INSTITUTO KHIPU

Semestre III

Sesión 22

PROGRAMA DE ESTUDIOS

DESARROLLO DE SISTEMAS DE INFORMACIÓN

ANÁLISIS Y DISEÑO DE SISTEMAS

Tema:

PUDS - INTEGRACIÓN DE DIAGRAMAS DE SECUENCIA EN EL PROCESO DE DESARROLLO.

PUDS - INTEGRACIÓN DE DIAGRAMAS DE SECUENCIA EN EL PROCESO DE DESARROLLO.

Diagrama de secuencia UML

Un diagrama de secuencia UML es una representación gráfica que muestra la interacción de objetos en un sistema a lo largo del tiempo. Estos diagramas capturan la secuencia de mensajes intercambiados entre objetos y el orden en que ocurren estas interacciones, presentándolos como líneas de vida verticales y flechas horizontales. Esencialmente, los diagramas de secuencia ofrecen una visualización cronológica de las interacciones de objetos, brindando perspectivas sobre el comportamiento dinámico y flujo de un sistema.

¿Por qué es esencial esta visualización? En el ámbito del diseño de software y sistemas, la claridad, colaboración y comprensibilidad son primordiales. Crear un diagrama de secuencia UML puede actuar como un manual visual, garantizando que todos los interesados tengan un entendimiento claro y compartido del comportamiento del sistema. Esto no solo ayuda en el desarrollo preciso del sistema sino que también agiliza la solución de problemas, modificación y esfuerzos colaborativos.

¿Qué es un diagrama de secuencia?

Los diagramas de secuencia siguen la notación UML estandarizada para representar las interacciones en un comportamiento de sistema específico. Pueden usarse para mejorar la comprensión de interacciones de software complejas, potenciar la eficiencia del diseño y mejorar la comunicación dentro de un proyecto.

Hay dos tipos de diagramas de secuencia:

1. Diagramas UML

En el contexto de UML, los diagramas de secuencia proporcionan una línea de tiempo visual de eventos en un sistema de software.

Son excelentes para:

- Mapear el flujo de control entre objetos del sistema,
- Mostrar cómo interactúan los objetos a lo largo del tiempo,
- Ayudar a diseñar, comprender y solucionar problemas en la arquitectura del sistema, llevando al desarrollo de software eficiente.

2. Diagramas basados en código

Estos diagramas se generan a partir del código de software y son útiles para lo siguiente:

- El código fuente genera el diagrama, ahorrando tiempo y reduciendo errores,
- Simplificar la comprensión y depuración del código al visualizar la ejecución del programa,
- Documentar sistemas de software para apoyar a nuevos miembros del equipo.

Conocer estos dos tipos de diagramas de secuencia te ayuda a elegir el que mejor se adapte a los requisitos de tu proyecto.

Cuándo usar un diagrama de secuencia

Los diagramas de secuencia UML son especialmente útiles para visualizar y comprender las interacciones entre diferentes partes de un sistema a lo largo del tiempo. Aquí te presentamos cuándo podrías considerar usar un diagrama de secuencia UML:

Entender el comportamiento del sistema

Si deseas capturar el comportamiento de un sistema, especialmente las interacciones entre objetos y componentes, un diagrama de secuencia puede representar visualmente estas interacciones a lo largo del tiempo.

Modelar casos de uso

Cuando tengas un escenario de caso de uso específico y quieras visualizar cómo interactúan los diferentes componentes del sistema durante ese escenario, un diagrama de secuencia es una elección adecuada. Proporciona un desglose paso a paso de los mensajes intercambiados entre objetos durante la ejecución del caso de uso.

Diseñar nuevos sistemas o características

Antes de que comience la codificación, los diagramas de secuencia pueden proporcionar una guía clara y visual para que los desarrolladores comprendan cómo las diferentes partes del sistema se comunicarán. Esto puede ayudar a identificar posibles problemas o oportunidades de optimización en la fase de diseño.

Documentar sistemas existentes

Para sistemas complejos donde las interacciones pueden no ser inmediatamente obvias, los diagramas de secuencia pueden servir como documentación para

ayudar a los nuevos miembros del equipo a comprender cómo interactúan las partes del sistema.

Detectar y abordar problemas

Al visualizar la secuencia de operaciones, se pueden identificar posibles cuellos de botella, interacciones innecesarias o secuencias problemáticas. Los diagramas de secuencia pueden ser fundamentales en los esfuerzos de optimización y depuración.

Representar procesos concurrentes

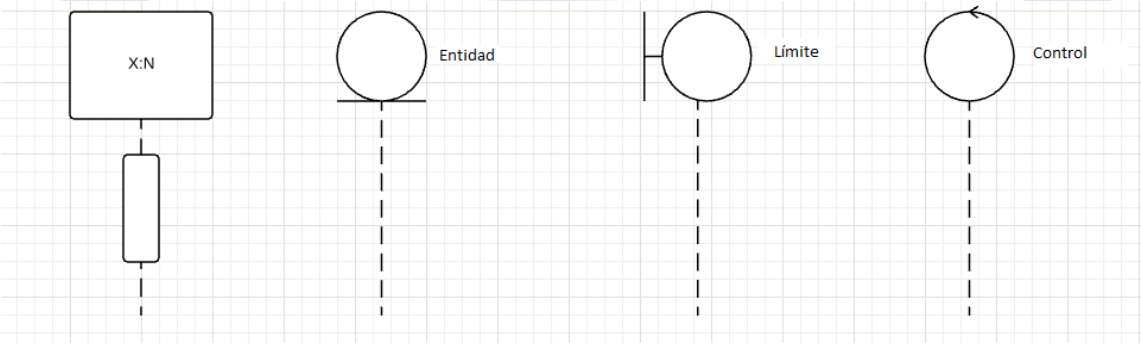
Si un sistema tiene operaciones concurrentes que necesitan ser visualizadas simultáneamente, los diagramas de secuencia pueden capturar estas interacciones con líneas de vida paralelas y activaciones.

Mostrar la creación y destrucción de objetos

Para sistemas donde el ciclo de vida de los objetos es crucial, los diagramas de secuencia pueden capturar claramente los momentos de creación y destrucción de objetos.

Aunque los diagramas de secuencia son poderosos, también es importante reconocer cuándo no usarlos. Si buscas representar relaciones estáticas entre clases o la estructura del sistema, otros diagramas UML como diagramas de clases o diagramas de componentes podrían ser más apropiados. Siempre elige el tipo de diagrama basado en el aspecto específico del sistema que deseas visualizar o comunicar.

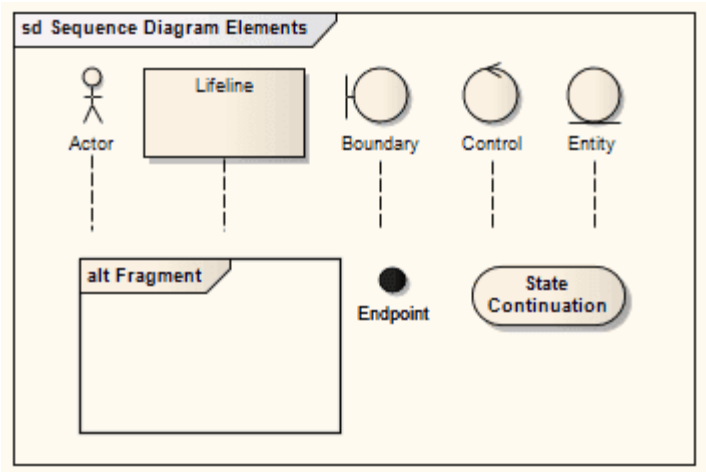
Componentes clave de los diagramas de secuencia



Los diagramas de secuencia UML incluyen varios elementos esenciales.

Repasemos estos componentes uno por uno:

Líneas de vida



Las líneas de vida simbolizan objetos o componentes en un sistema.

Se ven como líneas verticales discontinuas con el nombre del objeto o componente en la parte superior. A medida que desciendes por la línea de vida, muestra el paso del tiempo.

En situaciones que involucran interacciones humanas, las líneas de vida pueden representar actores, como usuarios. Por ejemplo, un sistema de tienda en línea puede tener una línea de vida para el cliente, el carrito de compras y el procesador de pagos.

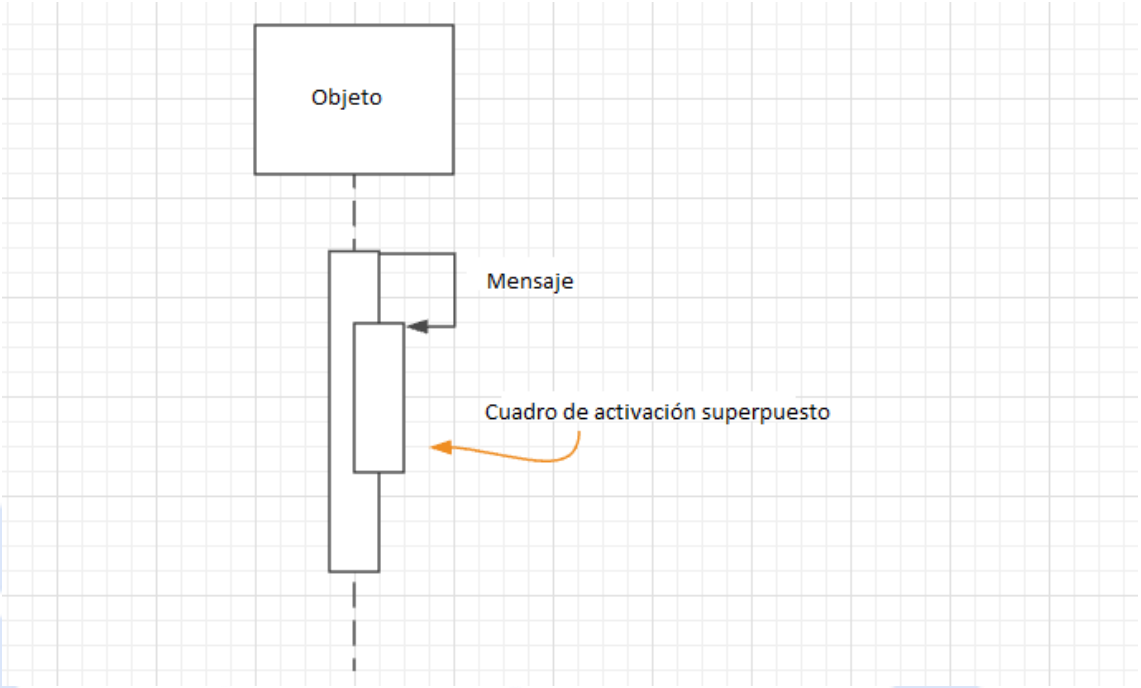
Mensajes

Estos marcan las interacciones entre las líneas de vida.

Se representan como flechas o líneas con diferentes notaciones. En una tienda en línea, un mensaje podría simbolizar a un cliente agregando un artículo al carrito de compras.

Hay diferentes tipos de mensajes:

- **Mensajes síncronos**: se muestran como flechas sólidas con puntas de flecha llenas. El remitente espera una respuesta antes de continuar.
- **Mensajes asíncronos**: presentados como flechas sólidas con puntas de flecha abiertas. El remitente no espera una respuesta antes de avanzar.
- **Mensajes de respuesta**: Son flechas discontinuas con puntas de flecha abiertas, indicando el retorno de información de un receptor al remitente.



- *Barras de activación*

Estas muestran la duración de la actividad o tiempo de procesamiento de un objeto.

En un sistema de tienda en línea, una barra de activación para el procesador de pagos podría representar el tiempo que lleva procesar un pago.

Las barras de activación son rectángulos delgados sobre la línea de vida de un objeto, comenzando cuando el objeto se activa y terminando cuando ya no está activo.

Fragmentos

Estos encapsulan comportamientos complicados o condicionales como bucles o alternativas.

Por ejemplo, en una tienda en línea, un fragmento de bucle podría simbolizar a un cliente agregando repetidamente artículos a su carrito de compras hasta que esté listo para pagar.

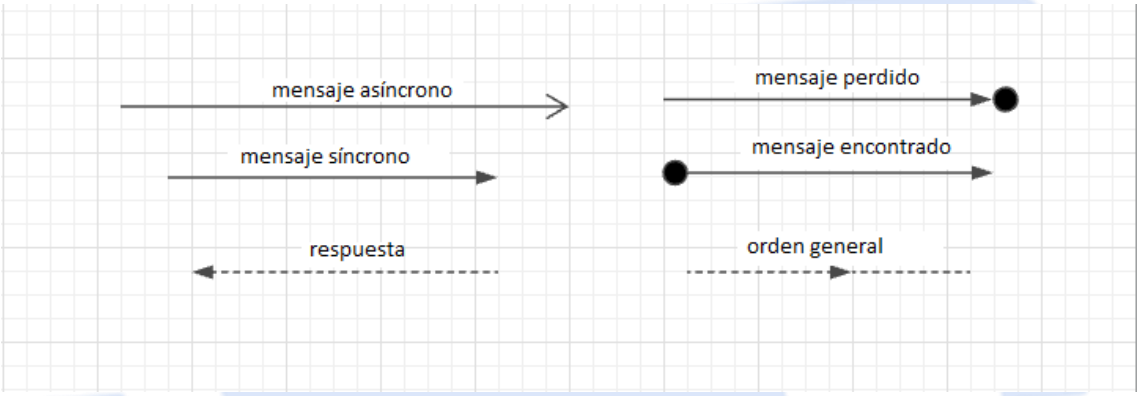
Los fragmentos son cajas con una etiqueta en la esquina superior izquierda que muestra el tipo de fragmento:

- **Alternativa (alt)**: Representa un comportamiento condicional donde solo una de varias interacciones ocurrirá basándose en condiciones específicas.

- **Opción (opt)**: Simboliza una interacción opcional que ocurre solo si se cumple una cierta condición.
- **Bucle (loop)**: Muestra una secuencia repetitiva de interacciones que ocurrirá mientras una condición dada sea verdadera.
- **Paralelo (par)**: Se utiliza para interacciones concurrentes que pueden ocurrir simultáneamente dentro del sistema.

Entender estos componentes te permite visualizar, comprender y comunicar interacciones complejas en sistemas de software.

Notaciones y símbolos en los diagramas de secuencia UML



Los diagramas de secuencia UML emplean diversas notaciones y símbolos para transmitir componentes e interacciones.

Conocer estas notaciones te ayuda a crear, entender y compartir interacciones complejas.

Líneas discontinuas verticales

Estas representan líneas de vida, ilustrando la duración de un objeto o componente en un sistema. El nombre del objeto o componente se encuentra en la parte superior de estas líneas.

Flechas sólidas con puntas de flecha llenas

Estas muestran mensajes síncronos. Cuando ves una flecha sólida con una punta de flecha llena, indica que el remitente envía un mensaje y luego espera una respuesta antes de continuar.

Flechas sólidas con puntas de flecha abiertas

Estas flechas representan mensajes asíncronos. Esto significa que el remitente envía un mensaje y no espera una respuesta antes de continuar.

Flechas discontinuas con puntas de flecha abiertas

Esta notación representa mensajes de respuesta. Estas flechas ilustran la información que regresa del receptor al remitente.

Rectángulos delgados superpuestos en una línea de vida

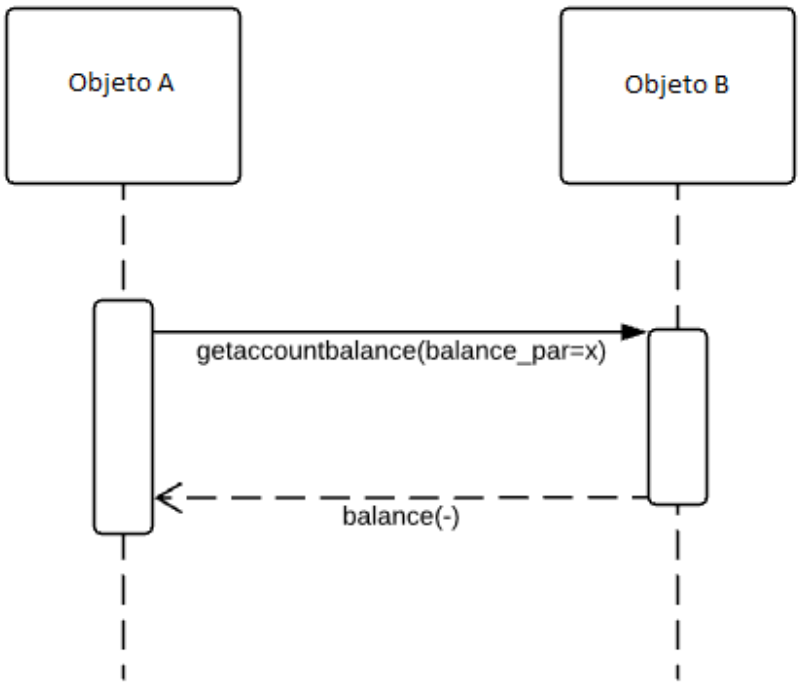
Estos rectángulos representan barras de activación. Indican la duración en que un objeto o actor está activo o interactuando.

Caja con una etiqueta en la esquina superior izquierda

Este símbolo se usa para denotar fragmentos. La etiqueta en la esquina superior izquierda indica el tipo de fragmento (alternativa, opción, bucle o paralelo), encapsulando comportamientos complejos o condicionales en tu sistema.

¿Cómo hacer un diagrama de secuencia UML?

Crear un diagrama de secuencia puede parecer desafiante, especialmente para sistemas complejos.



Sin embargo, con una planificación cuidadosa y comprensión del sistema, puedes crear un diagrama de secuencia claro y eficiente que ilustre las interacciones entre los componentes del sistema con el tiempo. Aquí hay consejos para hacer un diagrama de secuencia con éxito:

1. Identifica los actores y el sistema

Comienza delineando quién o qué interactúa con el sistema.

Esto puede ser un usuario, otro sistema o un dispositivo externo. Es importante recordar que los actores y el sistema con el que interactúan son los principales componentes que estás mostrando en tu diagrama.

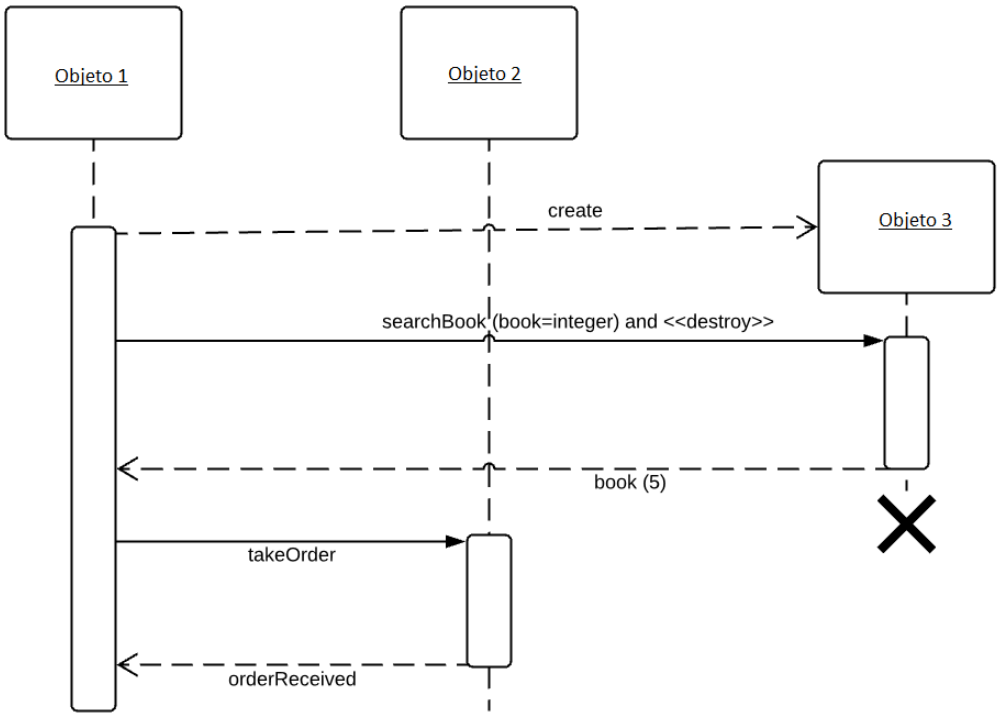
2. Define el alcance del diagrama de secuencia

Antes de comenzar a crear tu diagrama, debes definir su alcance.

¿Mostrarás un caso de uso individual, una característica particular o todo el proceso del sistema? Saber lo que estás tratando de representar guiará la creación de tu diagrama.

3. Identifica las líneas de vida

Las líneas de vida representan componentes individuales en tu sistema y sus interacciones con el tiempo.



Se representan como líneas verticales, con interacciones representadas como líneas horizontales o flechas. Identificar las líneas de vida involucradas en tu

proceso del sistema o caso de uso te ayuda a preparar el escenario para tu diagrama de secuencia.

4. Identifica e ilustra las interacciones

Las interacciones son el corazón de tu diagrama de secuencia.

Este paso implica decidir qué líneas de vida interactúan entre sí, cómo se comunican y la secuencia de estas interacciones. Comprender estas interacciones es crítico para representar tu proceso del sistema.

5. Añade barras de activación

Una barra de activación, representada como un rectángulo delgado en una línea de vida, muestra cuándo un objeto o actor está activo o tiene control. Agregar estas barras ayuda a aclarar la secuencia y el tiempo de las interacciones.

6. Dibuja mensajes

Usa flechas para representar interacciones entre objetos o actores. Asegúrate de que las flechas apunten del emisor al receptor y estén etiquetadas con el nombre del mensaje.

7. Usa fragmentos para comportamientos complejos

A veces, tu proceso del sistema o caso de uso podría incluir comportamientos complejos como bucles, condiciones o procesos paralelos.

Los fragmentos te permiten encapsular estos comportamientos en tu diagrama, proporcionando una representación clara de interacciones más complejas.

8. Revisa y refina tu diagrama de secuencia

Después de diseñar tu diagrama, tómate el tiempo para revisarlo y refinarlo. Asegúrate de que capture el proceso del sistema o caso de uso y que sea fácil de entender.

Entendiendo un diagrama de secuencia UML

Dominar los diagramas de secuencia UML requiere tanto conocimiento teórico como aplicación práctica.

La siguiente guía te ayudará a entender y aplicar estos diagramas.

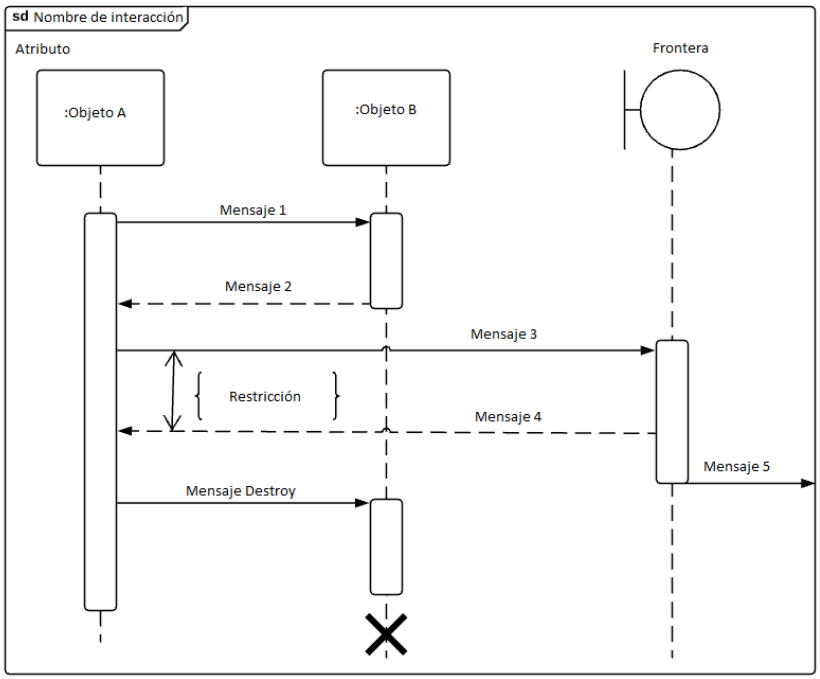
Cómo leer un diagrama de secuencia UML

1. Comienza desde la parte superior del diagrama y procede hacia abajo, siguiendo los mensajes intercambiados entre objetos o actores.
2. Presta especial atención a los tipos de mensajes ya que dan pistas sobre diferentes tipos de interacciones (como sincrónicas o asincrónicas).

3. Busca fragmentos para interpretar comportamientos condicionales o repetidos en la secuencia.
4. Consulta las barras de activación para determinar cuándo un objeto o actor está participando activamente en el proceso.

Mejores prácticas y técnicas avanzadas para diagramas de secuencia UML

Crear diagramas de secuencia UML claros y efectivos requiere una comprensión de las mejores prácticas y técnicas avanzadas.



Mejores prácticas

- **Simplicidad**: Mantén tus diagramas lo más simples posible. Enfócate en capturar interacciones esenciales y evitar detalles innecesarios que puedan llevar a la confusión.
- **Convenciones de nomenclatura**: Usa nombres consistentes y significativos para objetos y mensajes para que los diagramas sean fáciles de entender.
- **Color y formato**: Aplica diferentes colores y estilos para distinguir entre varios componentes de tu diagrama, lo que puede mejorar la legibilidad.
- **Modularización**: Si tienes escenarios complejos, divídelos en diagramas más pequeños y relacionados. Esta división puede facilitar su comprensión.
- **Anotaciones**: Utiliza comentarios y notas para proporcionar contexto adicional o explicar interacciones complejas.

FUENTE:

- https://www.uml.org/
- https://www.codingdojo.la/2023/06/16/guia-del-ciclo-de-vida-del-desarrollo-de-software/
- https://aws.amazon.com/es/what-is/sdlc/
- BURCH, John; GRUDNISKY, Gary. "Diseño de Sistemas de Información", Grupo Noriega editores.
- SENN, James A. "Análisis y diseño de sistemas de información", 2da. ed., McGraw-Hill.

INSTITUTO

KHIPU