

M6 Ingeniería de software FIN A

Proyecto modular

|  |  |
| --- | --- |
| **Tutor:** | **Claudia Hernandez Flores** |
| **Estudiante:** | **José Ramón Ibáñez Posadas** |
| **Matricula:** | **BNL098377** |

|  |  |
| --- | --- |
| Monterrey, Nuevo León | viernes, 27 de Junio de 2025 |

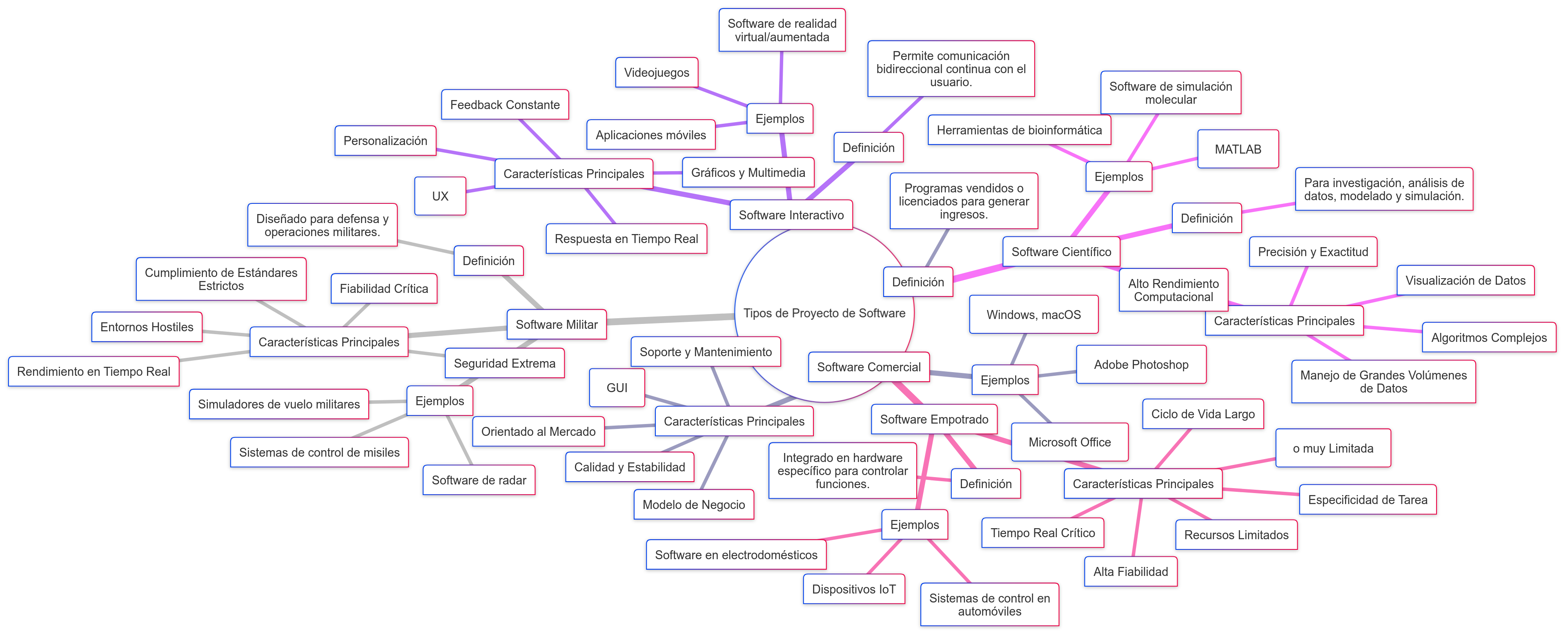
Introducción

En el panorama actual del desarrollo de software, la creación de sistemas que realmente satisfagan las necesidades de los usuarios y las organizaciones es un desafío constante. Históricamente, muchos proyectos de software han enfrentado dificultades, o incluso el fracaso, debido a una comprensión inadecuada o a una definición deficiente de los requerimientos iniciales. Este antecedente subraya la importancia crítica de la ingeniería de requerimientos, una disciplina fundamental que actúa como puente entre lo que el cliente desea y lo que el equipo de desarrollo construye.

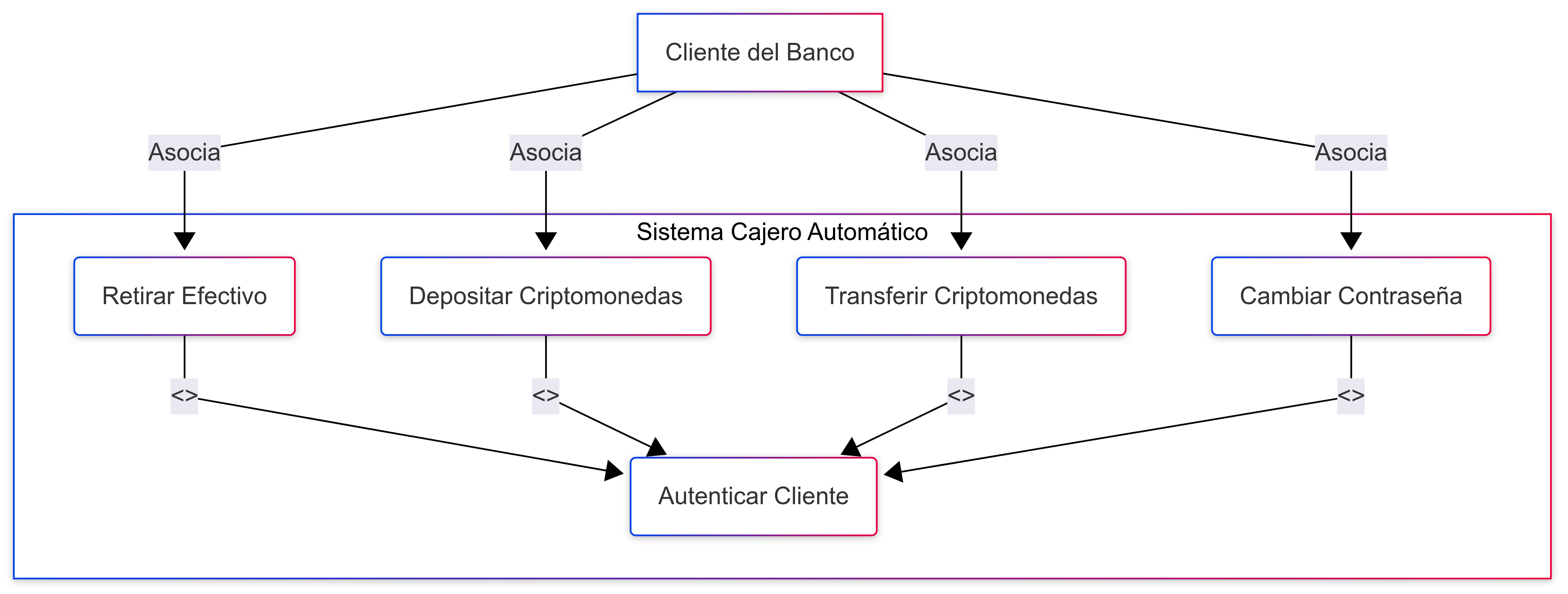
En la actualidad, la ingeniería de requerimientos es más relevante que nunca. Con la creciente complejidad de los sistemas y la diversidad de aplicaciones (desde software comercial y militar hasta sistemas interactivos y empotrados), la capacidad de identificar, analizar, especificar y validar los requerimientos de manera efectiva es indispensable. Un proceso de ingeniería de requerimientos robusto no solo minimiza errores costosos en etapas avanzadas del desarrollo, sino que también asegura que el producto final sea pertinente, útil y aceptado por sus usuarios.

El objetivo de esta actividad es doble. Por un lado, busca profundizar en la clasificación de los tipos de proyectos de software existentes, comprendiendo las características y los impactos distintivos de cada categoría, lo cual nos permite apreciar la amplitud y especialización de este campo. Por otro lado, la finalidad es aplicar los principios de la ingeniería de requerimientos mediante la elaboración de diagramas de casos de uso. Estos diagramas son herramientas esenciales para modelar las interacciones entre los actores y el sistema, permitiéndonos visualizar y comprender de forma clara las funcionalidades que un software debe ofrecer. Al realizar estos ejercicios, reforzamos nuestra habilidad para traducir necesidades complejas en representaciones concisas y claras, sentando las bases para un desarrollo de software exitoso y alineado con las expectativas del cliente.

Mapa mental: Tipos de proyecto de software



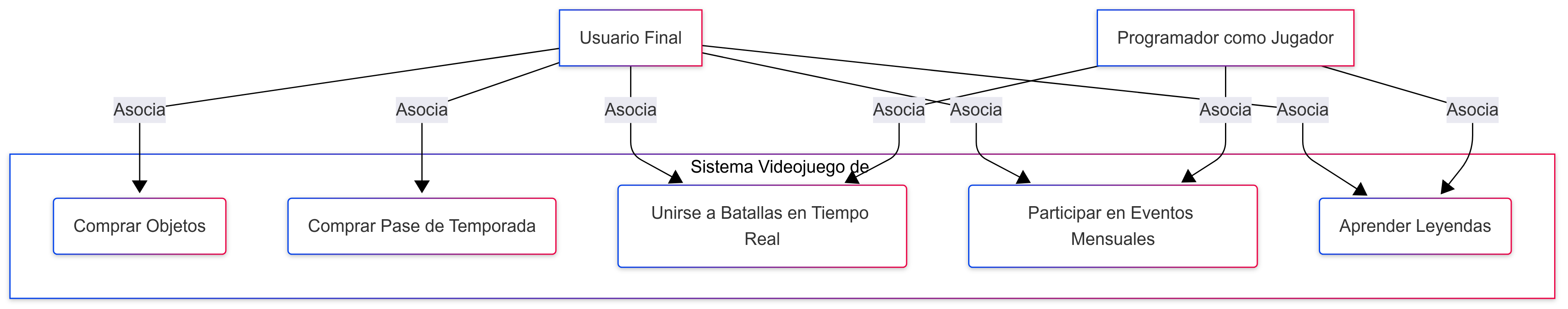
Ejercicio 1 – Caso de uso: Cliente banco



**Explicación del Diagrama:**

* **Actor (Cliente del Banco):** Representa a la persona que utiliza el cajero.
* **Sistema (Cajero Automático):** Delimita las funcionalidades del cajero.
* **Casos de Uso Principales:**
* **Retirar Efectivo:** Permite al cliente obtener dinero físico de su cuenta.
* **Depositar Criptomonedas:** Permite al cliente ingresar criptomonedas a su cuenta.
* **Transferir Criptomonedas:** Permite al cliente mover criptomonedas entre cuentas o a otros destinatarios.
* **Cambiar Contraseña:** Permite al cliente modificar su PIN o contraseña.
* **Caso de Uso Incluido (Autenticar Cliente):** Este caso de uso es esencial y se *incluye* en cada una de las operaciones principales (retiro, depósito, transferencia, cambio de contraseña). Esto significa que para realizar cualquiera de esas acciones, el cliente primero debe ser autenticado por el sistema (por ejemplo, insertando la tarjeta e ingresando el PIN).

Ejercicio 2 – Caso de uso: Programador de videojuego



**Explicación del Diagrama:**

* **Actores (Usuario Final, Programador como Jugador):** Representan a las personas que interactúan con el videojuego. El programador, en este contexto, es un tipo especial de usuario que también experimenta el juego.
* **Sistema (Videojuego de Aventuras):** Delimita las funcionalidades ofrecidas por el juego.
* **Casos de Uso Principales:**
  + **Comprar Objetos:** Permite a los jugadores adquirir ítems dentro del juego.
  + **Comprar Pase de Temporada:** Permite a los jugadores desbloquear contenido premium.
  + **Unirse a Batallas en Tiempo Real:** Permite a los jugadores participar en combates multijugador.
  + **Participar en Eventos Mensuales:** Permite a los jugadores tomar parte en actividades programadas.
  + **Aprender Leyendas:** La interacción pasiva donde el jugador descubre la historia y el trasfondo a medida que avanza.
* **Interacción de Actores:** Tanto el "Usuario Final" como el "Programador (como Jugador)" pueden realizar la mayoría de las interacciones. Se especificó que el programador desea "vivir la experiencia de aventura en el mapa, unirse a batallas en tiempo real y a los eventos", por lo que sus asociaciones se dirigen a esos casos de uso.

Diferencias entre ambos diagramas de caso de uso

La principal diferencia entre el diagrama del **Cajero Automático** (Ejercicio 1) y el del **Videojuego de Aventuras** (Ejercicio 2) radica en varios aspectos clave, que reflejan la naturaleza y los objetivos de cada tipo de software:

1. **Propósito y Naturaleza del Sistema:**
   * **Cajero Automático:** Es un sistema transaccional y de servicio. Su propósito principal es la **ejecución precisa y segura de operaciones financieras**. Los casos de uso son acciones bien definidas y controladas que el sistema debe realizar para el cliente. Se enfoca en la funcionalidad crítica del negocio.
   * **Videojuego de Aventuras:** Es un sis**Explicación del Diagrama:**
     + **Actores (Usuario Final, Programador como Jugador):** Representan a las personas que interactúan con el videojuego. El programador, en este contexto, es un tipo especial de usuario que también experimenta el juego.
     + **Sistema (Videojuego de Aventuras):** Delimita las funcionalidades ofrecidas por el juego.
     + **Casos de Uso Principales:**
       - **Comprar Objetos:** Permite a los jugadores adquirir ítems dentro del juego.
       - **Comprar Pase de Temporada:** Permite a los jugadores desbloquear contenido premium.
       - **Unirse a Batallas en Tiempo Real:** Permite a los jugadores participar en combates multijugador.
       - **Participar en Eventos Mensuales:** Permite a los jugadores tomar parte en actividades programadas.
       - **Aprender Leyendas:** La interacción pasiva donde el jugador descubre la historia y el trasfondo a medida que avanza.
     + **Interacción de Actores:** Tanto el "Usuario Final" como el "Programador (como Jugador)" pueden realizar la mayoría de las interacciones. Se especificó que el programador desea "vivir la experiencia de aventura en el mapa, unirse a batallas en tiempo real y a los eventos", por lo que sus asociaciones se dirigen a esos casos de uso.
2. **Complejidad en la Interacción y Requerimientos Implícitos:**
   * **Cajero Automático:** La interacción es lineal y predecible. Los requerimientos son principalmente funcionales y se centran en la seguridad, la exactitud de los datos y la fiabilidad. La relación <<include>> con "Autenticar Cliente" resalta un requisito de seguridad transversal y obligatorio para todas las transacciones.
   * **Videojuego de Aventuras:** La interacción es más dinámica y abierta. Los requerimientos incluyen funcionalidades (comprar, unirse a batallas) pero también aspectos más subjetivos y de experiencia (aprender leyendas, disfrutar de la aventura). El ejercicio menciona las posibles complicaciones del programador con batallas y eventos en tiempo real, lo que subraya la importancia de la **viabilidad y la negociación de requerimientos**, un concepto clave en la ingeniería de requerimientos (como se menciona en el Tema 10 sobre la validación de requerimientos).
3. **Roles de los Actores:**
   * **Cajero Automático:** Solo hay un tipo de actor externo principal, el "Cliente del Banco", cuyo rol es claro y está bien delimitado por las operaciones del cajero.
   * **Videojuego de Aventuras:** Se introducen dos actores principales: "Usuario Final" y "Programador (como Jugador)". La presencia del "Programador (como Jugador)" es interesante porque el desarrollador se convierte en un usuario más, lo que puede influir en la forma en que se perciben y priorizan los requisitos, dado que él mismo experimentará el juego. Esto resalta la importancia de identificar a todos los "stakeholders" o partes interesadas.

Conclusión

A lo largo de este Proyecto Modular, he podido consolidar una comprensión profunda sobre la vital importancia de la ingeniería de requerimientos en el ciclo de vida del desarrollo de software. Queda claro que una definición precisa y una gestión adecuada de los requisitos desde las etapas iniciales son el cimiento para evitar los desafíos y posibles fracasos que surgen cuando no se escucha o se malinterpreta la necesidad del cliente. Este proceso no solo se trata de documentar lo que el sistema debe hacer, sino de asegurar una alineación constante con las expectativas del usuario final y los objetivos del proyecto.

Además, la exploración de los diversos "tipos de proyectos de software" —desde el software comercial que busca amplios mercados, hasta el militar con sus exigencias de seguridad y fiabilidad extrema, pasando por el interactivo que prioriza la experiencia del usuario, el científico que maneja complejidades computacionales, y el empotrado que opera con recursos limitados— ha puesto de manifiesto la vasta y especializada naturaleza de la industria. Cada tipo presenta retos y prioridades únicas, lo que subraya la necesidad de adaptar las metodologías de ingeniería de software a las particularidades de cada proyecto.

Finalmente, la aplicación práctica a través de la elaboración de diagramas de casos de uso ha sido fundamental para traducir los requisitos textuales en representaciones visuales claras. Al identificar actores y sus interacciones con el sistema, como en los ejemplos del cajero automático y el videojuego, se logra una visión estructurada de las funcionalidades esperadas. La comparación entre ambos diagramas reveló cómo el propósito y la naturaleza del sistema impactan directamente en la forma de modelar sus interacciones y en la complejidad de sus requerimientos. En síntesis, este trabajo ha reforzado la idea de que una ingeniería de requerimientos sólida y la comprensión de los tipos de software son pilares para garantizar el éxito y la relevancia de cualquier desarrollo tecnológico.

**Bibliografía**

