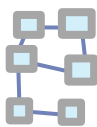
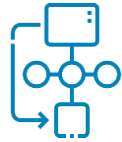




Tecnología de Programación



Modelando un videojuego – parte 2

Modelado de Road Fighter.

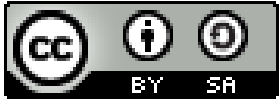
Diagramas de clases reducido.

Aplicando modelo de 3 capas.

Dr. Federico Joaquín 
federico.joaquin@cs.uns.edu.ar

Algunos derechos reservados

Diseño OO - Modelado de un videojuego - parte 2



© Octubre 2025 por Federico Joaquín

Esta obra está bajo una Licencia Creative Commons Atribución-CompartirIgual 4.0 Internacional

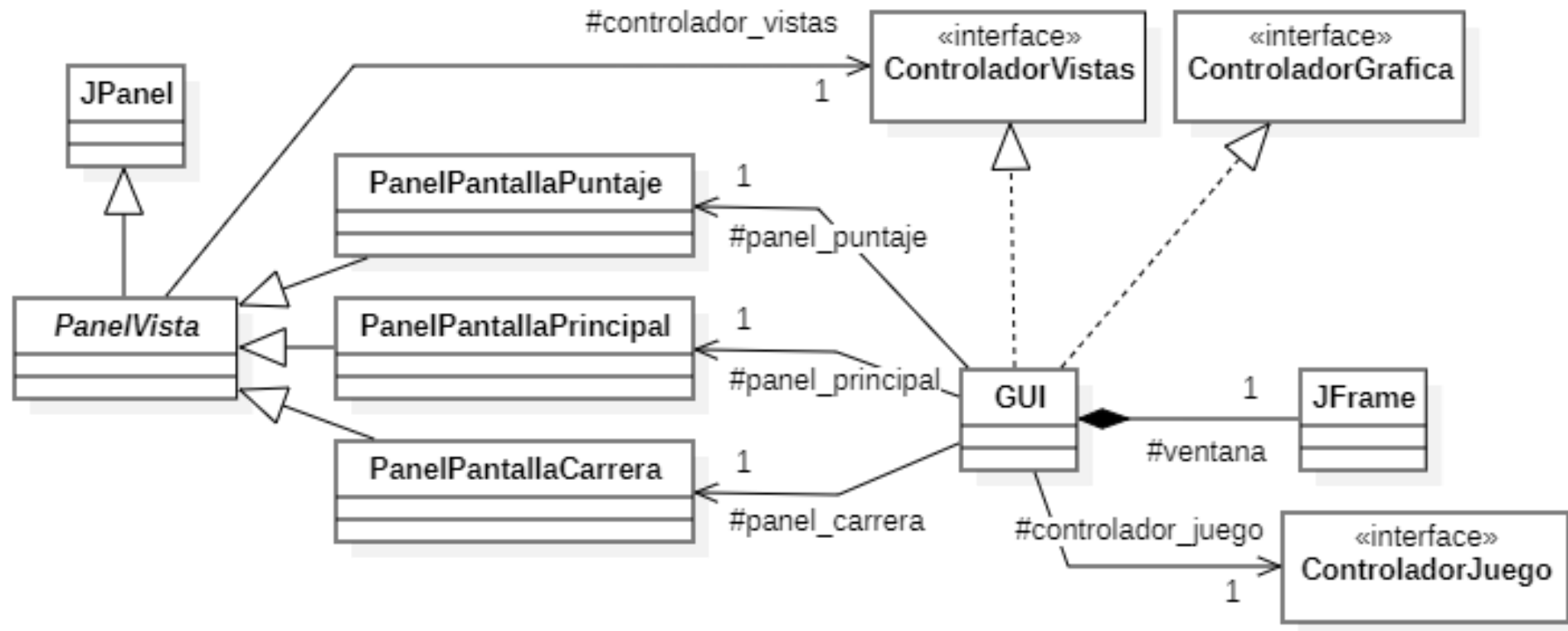
Continuemos modelando

REFACTORIZACIÓN DE DIAGRAMA REDUCIDO DE CLASES
APLICANDO MODELO DE 3 CAPAS
CASO DE ESTUDIO: ROAD FIGHTER

A screenshot from the game 'Road Runner' showing a top-down view of a road. A red car is in the center lane, a yellow car is in the left lane, and a green car is in the right lane. The road is flanked by green grass and trees. The score is 1P 0002 and the fuel gauge is full.

*Durante esta clase, se **extenderá** dicho modelo, **refactorizándolo**, para incorporar las nociones del **modelo de 3 capas**, de forma tal que se pueda **desarrollar una aplicación que permita operar el juego**.*

Capa de vista del modelo refactorizado



En este **fragmento** del modelo propuesto, se observan **algunas** de las **entidades** asociadas a la **capa de vista**.

Particularmente, estas clases permiten llevar a cabo la tarea de **visualizar** las diferentes **pantallas** de la aplicación.

La clase **GUI** manipula todo el **flujo de ejecución** entre ellas.

Esta imagen es un **fragmento del modelo**, que deliberadamente **invisibiliza** algunas de las relaciones modeladas, para centrarse en el aspecto más relevante de la slide actual de la presentación.

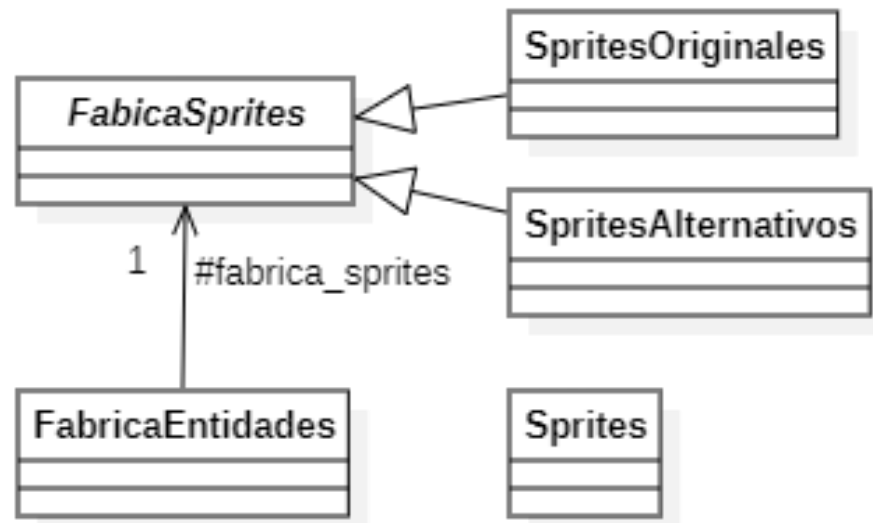


En este **fragmento** del modelo propuesto, se observan **algunas** de las **entidades** asociadas a la **capa de vista**.

Particularmente, estas clases permiten llevar a cabo la tarea de **creación** de las diferentes **entidades**, junto con los **Sprites** que las caracterizan en relación al **modo de juego seleccionado**.

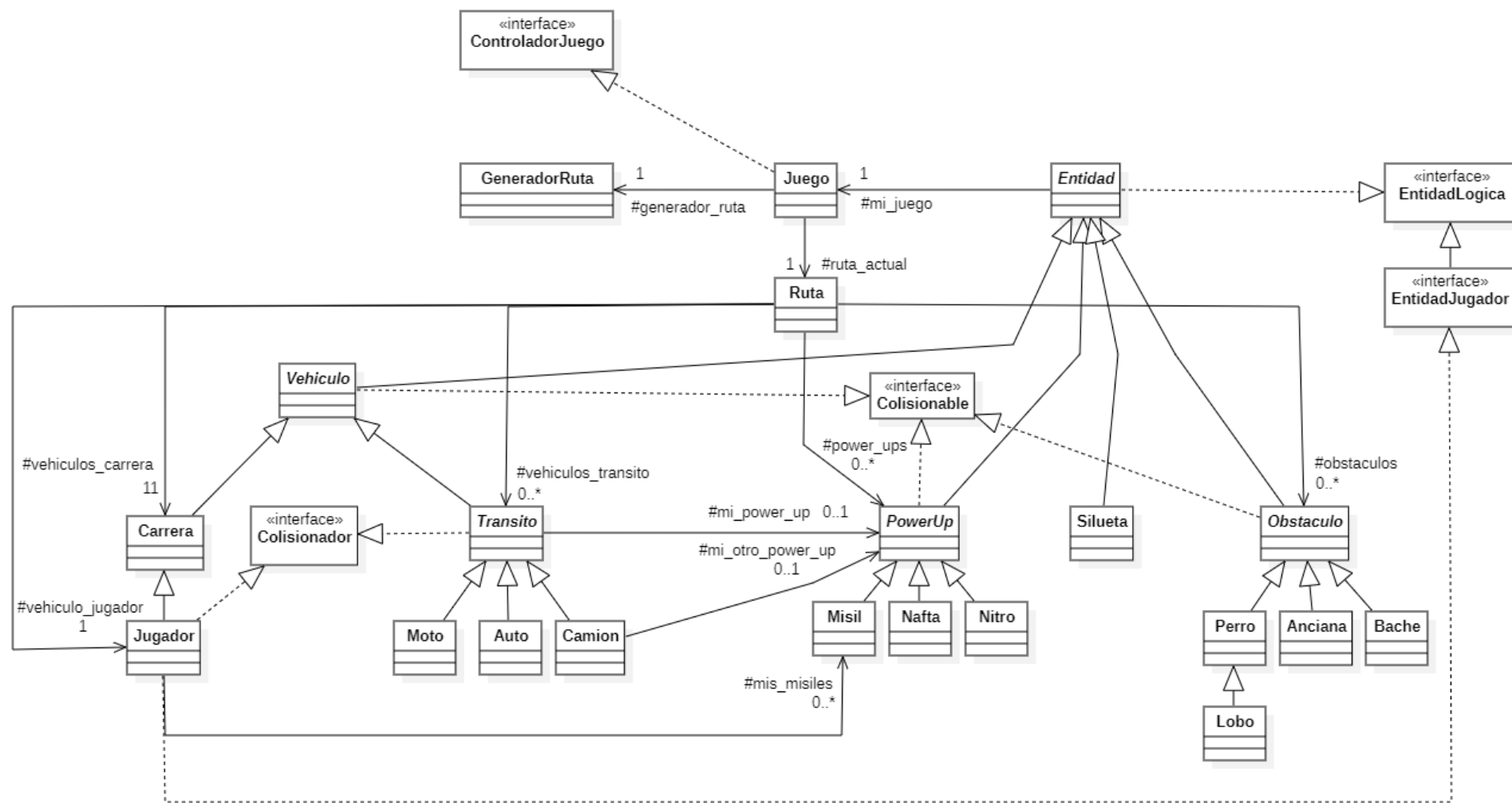
viernes, 3 de octubre de 2025

Capa de vista del modelo refactorizado



Esta imagen es un **fragmento del modelo**, que deliberadamente **invisibiliza** algunas de las relaciones modeladas, para centrarse en el aspecto más relevante de la slide actual de la presentación.

Capa de lógica del modelo refactorizado



Esta imagen es un **fragmento del modelo**, que deliberadamente **invisibiliza** algunas de las relaciones modeladas, para centrarse en el aspecto más relevante de slide actual de la presentación.



En este **fragmento** del modelo propuesto, se observan **algunas** de las **entidades** asociadas a la capa lógica.

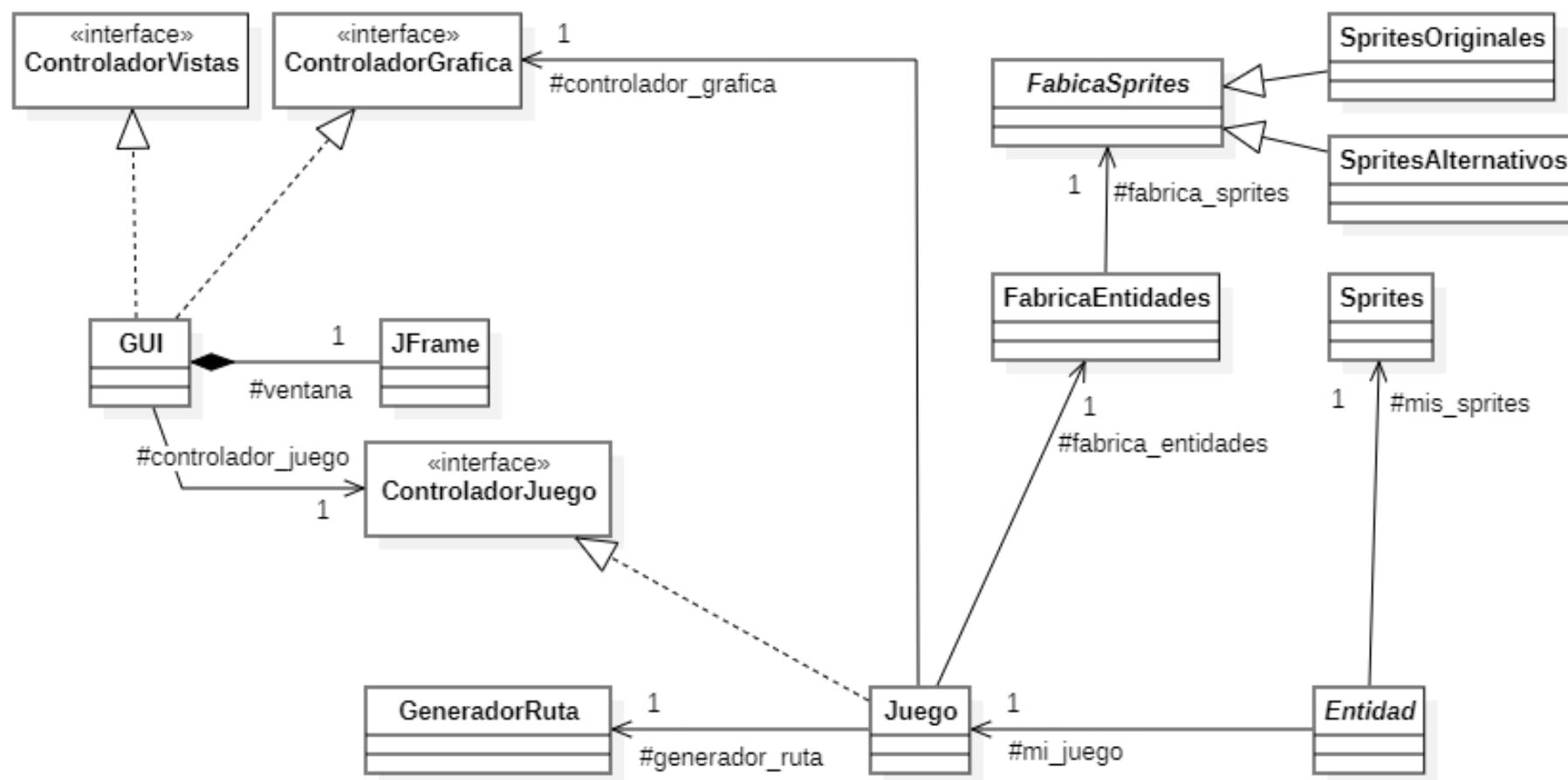
Se realizó alguna refactorización menor, contemplando la clase **Entidad. Juego** implementa **ControladorJuego**, la interfaz conocida desde la capa de vista. Además, conoce a **Ruta** y a **GeneradorRuta** que permite construir estas.



Puentes entre capas lógica y de vista

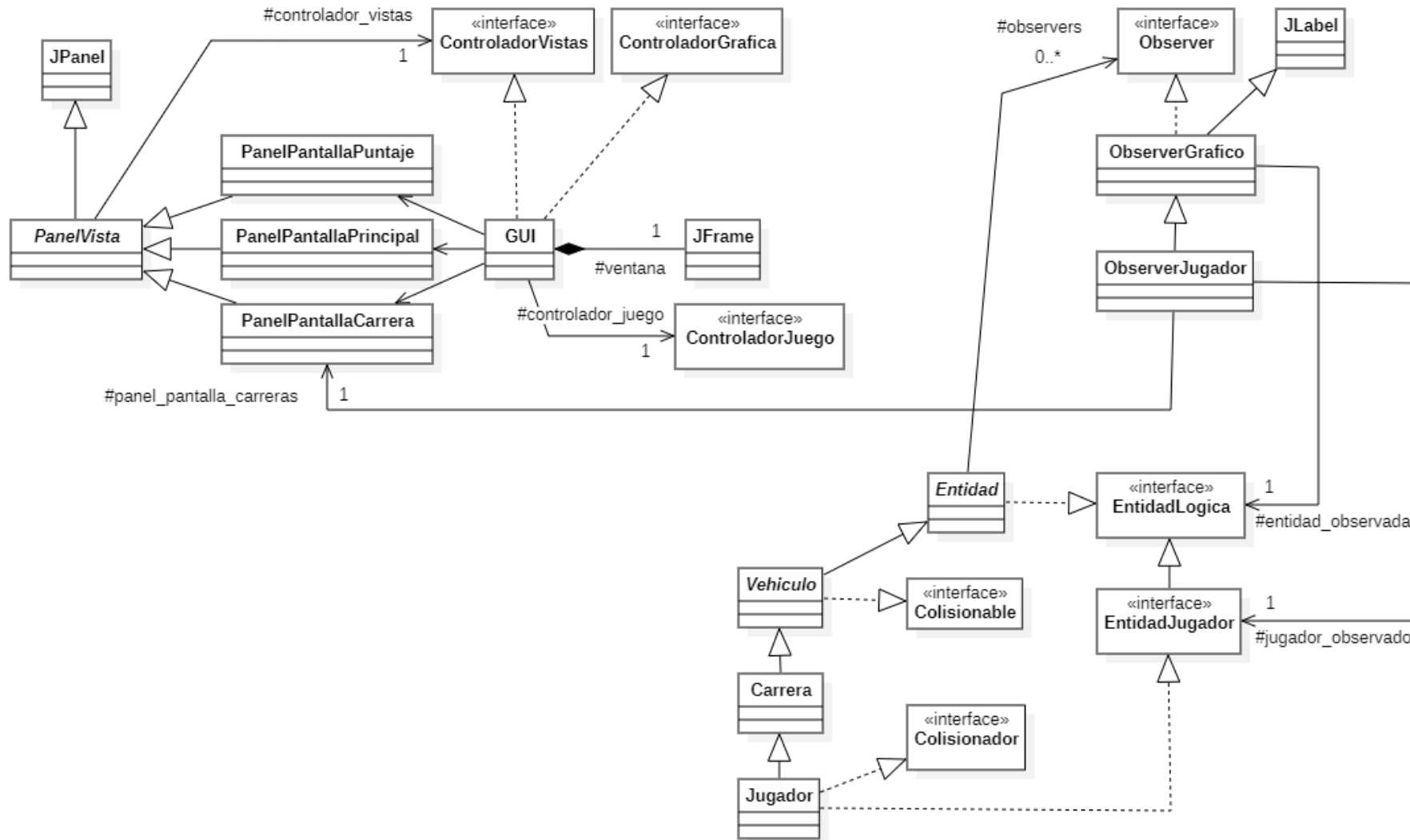
En este **fragmento** del modelo propuesto, se observan **algunos puentes** de interconectar diferentes **capas**.

Particularmente, existen **interfaces** para mediar los servicios a ofrecer entre la **vista** (clase GUI) y la **lógica** (clase Juego).



Esta imagen es un **fragmento del modelo**, que deliberadamente **invisibiliza** algunas de las relaciones modeladas, para centrarse en el aspecto más relevante de la slide actual de la presentación.

Puentes entre capa de vista y capa lógica

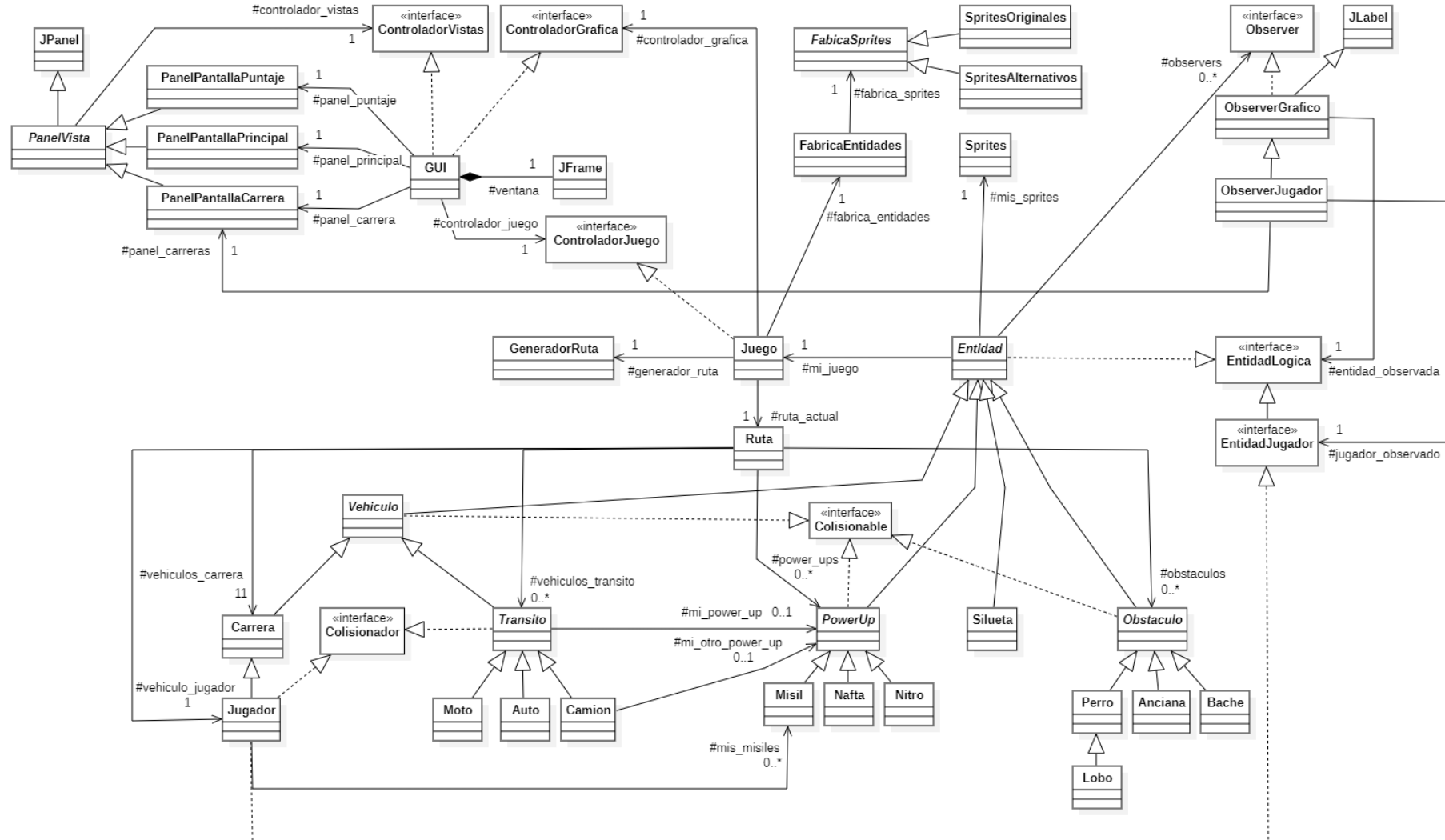


En este **fragmento** del modelo propuesto, se observan **algunos puentes** de interconectan diferentes **capas**.

Esta **punte** permite **mantener** las **responsabilidades** de **lógica** y **vista separadas**, pero al mismo tiempo que **toda modificación** en una entidad **lógica** se vea **replicada** en su correspondiente entidad **lógica**.

Esta imagen es un **fragmento del modelo**, que deliberadamente **invisibiliza** algunas de las relaciones modeladas, para centrarse en el aspecto más relevante de slide actual de la presentación.

Visualización del diagrama reducido completo



Implementación del diagrama propuesto

- Como tarea, **analice y comprenda el diagrama de clases**, que puede encontrar en el siguiente enlace (archivo para StarUML):
 - <https://github.com/tecnologiaprogramacion/road-fighter-3-capas/>
- En el mismo enlace, puede observar una **implementación parcial** del modelo propuesto, **no exenta de errores** o funcionalidad todavía a incorporar:
 - Observe como se **respetan los nombres del modelo** en la **implementación** propuesta.
 - Note las **convenciones de identificadores es respetada** en **todo momento**.
 - Note cómo se **sobre-simplifican** todas las operaciones, en post de que cada una **haga más o menos una sola tarea**, mejorando notablemente la **legibilidad** del código fuente.
 - Note como **no existe prácticamente ningún comentario** en el código fuente: **¿por qué ocurrirá esto?**



¡Homework!

Analice, estudie, comprenda y tome los mejores tips de una implementación parcial del modelo propuesto para el Road Fighter.



Fin de la presentación.



Referencias

- **Primer Diagrama de clases para Road Fighter.** Repositorio TdP en GitHub.
<https://github.com/tecnologiaprogramacion/road-fighter/>
- **Refactorización según Modelo 3 Capas del Diagrama de clases para Road Fighter.** Repositorio TdP en GitHub.
<https://github.com/tecnologiaprogramacion/road-fighter-3-capas/>