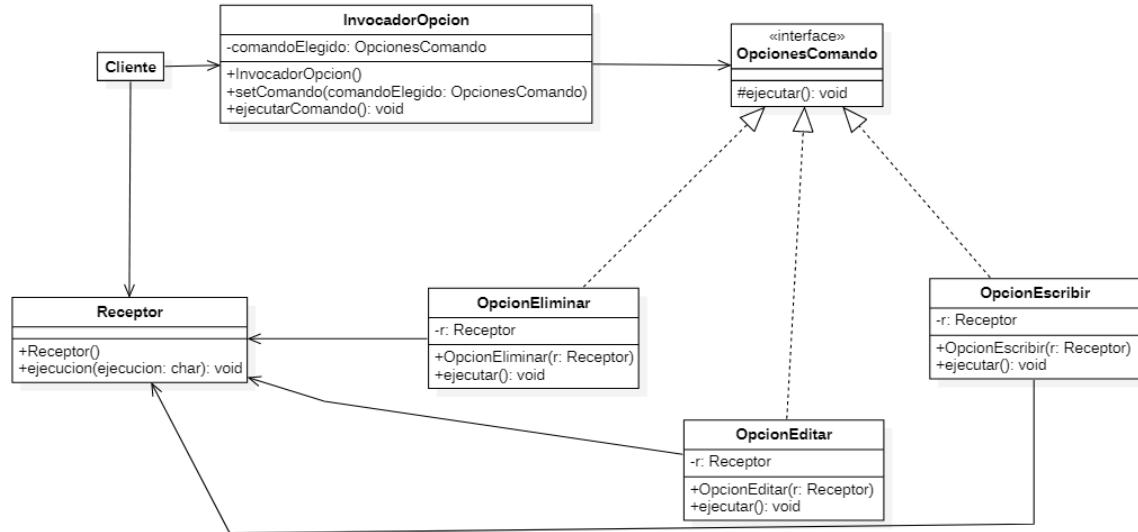


Método fabrica Usuarios

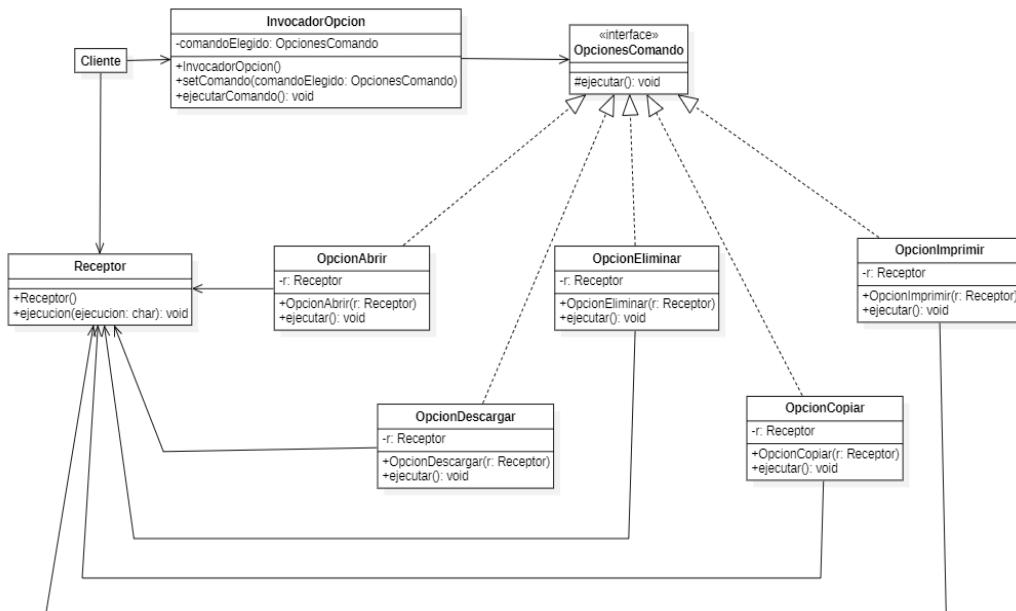


COMMAND

Método Command para Mensajes

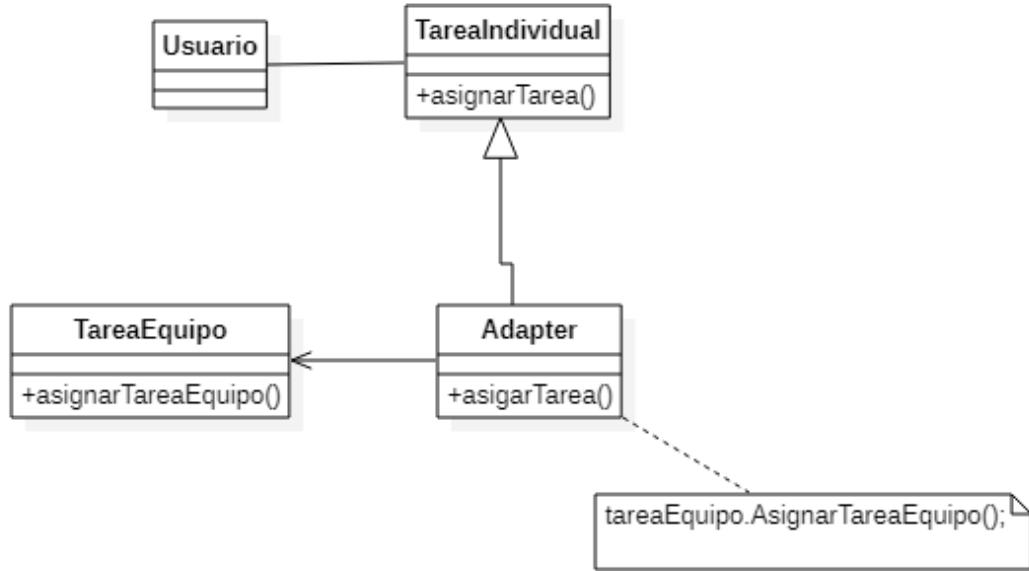


Método Command para Contenido

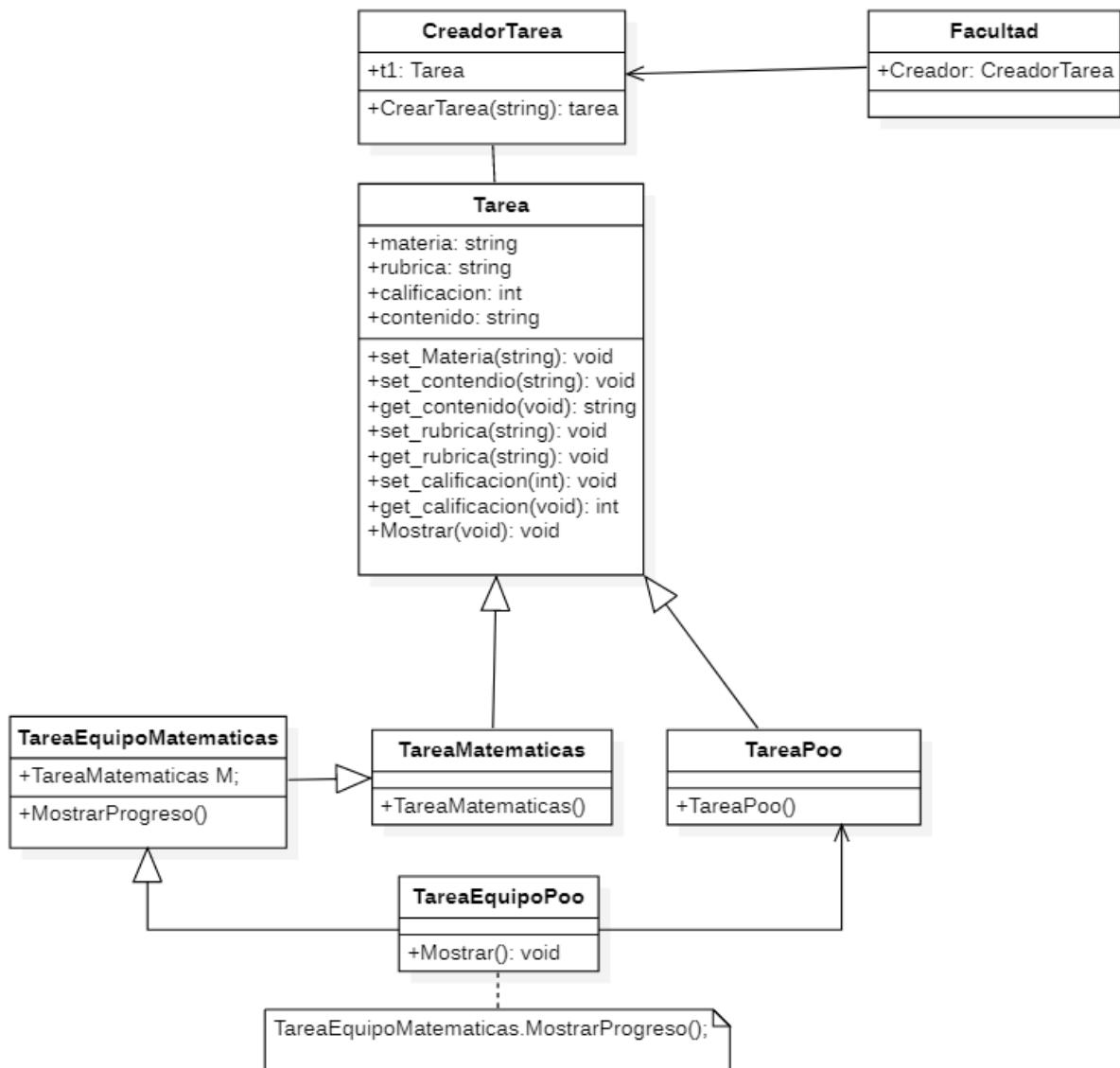


ADAPTER

Tareas en Equipo



Progreso de Actividad



ANEXO 4. LINK a repositorio en Github del proyecto

https://github.com/Vidal2207/Ingenieria_De_Software