# Identificación del problema y análisis de requerimientos

## Caso de Estudio : PokeCollector

| Cliente | Pokemon Company |
| --- | --- |
| Usuario | Designados por el cliente |
| Requerimientos funcionales | *RF1: Registrar Cartas con id único*  *RF2: Ordenar cartas por puntos de vida de mayor a menor* |
| Contexto del problema | *Pokemon Company necesita un sistema que permita gestionar su inventario de cartas “PokeCollector". Esta gestión permitirá al usuario registrar, consultar, actualizar y eliminar información sobre cada carta. El sistema prioriza la singularidad de cada carta mediante un ID único, ordena las cartas por puntos de vida y ofrece una interfaz sencilla para interactuar con los datos. Para ello se requiere que el usuario registre información sobre las cartas como, id, nombre de carta, puntos de vida, tipo y rareza.* |
| Requerimientos no funcionales | * Interfaz intuitiva y agradable * Modularidad para posibles expansiones futuras |
| Requerimientos de proceso | * Construcción en Java * Construcción mediante el paradigma de POO * Contener la documentación respectiva * Uso adecuado de la metodología CRUD. |

| Identificador y nombre | *RF1: Registrar Cartas con id único* | | | |
| --- | --- | --- | --- | --- |
| Resumen | *El sistema permite a partir de un id único, validarlo siempre que no se repita, e informar el estado de la validación al usuario. Luego de validar e id previamente ingresado, el usuario deberá ingresar los siguiente datos: nombre de carta, puntos de vida, tipo y rareza. Seguidamente, se mostrarán mensajes en pantalla por cada solicitud y validación, para luego almacenar estos datos en el sistema.* | | | |
| Entradas | **Nombre entrada** | **Tipo de dato** | | **Condición valores válidos** |
| id | String | | *-diferente a todos los demás* |
| nombre | String | | *-de uno de los pokemons existentes* |
| puntosVida | int | | -no contener letras ni caracteres |
| tipo | String | | -Debe ser una de las opciones válidas(Agua, Fuego, Viento etc…) |
| rareza | String | | -Debe ser una de las opciones válidas(Básico,Raro,Mítico o legendario) |
| Resultado o Postcondición | Construida la carta, se procede a almacenarla en memoria | | | |
| Salidas | **Nombre salida** | | **Tipo de dato** | **Formato** |
| Respuesta.true | | String | *“!registro exitoso¡”* |
| Respuesta.false | | String |  |

| Identificador y nombre | *RF2: Ordenar cartas por puntos de vida de mayor a menor* | | | |
| --- | --- | --- | --- | --- |
| Resumen | *El sistema permite a partir de los puntos de vida de cada carta, ordenarlas de mayor a menor para así mostrarlas al usuario, en una posterior consulta* | | | |
| Entradas | **Nombre entrada** | **Tipo de dato** | | **Condición valores válidos** |
| Resultado o Postcondición | las cartas del arreglo memoria quedan ordenadas por puntos de vidad de menor a mayor | | | |
| Salidas | **Nombre salida** | | **Tipo de dato** | **Formato** |
| construirListaDeCartas() | | String | *“|Id||Nombre||Puntos de vida||Tipo|*  ***…*** |