# DOCUMENTO DE ARQUITECTURA PROYECTO POKEMON

En este documento se presentan los diagramas que representan la arquitectura del software *InverterMatchPokemon*, el cual brinda solución a lo solicitado en el documento “*Pokemón Arena.docx”*. Para representar la arquitectura se utilizaron diferentes tipos de diagramas, indicados a continuación:

## *Diagramas de la arquitectura*

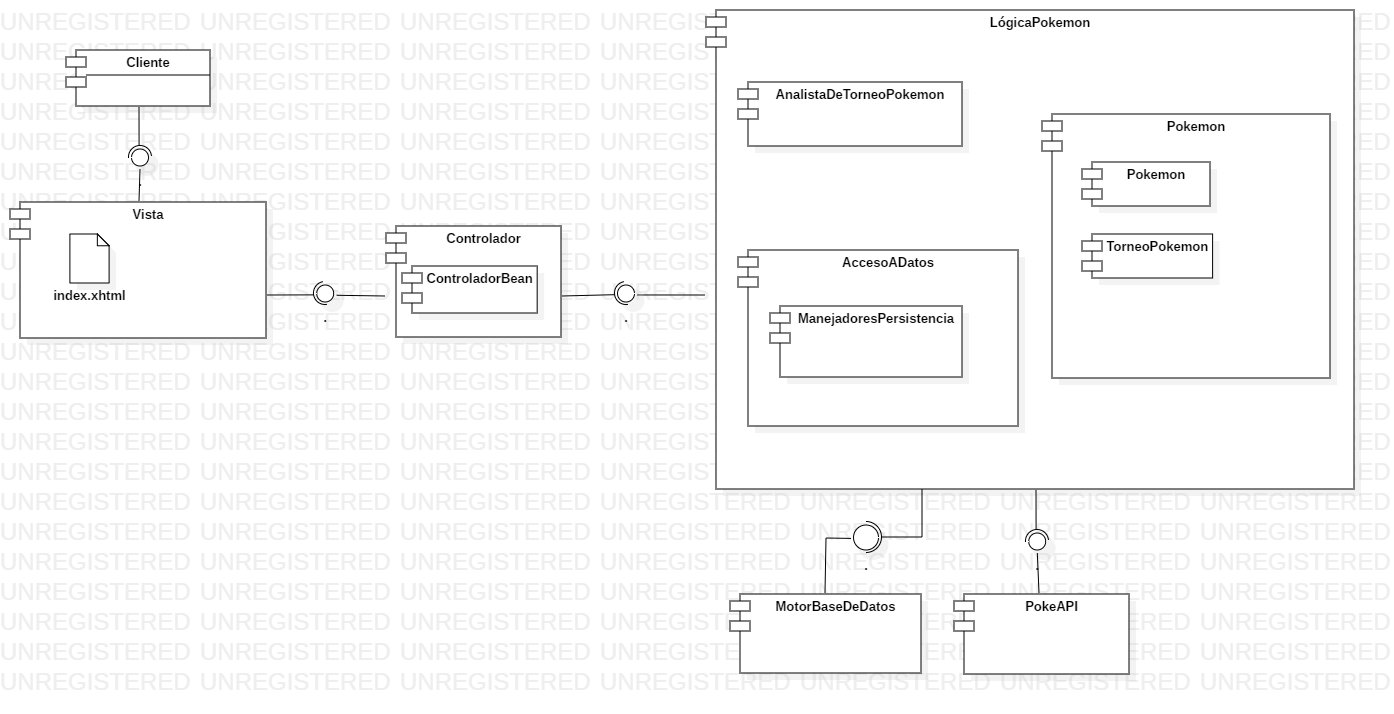
### *Diagrama de componentes*

Para dar solución al ejercicio se ha diseñado una arquitectura orientada a servicios, en la cual se establecieron diferentes componentes que realizan funciones específicas e interactúan entre ellos a través de interfaces.

El primer componente es el componente visual, a través del cual el usuario es capaz de interactuar con el resto del sistema. Este se comunica directamente con el controlador, el cual se encarga de invocar los servicios de la lógica dependiendo de la interacción del usuario. En este caso el componente controlador está compuesto por el “ControladorBean”, que es una clase encargada de recibir la información de la vista, transformarla en objetos java e invocar los servicios de “LógicaPokemon”.

El componente “LógicaPokemon” es el encargado de realizar todos los procesos lógicos de la aplicación como:

* Obtener la información de los pokemones a partir de la PokeApi
* Almacenar la información de los pokemones en la base de datos
* Calcular la cantidad mínima de pasos para llegar al resultado de un torneo.
* Indicar si un torneo fue saboteado
* Etc.



### *Diagrama de clases*

Con el fin de estructurar la aplicación en diferentes servicios se han agrupado las clases en los siguientes paquetes:

* Controlador
* Aanalistadetorneopokemon
* Pokemonclases
* Accesoadatos
  + Manejadorpersistencia

El paquete controlador contiene la clase ControladorBean, la cual se encarga de orquestar las funcionalidades a partir de las solicitudes del cliente.

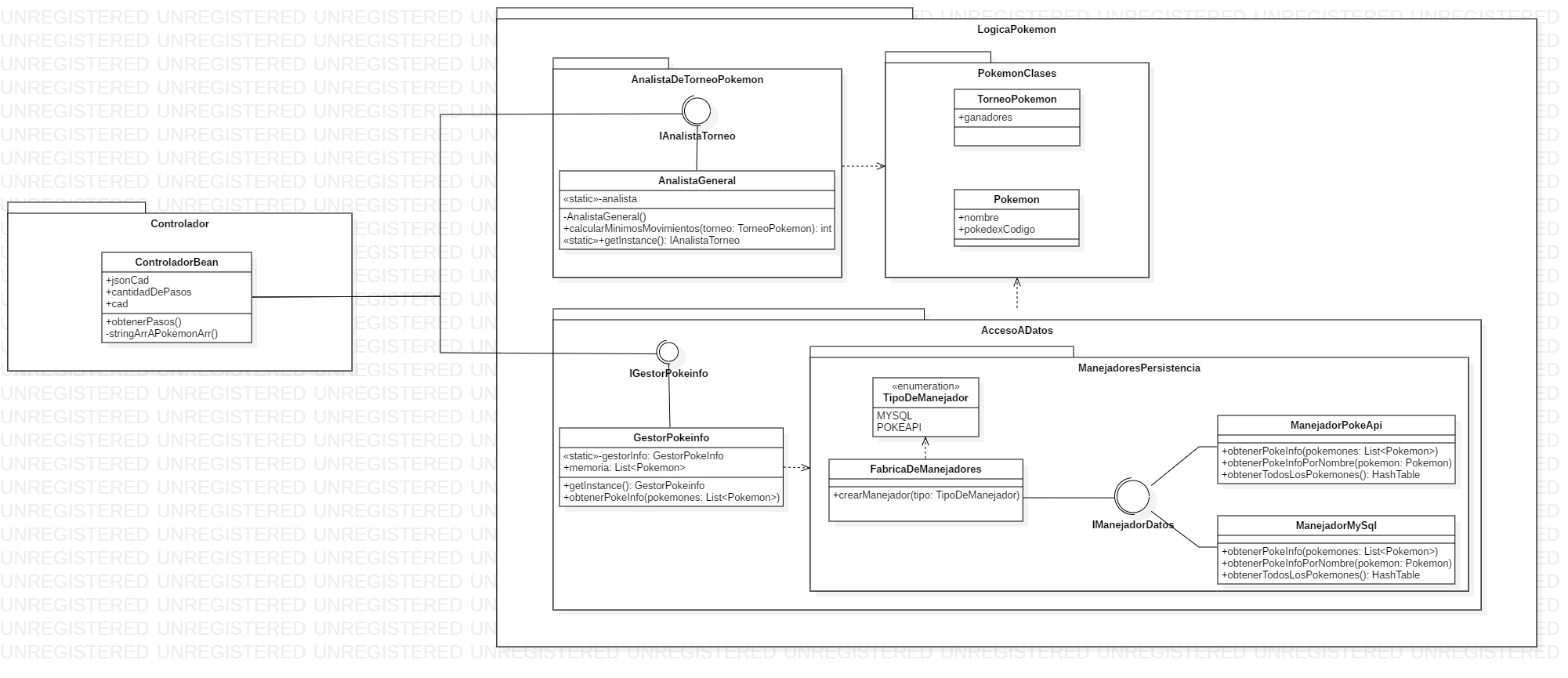
El paquete AnalistaDeTorneoPokemon contiene los servicios que indican:

* Cantidad mínima de movimientos.
* Si el torneo fue saboteado o no.

Para este paquete se ha definido la interfaz “IAnalistaTorneo” implementada por la clase “AnalistaGeneral” garantiza un único punto de acceso a los servicios de análisis del torneo gracias a la aplicación del patrón Singleton.

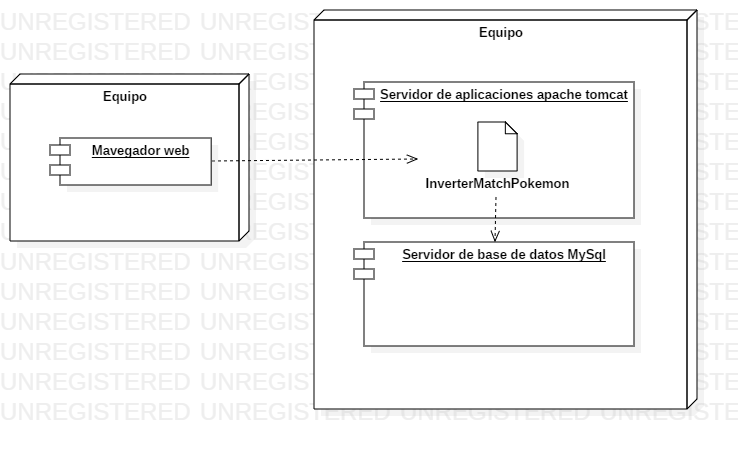
Para obtener la información de los pokemones se ha definido el paquete “AccesoADatos” el cual se puede acceder a través de la interfaz “IGestorPokeinfo”, implementada por la clase “GestorPokeinfo” la cual cuenta con una instancia única para toda la sesión establecida por el patrón de diseño Singleton. Esta clase obtiene la información de los pokemones de la base de datos y la almacena en una memoria cache para futuras consultas dentro de la misma sesión.

Para el acceso a la información persistente se ha implementado el patrón Factory, el cual permite establecer diferentes métodos de acceso a esta, brindándole flexibilidad al software en caso de que se desee utilizar un motor de persistencia distinto.

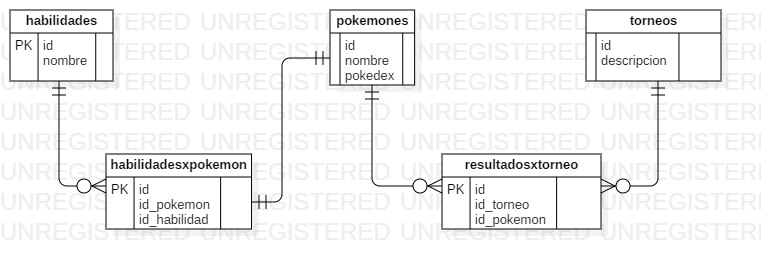


### *Diagrama de despliegue*

El despliegue representa el despliegue físico de la aplicación, para lo que se requiere un equipo que funcione como servidor, el cual contenga el servidor de aplicaciones apache tomcat y el motor de bases de datos MySql. El software estará contenido dentro del servidor apache tomcat y utilizará los servicios del servidor MySql, y a su vez será invocado a través de un navegador web instalado en el equipo del cliente.



### *Diagrama de base de datos*



## *Fuentes del proyecto*

La fuentes del proyecto se encuentran publicadas en el repositorio de github <https://github.com/etrivino/pokemon.git>