



UNIVERSITAT  
POLITÈCNICA  
DE VALÈNCIA



UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Diseño y desarrollo de un videojuego de gestión  
empresarial y logística usando el motor de juegos Godot.

Trabajo Fin de Grado

Grado en Ingeniería Informática

AUTOR/A: Pérez López, Víctor

Tutor/a: Molina Marco, Antonio

CURSO ACADÉMICO: 2023/2024

# Resumen

---

Este proyecto se centra en el diseño y desarrollo de un videojuego de gestión empresarial y logística utilizando como herramienta el motor de juegos Godot, siendo la creación del videojuego el objetivo principal.

El proceso se inicia con una revisión de la oferta actual de juegos dentro del nicho de simulación empresarial, analizando las fortalezas y debilidades de cada producto existente y usando los resultados del análisis para perfilar nuestra propuesta.

La fase de análisis da paso a la de diseño, en la que se elaboran los diagramas de clases y de casos de uso para nuestra propuesta, así como la creación de múltiples prototipos para ilustrar el diseño inicial de las pantallas.

A continuación, se habla del proceso de implementación de la propuesta, de los problemas encontrados y de las soluciones ofrecidas. También se menciona cómo se exportó el proyecto.

Por último, se presentan los resultados de la prueba realizada con usuarios, cómo fue su experiencia en el juego y qué cosas cabría mejorar.

**Palabras clave:** videojuego, Godot, simulación, logística, transporte.

# Abstract

---

This Project focuses on the design and development of a business and logistics management videogame using the Godot game engine as a tool, being the creation of the videogame the main objective.

The process starts with a review of the current offer of games within the business simulation category, studying the strengths and weaknesses of each existing product, and using the results to define our proposal.

The analysis phase gives way to the design phase, in which the class and use case diagrams for our proposal are created, as well as the drawing of multiple mock-ups to illustrate the initial design of the screens.

This is followed by a description of the implementation process of the proposal, the problems that were encountered and the solutions offered. We also mention how the project was exported.

Finally, the results of a survey to the users are presented, showing how their experience with the game was, and what could be improved.

**Keywords** : videogame, Godot, simulation, logistics, transport.

# Tabla de contenidos

---

1.	Introducción.....	9
1.1.	Motivación .....	10
1.2.	Objetivos .....	11
1.3.	Metodología .....	12
2.	Estado del arte.....	13
2.1.	Elección del motor de juego .....	13
2.1.1.	Definición de motor de juego .....	13
2.1.2.	Comparativa de motores .....	13
2.1.3.	Motor escogido.....	16
3.	Análisis del problema .....	17
3.1.	Psicología del juego .....	18
3.1.1.	Teorías .....	18
3.1.2.	Encuadre de la propuesta dentro de los modelos.....	21
3.2.	Propuestas similares.....	23
3.2.1.	Metodología de selección de propuestas .....	23
3.2.2.	Juegos similares.....	24
3.2.3.	Análisis de los resultados del estudio .....	35
3.2.4.	Extrapolación a la propuesta.....	36
3.3.	Solución propuesta.....	37
3.3.1.	Conceptos previos .....	37
3.3.2.	Propuesta.....	38
3.3.3.	Requisitos de la propuesta.....	40
3.4.	Planificación del trabajo .....	41
3.4.1.	Cronograma.....	42
3.4.2.	Presupuesto.....	44
4.	Diseño .....	45
4.1.	Diagrama de clases UML.....	46
4.2.	Diagrama de casos de uso .....	48
4.3.	<i>Mock-ups</i> de las pantallas .....	51
4.4.	Tecnologías empleadas.....	56

5.	Desarrollo.....	57
5.1.	Diagrama implementado.....	58
5.2.	Estructura del proyecto .....	60
5.3.	Adaptaciones al motor .....	61
5.4.	Decisiones tomadas.....	65
5.4.1.	Selección de elementos .....	65
5.4.2.	Navegación.....	66
5.4.3.	Distritos y tiendas.....	67
5.5.	Aspecto final.....	68
5.6.	Implantación.....	71
6.	Validación con usuarios .....	72
6.1.	Preguntas de la encuesta .....	72
6.2.	Respuestas obtenidas.....	75
7.	Conclusión .....	78
7.1.	Lecciones aprendidas .....	79
7.2.	Relación con los estudios .....	81
7.3.	Competencias transversales .....	82
8.	Trabajo futuro.....	83
9.	Bibliografía y referencias .....	84



# Índice de figuras

---

Ilustración 1. Captura de pantalla promocional de PayDay3 (2023), videojuego del género FPS con temática criminal (heisting) desarrollado por la compañía Starbreeze Studios usando el motor de juegos Unreal Engine .....	14
Ilustración 2. Captura de pantalla promocional de Road 96 (2021), videojuego de aventura y exploración desarrollado por el estudio Digixart usando el motor de juegos Unity.....	15
Ilustración 3. Captura de pantalla promocional de Brotato (2023), videojuego de acción tipo roguelike desarrollado por el estudio Blobfish usando Godot. ....	16
Ilustración 4. Modelos de motivación encontrados por Quantic Foundry. ....	20
Ilustración 5. Captura de pantalla promocional de Industry Giant (1998) .....	24
Ilustración 6. Industry Giant 2 (2015).....	25
Ilustración 7. Captura de pantalla promocional de Industry Giant 4.0 (2024).....	25
Ilustración 8. Número de jugadores de Industry Giant en Steam .....	26
Ilustración 9. Número de jugadores de Industry Giant 2 en Steam .....	27
Ilustración 10. Captura de pantalla promocional de Rise of Industry (2019) .....	29
Ilustración 11. Número de jugadores de Rise of Industry en Steam.....	30
Ilustración 12. Captura de pantalla de Workers & Resources: Soviet Republic (2019) .	32
Ilustración 13. Número de jugadores de Workers & Resources: Soviet Republic en Steam .....	33
Ilustración 14. Diagrama de clases UML.....	46
Ilustración 15. Diagrama de casos de uso, parte 1. ....	48
Ilustración 16. Diagrama de casos de uso, parte 2.....	49
Ilustración 17. Mock-up de la pantalla principal. ....	51
Ilustración 18. Mock-up del mapa de una ciudad.....	52
Ilustración 19. Mock-up del menú de las tiendas. ....	53
Ilustración 20. Mock-up de la pantalla de las fábricas.....	53
Ilustración 21. Mock-up de la pantalla de los depósitos.....	54
Ilustración 22. Mock-up de la pantalla de rutas. ....	55
Ilustración 23. Diagrama de clases UML implementado. ....	58
Ilustración 24. Menú principal.....	68
Ilustración 25. Pantalla de selección de mapa. ....	68
Ilustración 26. Mapa del juego.....	69
Ilustración 27. Menú de los depósitos de vehículos. ....	69
Ilustración 28. Menú de las fábricas.....	70
Ilustración 29. Menú de edición de rutas.....	70

Ilustración 30. Frecuencia de uso de videojuegos. ....	75
Ilustración 31. Pregunta sobre la intuitividad de los menús. ....	76

# Índice de tablas

---

Tabla 1. Comparativa con propuestas similares.....	35
Tabla 2. Distribución temporal de las tareas. ....	42



# 1. Introducción

---

¿En qué momento dejamos de ser niños o niñas?

Una de las respuestas que parecen intuitivas es “cuando dejamos de jugar”. Es fácil asociar la pérdida de ilusión e imaginación con el cambio de etapa, cuando nuevos desafíos y problemas se nos presentan, y requieren una mentalidad más realista, pragmática y con los pies en la tierra.

El juego, como comportamiento observado en animales, es una parte importante de la infancia de algunas especies. Según Bruner, Jolly y Sylva[1], la evolución favorece a las especies que expresan sus instintos en situaciones lúdicas, en las que no hay una necesidad inmediata. En uno de los ejemplos, hablan de cómo los tigres, a pesar de tener el impulso de acechar a una presa y saltar sobre ella por sorpresa, la ausencia de práctica durante su etapa de cachorros a través de juegos con sus padres o hermanos podría mermar sus éxitos en la vida adulta.

Pero también hablan de cómo el juego es importante en adultos. En la página 108, afirman que “los jóvenes y los adultos no se encuentran muy lejos de los niños en estas travesuras”, y nos ponen como ejemplo el manteamiento que practicaban los soldados romanos y que tanta impresión causó en Sancho Panza, o las bromas que se gastan los estudiantes entre sí o a los profesores.

El juego es un comportamiento animal, propio de todas las edades, que se alimenta de nuestros instintos. Es uno de los otros tantos vestigios evolutivos que los humanos hemos arrastrado, y alrededor de los cuales hemos construido nuestras culturas y nuestras sociedades.

E igual que los romanos satisfacían esa necesidad lúdica con el manto (en latín, sagatio), en tiempos modernos recurrimos a otras alternativas menos crueles. Actualmente, una de ellas, son los videojuegos.

El uso de la tecnología nos permite idear situaciones lúdicas en las que expresar los diferentes deseos e instintos que pueden tener los diferentes perfiles de cada jugador, y en los que se ahondará más adelante.



## 1.1. Motivación

La motivación de crear un videojuego se basa en dos de los principios, que veremos más adelante, que impulsan a jugarlos: la capacidad de creación y expresión, y la gratificación de completar un reto.

Por una parte, el diseño de un videojuego otorga una serie de licencias que el desarrollo de una aplicación de otra clase no permite. El software tiende a concebirse atendiendo a unas necesidades concretas y proponiendo una serie de soluciones confeccionadas a medida para el problema en cuestión. Pero ¿qué solución se le puede dar al aburrimiento? ¿O al deseo de vivir, explorar y experimentar? ¿Tiene una solución concreta la mente humana?

En este sentido, se podría dar a los videojuegos un enfoque similar al que se le da a los libros, a las películas o a la lírica. Como dijo Gautier en su prefacio de *Mademoiselle de Maupin*[2]: *Ars gratia artis*, o “El arte por la gracia del arte”. En este contexto, uno de los motivos para la creación de un videojuego es el mero acto de creación, es la estimulación artística y creativa.

Sin embargo, el desarrollo de un videojuego requiere una serie de destrezas y conocimientos que un libro o un guion no exigen. La benevolencia del papel y la tinta se encuentra lejos de la sofisticación que proporciona la plétora de herramientas disponibles para la creación de videojuegos. Algunas son más sencillas, otras más potentes; todas ellas exigen un periodo de entrenamiento y aprendizaje como requisito previo a la creación de cualquier obra, y puede suponer una barrera u obstáculo para las personas que las utilizan por primera vez.

Este reto es el segundo elemento motivador que impulsa este proyecto: el afán de superación y consecución de objetivos. En un primer momento, se tuvo la idea de programar un videojuego directamente en C++, por el reto que suponía y como excusa para aprender el lenguaje en profundidad. Sin embargo, finalmente se decidió usar un motor de juego que facilitase el desarrollo, permitiendo que el desafío se estableciera acorde a cómo de complejo se deseara diseñar el juego.

La temática logístico-industrial se escogió por dos motivos. Por una parte, los intereses propios coinciden con las temáticas y mecánicas que presentan juegos con esta tipología: la gestión de la cadena de suministro, la posibilidad de diferentes estrategias de diferenciación-precio, y las diferentes estrategias de expansión en situaciones de bonanza económica.

Por otra parte, un juego centrado en la simulación de negocios implica un menor peso de los efectos visuales o de escenas de acción, por lo que la complejidad de su desarrollo estriba en mayor medida en la codificación de procesos de negocio, dependiendo del nivel de detalle y de la profundidad que se le quiera dar a la simulación; es un tipo de juegos más apto para desarrolladores con un perfil de programadores de lo que lo serían, por ejemplo, un juego de disparos, de plataformas o de rol.

## 1.2. Objetivos

El objetivo angular de este proyecto es la creación de un videojuego de gestión empresarial y logística usando el motor de juegos Godot. No obstante, también se quiere enmarcar dentro de este proyecto el supuesto de comercialización del mismo, con un estudio previo y otro posterior. Por tanto, los objetivos principales quedarían de la siguiente manera:

- Estudiar el sector de los juegos de gestión y simulación empresarial
- Diseñar las especificaciones de un videojuego de gestión y simulación empresarial
- Desarrollar una versión MVP del videojuego obedeciendo a los requisitos y especificaciones establecidas
- Recoger y analizar información sobre la experiencia de los jugadores y de las jugadoras.
- Formular una serie de posibles mejoras a partir de los datos obtenidos de los usuarios y las usuarias.

## 1.3. Metodología

Para este proyecto, se tenía inicialmente planificado el uso de una metodología de desarrollo iterativo, con al menos dos iteraciones con sus respectivos entregables.

Sin embargo, por limitaciones temporales y algunos obstáculos encontrados, se optó finalmente por emplear una metodología más tradicional de cascada, el equivalente a un único ciclo iterativo.

La memoria refleja esta metodología, y sigue secuencialmente las fases que se llevaron a cabo. Se comienza con la etapa de análisis de la situación, seguida por el diseño de la propuesta. A continuación, se llevó a cabo el desarrollo e implantación. Por último, tuvo lugar una pequeña prueba de validación por parte de los usuarios y usuarias.

Cada una de estas fases se llevó a cabo una única vez por los motivos expuestos, y no se pudieron realizar posteriores iteraciones.

## 2. Estado del arte

---

El desarrollo de un videojuego requiere de una serie de herramientas específicas para diferentes tareas. El entorno en el que se ejecute el juego es la más evidente, pero conviene recordar otras tareas como el diseño de los diferentes diagramas o el dibujo del arte que se va a utilizar.

### 2.1. Elección del motor de juego

La discusión más importante es, sin embargo, acerca del entorno de juego. Existe una colorida plétora de motores de juegos, entre los que conviene destacar tres: Unreal Engine, Unity, y Godot.

#### 2.1.1. Definición de motor de juego

Comenzaremos definiendo el concepto de motor de juego. De acuerdo con Wirtz[3]:

“Un motor de juego es un software o entorno de desarrollo. Consiste en varios ajustes y configuraciones que optimizan y simplifican el proceso de creación de videojuegos para desarrolladores en múltiples lenguajes de programación.

Un motor de juego a veces es llamado “arquitectura de juego” o “*framework* de juego”. Puede incluir un motor de renderizado de gráficos 2D o 3D compatible con diferentes formatos de importación, un motor de físicas para simular actividades del mundo real, un motor de sonido, inteligencia artificial (IA), etc.

En resumidas cuentas, los motores de juego proporcionan un *framework* para que los desarrolladores puedan trabajar en el contenido del juego.

#### 2.1.2. Comparativa de motores

Sobre las principales alternativas que existen en el mercado, podemos decir lo siguiente de los tres motores anteriormente mencionados.

##### 2.1.2.1. Unreal Engine

Unreal Engine es uno de los motores de gráficos 3D para juegos y herramienta de creación más populares (Wirtz)[3]. Presenta una serie de facilidades que lo hacen muy atractivo para el desarrollo profesional, tales como la integración de *pipelines* que facilita la conexión con flujos de trabajo multimedia, el editor de mundos Unreal Editor para crear niveles con capacidad multi-usuario en tiempo real, la posibilidad de crear

Diseño y desarrollo de un videojuego de gestión empresarial y logística usando el motor de juegos Godot.

efectos visuales con calidad de película, su soporte para multijugador e IA avanzada, y su uso de C++.

Todo este paquete de atractivas características es gratuito de usar mientras el proyecto en cuestión genere menos de \$1.000.000. A partir de ese momento, se aplica una tarifa de un 5% sobre los ingresos del proyecto. El elevado umbral, sin embargo, no nos debería preocupar para proyectos pequeños.



*Ilustración 1. Captura de pantalla promocional de PayDay3 (2023), videojuego del género FPS con temática criminal (heisting) desarrollado por la compañía Starbreeze Studios usando el motor de juegos Unreal Engine. Fuente: página de Steam del producto[4]. [https://store.steampowered.com/app/1272080/PAYDAY\\_3/](https://store.steampowered.com/app/1272080/PAYDAY_3/)*

Unreal Engine es, por tanto, una herramienta muy potente, ideal para equipos de trabajo profesionales y proyectos de alto acabado. No es, sin embargo, el motor más idóneo para juegos más pequeños y con un equipo de desarrollo reducido o, en este caso, unipersonal.

### **2.1.2.2. Unity**

En segundo lugar, tenemos el motor Unity. John Riccitiello, anterior CEO de Unity Technologies, empresa que desarrolla el motor, afirmó que “la mitad de los juegos se construyen en Unity”[5].

Se trata de un motor de juegos multiplataforma, con capacidad para gráficos en dos y tres dimensiones, y es conocido por su flexibilidad, característica que contribuye a su amplio uso. Otras ventajas frente a Unreal Engine son su facilidad de uso, la cantidad de contenido (*assets*) en su Unity’s Asset Store, sus herramientas para la animación, y sus capacidades para nuevas tecnologías como realidad aumentada (AR) y virtual (VR)[6].

Su política de precios es más similar a otras aplicaciones populares. La versión base es gratuita de usar, pero existen otras versiones, entre las que se encuentran Unity Pro y Unity Industry, con un precio de \$2.040 y \$4.950 al año por licencia, respectivamente.



Ilustración 2. Captura de pantalla promocional de Road 96 (2021), videojuego de aventura y exploración desarrollado por el estudio Digixart usando el motor de juegos Unity. Fuente: página de Steam del product[7]. [https://store.steampowered.com/app/1466640/Road\\_96/](https://store.steampowered.com/app/1466640/Road_96/)

Unity se presenta, por tanto, como una herramienta versátil, suficientemente intuitiva y accesible para equipos pequeños y con poca experiencia en el desarrollo de juegos, pero a la vez capaz de satisfacer con creces las necesidades que pueden requerir proyectos más grandes.

Suena una opción muy atractiva para desarrollar nuestra propuesta. Sin embargo, cabe poner sobre la mesa un tercer y último competidor: Godot.

### 2.1.2.3. Godot

Su característica más llamativa y aclamada es su naturaleza *open source*, esto es, código abierto. Es un motor que, a diferencia de Unreal y Unity, es totalmente gratuito de usar en cualquier caso; también es altamente personalizable, dado el fácil acceso a su código fuente.

En otros aspectos, es comparable con Unity: soporta gráficos 2D y 3D, es versátil y sencillo de usar. Sus dos principales inconvenientes, a fecha de redacción, están relacionados con su relativa novedad en el mercado. Su falta de presencia implica una menor cantidad de tutoriales o cursos disponibles en Internet, y su portal de contenido está notablemente mucho menos poblado que la Unity's Asset Store[8]. Además, algunos usuarios y usuarias destacan la escasez de documentación disponible.





Ilustración 3. Captura de pantalla promocional de Brotato (2023), videojuego de acción tipo roguelike desarrollado por el estudio Blobfish usando Godot. Fuente: página de Steam del producto[9]. <https://store.steampowered.com/app/1942280/Brotato/>

### 2.1.3. Motor escogido

Godot parece una alternativa algo peor en ciertos aspectos a Unity. La falta de recursos formativos, unida a la falta de experiencia en el desarrollo de videojuegos puede suponer un obstáculo. Sin embargo, vistas las similitudes, se decide optar por Godot por su código abierto, como forma de promocionar este tipo de software y darle mayor visibilidad y presencia.



### 3. Análisis del problema

---

Como se ha explicado anteriormente, la propuesta se plantea como una alternativa de ocio; el problema que podría decir que se resuelve es el del aburrimiento o la inquietud humana.

A pesar de esto, existe cabida a un análisis sobre cómo enfocar la propuesta. La literatura académica disponible se nutre de una serie de obras muy relevantes sobre la psicología del juego, con autores como Huizinga y Sutton-Smith.

La explosión del mercado de los videojuegos ha alimentado, a su vez, el creciente interés en estudios sobre los consumidores, así como la aparición de consultoras especializadas en trabajos de este mercado; estas fuentes también serán de interés y se tendrán en cuenta para desarrollar la propuesta.

Por último, cualquier producto o servicio propenso a ser comercializado debe haber sido concebido con una visión clara del mercado en el que se lanzará. Así, se realizará un análisis completo, si bien no en profundidad ni extensión, de otras propuestas de juegos existentes.

## 3.1. Psicología del juego

Antes de proceder a la parte más práctica del trabajo, queremos hacer un breve repaso de algunas de las teorías y aportaciones más importantes desde los campos de la psicología y la sociología sobre el juego.

### 3.1.1. Teorías

#### 3.1.1.1. Johan Huizinga

El primer trabajo al que haremos referencia será a Johan Huizinga y su obra *Homo Ludens*[10]. Las aportaciones de Huizinga han sido ampliamente citadas, y pueden ser consideradas la semilla de las teorías modernas sobre el juego.

En su trabajo, Huizinga establece una serie de condiciones o reglas que pueden delimitar el juego:

- Debe ser libre, o no obligatorio
- No forma parte de la vida ordinaria
- Tiene su propio espacio y su propio tiempo
- Crea orden a través de reglas
- No es productivo ni tiene interés material, no genera riqueza

No es necesario hacer un análisis profundo para afirmar que estas reglas se aplican, si no a todos, a la inmensa mayoría de videojuegos. Cabe hacer un apunte, no obstante. La última condición, que estipula que no tiene interés material, se incumple durante las competiciones y torneos, en los que los jugadores pueden recibir cuantías de dinero por sus victorias. No es, sin embargo, diferente a tantos otros juegos que se pueden practicar de forma profesional, tales como el ajedrez o la mayoría de los deportes.

#### 3.1.1.2. Roger Caillois

El segundo autor que queremos mencionar es Roger Caillois, por su obra *Les jeux et les hommes*[11]. Caillois se basa en, y critica, el trabajo de Huizinga, y hace sus propias aportaciones. Matiza dos de las condiciones del juego:

- Involucra ficción, que crea esa realidad separada que no forma parte de la vida ordinaria y real.
- Las reglas que lo rigen suspenden las leyes cotidianas.

Y añade una más:

- Implica incertidumbre, no se puede prever su resultado

Pero la aportación de Caillois más relevante para este trabajo son sus cuatro formas de juego:

- *Agon*, competición. Los participantes juegan para derrotar al resto y alzarse con la victoria, usando sus habilidades o talentos. Un ejemplo podría ser el ajedrez.
- *Alea*, suerte. El jugador o jugadores desafían a la fortuna para alzarse con la victoria. Se da, por ejemplo, en las máquinas tragamonedas.
- *Mimicry*, fantasía. El atractivo de esta forma de juego es la ilusión y el escape de la realidad. El juego Dragones y Mazmorras cae dentro de esta categoría.
- *Ilynx*, sensaciones. Algunas actividades lúdicas se centran en causar alteraciones de los sentidos o experiencias intensas. Las montañas rusas, los deportes de riesgo o incluso los niños y las niñas cuando juegan a dar vueltas hasta marearse son algunos ejemplos.

Estas formas de juego no son estancas, y pueden coexistir en un mismo juego. Por ejemplo, el póker tiene un fuerte elemento de suerte, pero también tiene una naturaleza competitiva. Los juegos de cartas coleccionables, por otra parte, involucran las tres primeras formas de juego, con la suerte determinando qué cartas se obtienen cuando un jugador compra un sobre, y la fantasía y la competición cuando dos oponentes se batan el duelo.

En el campo de los videojuegos, también es fácil ver las tres primeras formas de juego. La mayoría de los juegos de disparos en primera persona (FPS) tienen un fuerte aspecto competitivo, pero también de fantasía, poniendo al jugador en la piel de un soldado de élite, un superviviente apocalíptico o un peligroso criminal, y con un factor de suerte importante que determina dónde aparece el jugador después de morir, cómo afectará el retroceso a su arma o, en algunos casos, qué trayectoria exacta seguirá la bala.

### 3.1.1.3. Brian Sutton-Smith

El tercer autor que forma parte de este compendio es Brian Sutton-Smith, que en su obra *The Ambiguity of Play*[12], nos proporciona sus siete retóricas del juego. Sus aportaciones van más allá de estas, y su obra ha sido muy influyente en las posteriores investigaciones en sociología y psicología del juego. Sin embargo, en pos de la concisión, nos centraremos solamente en las retóricas que propuso:

- Destino: aquellos juegos que incorporan elementos de azar, en los que el jugador desafía a su fortuna.
- Poder: dominación y victoria sobre los oponentes, el juego como expresión de los talentos y habilidades del jugador.
- Identidad: juegos que permiten construir y reforzar el sentido de pertenencia a un grupo o comunidad, juegos como afirmación de una identidad social.
- Frivolidad: el juego como escape de la realidad, sin ganar ni perder nada importante.
- Progreso: juegos en los que el jugador puede desarrollar sus habilidades.
- Imaginación: los juegos como herramienta creativa, permiten al jugador plasmar sus ideas y ser innovadores.
- Ser: la capacidad de poder expresarse, de crear una identidad alternativa, y de evadirse de la real.



### 3.1.1.4. Modelo de las doce motivaciones

Por último, nos gustaría citar el modelo de las doce motivaciones de los jugadores (*Gamer Motivation Model*) desarrollado por la compañía *Quantic Foundry*[13], dedicada a la investigación de mercado y especializada en el sector de los videojuegos. En su modelo, proponen doce elementos motivadores, agrupados por parejas:

- Acción:
  - o Destrucción: explosiones, armas de fuego, etc.
  - o Emoción: frenesí, acción, sorpresas, tensión.
- Social:
  - o Competición: duelos, mejor puntuación.
  - o Comunidad: pertenencia a un equipo, chatear, interactuar.
- Maestría:
  - o Desafío: práctica, dificultad, retos.
  - o Estrategia: planificación, decisiones.
- Logros:
  - o Compleción: todos los coleccionables, completar todas las misiones.
  - o Poder: personaje fuerte, equipamiento poderoso.
- Inmersión:
  - o Fantasía: ser otra persona, en otro lugar.
  - o Narrativa: argumentos profundos, personajes complejos.
- Creatividad:
  - o Diseño: expresión, personalización.
  - o Descubrimiento: exploración, experimentación.

## GAMER MOTIVATION MODEL



Action "Boom!"	Social "Let's Play Together"	Mastery "Let Me Think"	Achievement "I Want More"	Immersion "Once Upon a Time"	Creativity "What If?"
<b>Destruction</b> Guns. Explosives. Chaos. Mayhem.	<b>Competition</b> Duels. Matches. High on Ranking.	<b>Challenge</b> Practice. High Difficulty. Challenges.	<b>Completion</b> Get All Collectibles. Complete All Missions.	<b>Fantasy</b> Being someone else, somewhere else.	<b>Design</b> Expression. Customization.
<b>Excitement</b> Fast-Paced. Action. Surprises. Thrills.	<b>Community</b> Being on Team. Chatting. Interacting.	<b>Strategy</b> Thinking Ahead. Making Decisions.	<b>Power</b> Powerful Character. Powerful Equipment.	<b>Story</b> Elaborate plots. Interesting characters.	<b>Discovery</b> Explore. Tinker. Experiment.

Ilustración 4. Modelos de motivación encontrados por Quantic Foundry.  
<https://quanticfoundry.com/#motivation-model>

### 3.1.2. Encuadre de la propuesta dentro de los modelos

Después de hacer un repaso a la literatura disponible, y de exponer algunos de los modelos motivacionales, resulta inevitable querer enmarcar la propuesta en ellos.

Empezando por las cuatro formas de juego de Caillois, podemos descartar desde un primer momento la forma *Ilynx*, quedándonos las otras tres. La forma que mejor cuadra para nuestra propuesta es *Mimicry*, fantasía, proporcionando al jugador la posibilidad de convertirse en una persona de negocios, gestionando su cadena de suministro y red logística, y tomando decisiones sobre dónde abrir tiendas, qué productos ofrecer, y a qué precios.

La propuesta no contempla, inicialmente, la implementación de una inteligencia artificial que pudiera hacer de rival para el jugador. Esta mejora, que potenciaría la experiencia, permitiría incluir la forma de juego *Agon*, competitividad, y una pequeña parte de *Alea*, suerte, de la cual podría depender el comportamiento de la IA.

Dentro del modelo de Sutton-Smith, encontramos varias retóricas que se emplean en el juego. La más clara es la retórica del ser, por la misma razón que antes apuntábamos que una de las formas de juego es *Mimicry*: nos permite meternos en el papel de un emprendedor, o de un directivo, y de gestionar el complejo ecosistema de fábricas, tiendas y rutas de transporte que permitan obtener una rentabilidad aceptable.

Emplea también, en menor medida, la retórica de la imaginación, concediendo al jugador la libertad de elegir qué forma tomará su imperio industrial, y plasmar sus ideas de cómo debería ser una gran corporación; tal vez quiera crear un gigante tecnológico, un conglomerado con una cartera de productos diversificada, o cumplir su sueño de ser la mayor empresa fabricante de juguetes del país.

Otras dos retóricas que usa, al igual que la mayoría de los videojuegos, son la de la frivolidad y la del progreso. Por una parte, son herramientas que permiten desconectar de los quehaceres diarios, en las que nuestras acciones no tienen consecuencias más allá del universo del juego, de donde se extrae el uso de la retórica de frivolidad. Por otra parte, la mayoría de los videojuegos incorporan sistemas de mecánicas propensos a ser comprendidos y dominados, de forma que cuanto más tiempo un usuario pasa jugando, mejores son sus habilidades dentro de este sistema.

Por último, del modelo propuesto por Quantic Foundry, usa de forma clara un total de tres motivaciones: estrategia, fantasía y diseño.

El componente de estrategia es fácil de intuir; el jugador tendrá que tomar decisiones interrelacionadas sobre cómo distribuir las fábricas en el espacio, cómo se conectan entre sí con las rutas de transporte, dónde abrir las tiendas dentro de las ciudades y qué precios establecer para los productos.

La motivación de la fantasía ya ha aparecido en los modelos de las formas de juego de Caillois y en el de las retóricas de Sutton-Smith. La premisa del juego es la posibilidad de convertirse en un empresario virtual y crear su propio imperio,



Diseño y desarrollo de un videojuego de gestión empresarial y logística usando el motor de juegos Godot.

sustituyendo las preocupaciones diarias por otras como la forma más económica de trasladar bienes del punto A al punto B, o de qué productos pueden ser más atractivos en un barrio humilde.

La motivación del diseño también se ha mencionado para el modelo de las retóricas: lejos de juegos como el ajedrez, aquí las estrategias viables son múltiples, y permiten al jugador expresar qué tipo de negocio quiere crear.

## 3.2. Propuestas similares

Una consideración importante a tener en cuenta cuando se lanza un producto es qué competidores puede tener. En pos de la simplicidad, delimitaremos la búsqueda a videojuegos similares, ignorando otros productos sustitutos, como los juegos de mesa, los salones recreativos o incluso series y películas.

A su vez, aunque se han encontrado un amplio número con un rango de paralelismos con la propuesta, solamente analizaremos los más parecidos. Al final de la sección, nombraremos aquellos que han quedado fuera del estudio, pero comparten características con la propuesta.

### 3.2.1. Metodología de selección de propuestas

Los videojuegos que se enumerarán han sido seleccionados por su grado de similitud a la propuesta, y han sido encontrados a través de búsquedas en navegadores generales, así como haciendo uso de las sugerencias del portal de ventas de *Steam*.

Haciendo uso de esta misma página, se proporcionará una concisa descripción del juego, comparándolo con la propuesta. En caso de requerirlo, se usarán otras páginas si la información disponible no es lo suficientemente completa.

Para evaluar la popularidad del juego, se usará la puntuación de las reseñas de *Steam*, así como la herramienta *SteamCharts* (<https://steamcharts.com/>) para conocer el número de jugadores.



## 3.2.2. Juegos similares

### 3.2.2.1. Industry Giant

El primer oponente es la saga en la que se inspira la propuesta: *Industry Giant*. Desarrollada por la empresa germana *JoWood*, su primera entrega salió al mercado en 1997 en Europa, y en 1998 en Norte América[14]. Recibió críticas polarizadas[15], pero resultó un éxito a nivel comercial, con un total de 800.000 ejemplares vendidos.

El segundo juego de la saga, *Industry Giant 2*, fue comercializado en 2002[16]. Obtuvo una mejor recepción de la crítica[17], y fue remasterizado en 2015[18]. En 2021, el estudio de videojuegos *Don VS Dodo* anunció una tercera entrega, *Industry Giant 4.0*. Según el sitio web de la empresa, el juego se encuentra en producción inicial[19], y en su página de *Steam*, aparece con una fecha de lanzamiento de 2024, sin especificar día ni mes.



Ilustración 5. Captura de pantalla promocional de *Industry Giant* (1998)[20]. Fuente: página web de Steam del producto. [https://store.steampowered.com/app/521090/Industry\\_Giant/](https://store.steampowered.com/app/521090/Industry_Giant/)





Ilustración 6. *Industry Giant 2* (2015). Fuente: página web de Steam del producto[21]. [https://store.steampowered.com/app/271360/Industry\\_Giant\\_2/](https://store.steampowered.com/app/271360/Industry_Giant_2/)



Ilustración 7. Captura de pantalla promocional de *Industry Giant 4.0* (2024)[22]. Fuente: página web de Steam del producto. [https://store.steampowered.com/app/1129570/Industry\\_Giant\\_40/](https://store.steampowered.com/app/1129570/Industry_Giant_40/)

Su premisa de juego es similar a la de la propuesta: el jugador podrá ubicar y construir fábricas en el mapa, con el fin de producir bienes, que deberá posteriormente distribuir para que puedan ser comercializados en tiendas. La gestión logística toma un papel muy relevante en el ciclo del juego, por encima de otros aspectos de negocio como los recursos humanos o las finanzas.

La principal diferencia con la propuesta es el enfoque de la distribución minorista; mientras que en *Industry Giant*, el jugador construye tiendas directamente en el mapa, y las ciudades se componen de múltiples edificios urbanos conectados por una red de

Diseño y desarrollo de un videojuego de gestión empresarial y logística usando el motor de juegos Godot.

carreteras, la propuesta de este proyecto opta por enfocar las ciudades como un elemento monolítico en el mapa, con la posibilidad de acceder al plano de cada una de ellas y establecer los comercios a nivel de distrito.

Las puntuaciones de *Industry Giant* en *Steam* son mixtas (69% de las 56 reseñas son positivas), en gran parte debido a los problemas técnicos, dado que es un juego bastante antiguo, a pesar de los esfuerzos de la empresa QLOC en actualizarlo para ser compatible con equipos actuales.

Las puntuaciones de *Industry Giant 2*, que como se ha dicho, fue remasterizado en 2016, son bastante más positivas (87% de las 56 reseñas son positivas).

Accediendo a *SteamCharts*, podemos ver el número de jugadores para ambas entregas lanzadas.

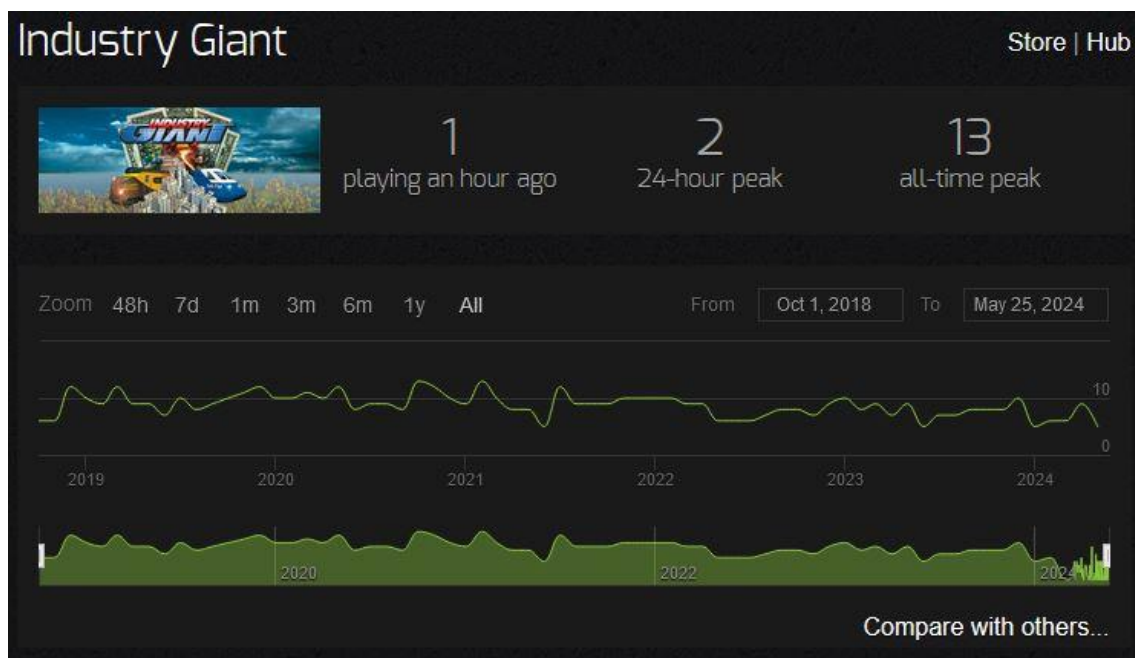


Ilustración 8. Número de jugadores de *Industry Giant* en *Steam*[23]. Momento de captura: 13:23h, 25/05/2024. Fuente: <https://steamcharts.com/app/521090>



Ilustración 9. Número de jugadores de *Industry Giant 2* en Steam[24]. Momento de captura: 13:25h, 25/05/2024. Fuente: <https://steamcharts.com/app/271360>

Como puede observarse, el número de jugadores de la entrega inicial es despreciable. Tampoco son esperanzadoras las cifras de actividad de *Industry Giant 2*, que oscilan sobre los cien jugadores concurrentes.

Estos datos tienen dos lecturas iniciales. Por una parte, es un nicho de mercado poco explotado; los juegos son antiguos, lo que podría explicar el poco interés que suscita entre los usuarios. Sin embargo, esta falta de actividad podría también deberse a un bajo atractivo hacia este tipo de juegos, por lo que nuestra propuesta podría fracasar y obtener una mala recepción. Estudiando otros juegos diferentes podremos aceptar o rechazar estas hipótesis.



### 3.2.2.2. Rise of Industry

Otro juego, similar en varios aspectos a nuestra propuesta y a la saga *Industry Giant*, es *Rise of Industry*, desarrollado por la difunta compañía española *Dapper Penguin Studios*, y lanzado al mercado el 2 de mayo de 2019.

Este juego cuenta con una premisa muy parecida a *Industry Giant*: el jugador debe construir edificios de producción, logística y soporte para proporcionar bienes a las poblaciones que se generan con el mapa. Sin embargo, también presenta una serie de diferencias esenciales.

En primer lugar, la producción jerarquiza los edificios destinados a ella en dos tipos: de extracción y de distribución. Por ejemplo, una planta de tratamiento de aguas se encarga de distribuir la carga líquida que recibe de pozos ubicados en sus cercanías, o las granjas reciben el trigo de las plantaciones cercanas y puede decidir, a su vez, a qué localizaciones enviar estos alimentos.

Relacionado con esto, el juego se centra en la producción de materias primas al inicio de la partida, tales como las granjas agrícolas o las explotaciones ganaderas, que más tarde servirán de base para construir fábricas textiles que usen el cuero o el algodón, por ejemplo. De esta forma, el ciclo de juego de una partida comienza por las materias primas, y se desarrolla “aguas abajo” en la cadena de suministro, conforme el jugador descubre las tecnologías pertinentes.

Un tema presente en *Rise of Industry* es la contaminación. Las fábricas pesadas y el transporte por carretera, especialmente en vías congestionadas, puede generar una alta polución, que el jugador debe minimizar a fin de no perder clientes en las ciudades cercanas.

Otra diferencia radica en el rol de las ciudades. Tanto en *Industry Giant* como en *Rise of Industry*, las ciudades se componen de edificios y carreteras, y crecen con el tiempo. Sin embargo, el juego de la firma española no permite al jugador abrir tiendas en las ciudades, sino que estas se generan con la ciudad. Al inicio, cada ciudad cuenta con un mercado de abastos y una tienda de bricolaje. Conforme la ciudad crece y se satisfacen sus necesidades, estos establecimientos suben de nivel y se abren nuevos tipos de tienda, permitiendo al jugador diversificar su entramado productivo.

Por último, aunque la logística juega un papel importante, llama la atención que en *Rise of Industry* no se pueden adquirir vehículos ni asignarles rutas de transporte personalizadas, como ocurría en *Industry Giant* y como se planea hacer en la propuesta. En su lugar, cada edificio de distribución se construye con un límite de camiones disponibles, y cada uno de ellos puede ser asignado a un edificio que recibirá el tipo de mercancía que el edificio gestione. Se pierde, por tanto, una cierta flexibilidad a la hora de planificar la logística, pero se gana en sencillez y facilidad de uso.



Ilustración 10. Captura de pantalla promocional de *Rise of Industry* (2019). Fuente: página web de Steam del producto[25]. [https://store.steampowered.com/app/671440/Rise\\_of\\_Industry/](https://store.steampowered.com/app/671440/Rise_of_Industry/)

Las reseñas son mayoritariamente positivas (75% de las 2962 reseñas son positivas). Un vistazo rápido a las reseñas negativas nos permite observar que la gran mayoría se deben algunas mecánicas innecesariamente restrictivas y mal explicadas, así como una falta de datos fiables sobre el rendimiento económico. Un número algo más reducido de reseñas se lamenta de la repetitividad y tediosidad del juego.

El número de jugadores activos es notablemente bajo, especialmente teniendo en cuenta que se trata de un lanzamiento relativamente moderno. Con menos de 50 jugadores concurrentes, y un máximo de 1.354 a la vez, *Rise of Industry* se presenta como un mal augurio para otras propuestas similares.



Ilustración 11. Número de jugadores de *Rise of Industry* en Steam[26]. Momento de captura: 21:24, 26/05/2024. Fuente: <https://steamcharts.com/app/671440>

Antes de pasar al siguiente juego, cabe mencionar que *Dappler Penguin Studios* cerró sus puertas por decisión de su fundador de perseguir una carrera como autónomo, y no por un mal rendimiento económico de sus productos[27]. Además, se va a lanzar una segunda entrega, *Rise of Industry 2*, desarrollada por *SomaSim* y distribuida por la misma empresa que el juego original, *Kasedo Games*[28]. Por tanto, a pesar de la pobre recepción, podemos sospechar que, a nivel económico, el juego tuvo buenos resultados.

### 3.2.2.3. Workers & Resources: Soviet Republic

Un juego también similar en varios aspectos es *Workers & Resources: Soviet Republic*, desarrollado por *3Division* y lanzado en *Steam* el 15 de marzo de 2019 como *Early Access* (acceso anticipado, es decir, jugable pero no terminado). Fue lanzado como juego terminado el 20 de junio de este año 2024. Como indica el nombre, el juego propone la construcción de un país comunista de Europa del Este, con elementos de construcción de ciudades (*City Builder*) así como un complejo sistema de cadena de suministro, con múltiples tipos de fábricas y opciones para transportar los bienes, tales como oleoductos, cintas transportadoras e incluso funiculares, así como los más tradicionales camiones, trenes y cargueros.

La principal diferencia entre *Workers & Resources* y nuestra propuesta es la ausencia del enfoque de construcción de ciudades, y el enfoque más centrado en la parte industrial.

Otra divergencia se encuentra en la comercialización de los bienes. En el juego de *3Division*, las tiendas no tienen ánimo lucrativo, sino que su principal finalidad es la de servir de puntos de distribución de los bienes; si un jugador desea obtener dinero por los bienes que su república fabrica, puede hacerlo a través de los edificios de aduanas, situados en los límites del mapa, así como usando cargueros que naveguen masas de agua con salida fuera de la república, y usando helicópteros o aviones de carga.

Otro aspecto de simulación, carente en nuestra propuesta, es la posibilidad que se le ofrece al jugador de que los vehículos consuman combustible y necesiten repostar cuando su depósito se encuentra próximo a agotarse, y que requieran un mantenimiento periódico con el consiguiente tiempo fuera de las carreteras y vías férreas.

Por último, en *Workers & Resources* las fábricas requieren mano de obra para funcionar, que el jugador debe transportar haciendo uso de autobuses, tranvías y metros, entre otros medios.





Ilustración 12. Captura de pantalla de *Workers & Resources: Soviet Republic* (2019)[29]. Fuente: página web de Steam del producto. [https://store.steampowered.com/app/784150/Workers\\_Resources\\_Soviet\\_Republic/](https://store.steampowered.com/app/784150/Workers_Resources_Soviet_Republic/)

Las puntuaciones de *Workers & Resources* en *Steam* son muy positivas (92% de las 14.645 reseñas son positivas). Si miramos únicamente las reseñas recientes, obtenemos el mismo porcentaje de aceptación (92% de las 191 reseñas de los últimos 30 días son positivas).

Las cifras ofrecen una lectura optimista con matices. El juego resulta extremadamente atractivo para sus usuarios. Sin embargo, el número de reseñas, sin ser tan bajo como en la saga de *Industry Giant*, se encuentra lejos de permitir considerar el juego como popular.

Esta hipótesis se refuerza consultando el número de jugadores activos en *Workers & Resources*, con un máximo de 4.005 jugadores de forma concurrente. De nuevo, no es una mala cifra, pero tampoco destacablemente alta.

Una posible explicación para estos números tan dispares es el nivel de complejidad del juego, que lo relega a un nicho pequeño, en el cual sus elementos de simulación lo pueden hacer muy atractivo frente a un público muy selecto.





Ilustración 13. Número de jugadores de *Workers & Resources: Soviet Republic* en Steam[30]. Momento de captura: 14:07h, 25/05/2024. Fuente: <https://steamcharts.com/app/784150>

### 3.2.2.4. Otros juegos similares:

Aunque no lo suficientemente parecidos a la propuesta, nos gustaría tratar muy brevemente los siguientes títulos:

- Trópico 6 (2019): última entrega de una serie de juegos de construcción de ciudades con posibilidad de construir cadenas de suministro y vender los bienes resultantes. La gestión logística es muy pobre, y la profundidad de la cadena de suministro es limitada. Está centrado en la construcción de una república tropical.
- Cities Skylines (2015): otro par de juegos de construcción de ciudades. Tiene un aspecto logístico muy importante, con posibilidad de construir múltiples tipos de carretera para facilitar el tránsito. Sin embargo, el control sobre los edificios industriales es muy escaso, no se pueden configurar las rutas de transporte, y el foco del juego se encuentra en la parte de construcción de la ciudad.
- Transport Fever 2 (2019): juego de simulación de negocios enfocado en la gestión de los medios de transporte, siguiendo su evolución durante las últimas décadas. La administración empresarial tiene un papel secundario.
- Factorio (2020): juego de simulación logística focalizado en la extracción de recursos y su secuencial procesamiento, con elementos de acción en los momentos en los que se producen ataques de criaturas hostiles. Presenta un enfoque más propio de la gestión de una línea de producción en una fábrica que de planificación logística y distribución.

Todos estos juegos presentan un alto número de jugadores activos y una buena puntuación en *Steam*, por lo que el género de gestión parece ser atractivo al público general, si bien el nicho concreto de la propuesta parece reunir un número de usuarios muy reducido.

### 3.2.3. Análisis de los resultados del estudio

A partir de los datos y cifras recopilados, podemos emitir un diagnóstico sobre el estado del género de los videojuegos de gestión.

Este tipo de juegos reúnen a un alto número de jugadores, los suficientes como para hacerlo un mercado atractivo. Las propuestas más populares incluyen mecánicas de construcción de ciudades, y la gestión de la cadena de suministro tiene un peso reducido o moderado. Una notable excepción es *Workers & Resources*, que se focaliza precisamente en este nicho proporcionando una compleja experiencia de simulación. Esta especialización le ha brindado abundantes reseñas positivas entre sus jugadores.

Los juegos de gestión de negocios y logística, sin embargo, representan un segmento de mercado algo menos atractivo, con menor número de potenciales usuarios, cuyas necesidades, atendiendo a las reseñas negativas, no se encuentran correctamente atendidas.

*Industry Giant 2* puede ser un ejemplo de juego exitoso por sus positivas reseñas, si bien no tanto por su número de jugadores, y presenta mecánicas atractivas que pueden servir de base para futuras propuestas en el género.

Por otra parte, *Rise of Industry* nos revela algunos errores que se deben evitar, tal como restringir la libertad del jugador de forma innecesaria, no ofrecer suficiente información económica, o la falta de dinamismo que pueda combatir la monotonía durante el ciclo del juego.

A continuación, se elabora una tabla resumen con las conclusiones sacadas:

Juego	Fortalezas	Debilidades	Recepción
<i>Industry Giant</i>	-Opciones de comercialización -Simplicidad	-Obsolescencia -Monotonía	Intermedia
<i>Industry Giant 2</i>	-Opciones de comercialización -Variedad de productos	-Monotonía	Positiva
<i>Rise of Industry</i>	-Simplicidad de -Sensación de progreso	-Monotonía -Falta de información	Negativa
<i>Workers &amp; Resources</i>	-Profundidad -Realismo	-Complejidad -No hay comercialización	Muy positiva

Tabla 1. Comparativa con propuestas similares. Elaboración propia.

### **3.2.4. Extrapolación a la propuesta**

A partir de las conclusiones sacadas del apartado anterior, podemos seleccionar qué elementos podría tener nuestra propuesta, y cuáles deberían ser evitados o mitigados.

En línea con el resto de los juegos, las acciones básicas consistirán en levantar edificios productivos y tejer redes de carreteras para comunicarlos; y en diseñar las rutas de transporte para mover materias primas y productos elaborados desde sus lugares de origen hasta el lugar que establezca el jugador o la jugadora.

Los edificios productivos serán fábricas genéricas, sin separar entre categorías, permitiendo que una misma fábrica tenga líneas de producción diferentes. De esta forma, se reduce la complejidad del juego.

Para el transporte de mercancías, se deberán adquirir vehículos. Los vehículos se adquirirán, almacenarán y controlarán en un almacén de vehículos. Por motivos de alcance y simpleza, solo se contemplan dos tipos de vehículos: de carretera, y de ferrocarril.

Las rutas se establecerán de forma muy similar a los otros juegos: haciendo clic sobre los elementos del mapa, configurando en qué orden se visitarán, y estableciendo qué operaciones de carga y descarga se llevarán a cabo.

Aunque la comercialización no parece un factor directo para determinar el éxito de un juego, sí que parece haber una tendencia inversamente proporcional entre la complejidad de la cadena de suministro y las opciones a la hora de vender productos. Por tanto, se desea ir un paso más allá y permitir al jugador abrir comercios en las ciudades, escogiendo la ubicación en función de la riqueza y popularidad de cada barrio.

Por último, se hará hincapié en que los menús sean fáciles de leer y navegar. Para evitar la monotonía, se propone la creación de adversarios controlados por Inteligencia Artificial que puedan hacer la competencia a negocios del jugador.

## **3.3. Solución propuesta**

### **3.3.1. Conceptos previos**

Antes de formular la propuesta, nos gustaría definir algunos conceptos necesarios sobre el diseño de videojuegos, que nos permitirán guiar y estructurar nuestra idea.

#### **3.3.1.1. Jugabilidad**

El término *gameplay*, o *jugabilidad* como posible traducción, aparece mencionado frecuentemente por los diversos actores existentes en el mercado de videojuegos, desde desarrolladores hasta usuarios, pasando por analistas y periodistas, aunque raramente se explica qué es, y tampoco existe un consenso sobre su definición (Fabricatore)[31].

El *gameplay* podría considerarse un marco que incluye aquello que el jugador puede hacer, y aquello que otras entidades pueden hacer como respuesta a las acciones del jugador (Fabricatore)[31]. Se puede incluir, también, las acciones que ocurren independientemente de la intervención del jugador y que permiten dar “vida” al mundo.

#### **3.3.1.2. Ciclo de juego**

Otro concepto clave para definir la *jugabilidad* de un videojuego es el denominado como *gameplay loop*, que podría traducirse al castellano como ciclo de juego. Consiste en la secuencia repetitiva de acciones que realiza un jugador, y supone el núcleo de la experiencia que disfrutará (Lynch)[32]. Un ciclo de juego entretenido y atrapante mantendrá al jugador interesado durante una mayor cantidad de tiempo, mientras que un ciclo de juego tedioso o poco gratificante resultará en una experiencia aburrida.

### 3.3.2. Propuesta

El objetivo de este juego es que el jugador obtenga los máximos beneficios posibles a través de la manufactura de productos terminados y su comercialización en tiendas de ciudades, transportando materia prima a las factorías y más tarde distribuyendo los productos fabricados.

Al iniciar una partida, se generará un mapa aleatorio consistente en una masa de tierra central rodeada de agua.

Sobre la masa de tierra se generarán un número de ciudades, y de edificios de explotación de materias primas. Otros elementos que se generarán inicialmente serán los puertos, en puntos de costa. Todos ellos, ciudades, edificios y puertos, se interconectarán entre sí mediante carreteras.

El jugador podrá construir una serie de edificios productivos y de distribución, tales como fábricas, almacenes, depósitos de vehículos de carretera y depósitos de ferrocarriles. Además, podrá crear nuevas carreteras o vías férreas. Una última opción de construcción será la de demoler edificios propios, carreteras y vías férreas.

Las fábricas contarán inicialmente con una única línea de producción, sobre la que podrá establecerse la producción de un determinado tipo de mercancía. Las fábricas tendrán espacio para aumentar hasta a un máximo de tres líneas de producción, pagando unos costes crecientes por cada una.

Se podrá cambiar el producto fabricado en cada línea. El coste de este cambio dependerá de las diferencias de materias primas requeridas por el nuevo producto. Por ejemplo, si una línea fabrica muebles, será más barato cambiarla para que fabrique juguetes de madera que cambiarla para que fabrique herramientas eléctricas.

Las fábricas tendrán integrado un pequeño almacén de materias primas. Las líneas de producción consumirán estos recursos y generarán productos de salida, que se acumularán en otro almacén diferente. Este proceso ocurre en tiempo real en intervalos discretos según el tipo de producto. Por ejemplo, un cargamento de muebles puede tardar 20 segundos en procesarse. Hasta que no transcurra este periodo, no se consumirá la madera, ni se generará el producto final. Una vez el ciclo se complete, se iniciará de nuevo, ad infinitum, hasta que el edificio agote las materias primas, hasta que el almacén de productos terminados se sature, o hasta que el jugador elimine la línea de producción.

Los productos tienen un valor intrínseco diferente para cada uno.

La distribución logística se llevará a cabo a través de vehículos de carretera y de ferrocarriles.

Los vehículos se adquieren desde el menú de su depósito correspondiente. Tienen una velocidad máxima dependiendo del tipo, además de un coste de mantenimiento. Los camiones pequeños pueden transportar un único cargamento de

producto, los camiones con tráiler hasta dos cargamentos, y los trenes hasta seis a la vez.

Los vehículos adquiridos pueden ser directamente vendidos desde el menú de su depósito, si no transportan mercancía en ese momento. El precio de venta inicial será el mismo que el coste de compra. Conforme pase el tiempo, el desgaste del vehículo aumentará, y su precio de venta disminuirá hasta un 20% del precio original.

La función de los depósitos será la de hacer de puntos de compra y venta de los vehículos, así como permitir acceder a ellos o moverlos a diferentes depósitos. Cada depósito tiene capacidad para 6 vehículos. Más allá de las labores de gestión, y servir de punto de partida cuando se compra el vehículo y comienza su primera ruta, los depósitos no interactúan con los vehículos en el mapa de forma directa.

Los vehículos del jugador pueden ser seleccionados directamente en el mapa, mostrando su mercancía y próximo destino. Desde el menú de su depósito, aparecerá también esta información. Bien desde el mapa o desde el depósito, se podrá abrir una nueva ventana para gestionar la ruta del vehículo.

Al establecer la ruta de un vehículo, se pueden establecer una serie de destinos. Si se establece un único destino, el vehículo se quedará en él y esperará nuevas órdenes. Para cada destino, se podrá seleccionar qué productos cargar o descargar, y en qué cantidad. También para cada destino, se podrá decidir si el vehículo debe esperar a estar lleno con toda la mercancía establecida antes de seguir con su ruta, o si puede continuar inmediatamente.

Para seleccionar un posible destino, debe de existir una conexión por carretera o por vía férrea. Esto implica que los depósitos deberían estar inicialmente conectados a la red.

Los vehículos por carretera pueden viajar a cualquiera de los siguientes elementos: explotaciones de materia prima, fábricas, almacenes, fábricas y puertos. Los ferrocarriles pueden viajar a estos mismos elementos, con excepción de las ciudades, incluso si están conectadas a la red de vías férreas. Los depósitos no se consideran un destino.

Cuando se selecciona como destino una ciudad, nos aparecerá un submenú listando las tiendas dentro de esa ciudad y en qué distrito se encuentran, para que podamos seleccionar una tienda específica. Todas las tiendas dentro de una ciudad cuentan como un mismo destino en el mapa, la ciudad misma, aunque en la ruta de transporte aparezcan las tiendas individuales.

Las explotaciones de materias primas generan recursos de forma similar a las fábricas. Por ejemplo, los campamentos madereros producirán madera cada tres segundos. Los materiales se almacenan en un almacén similar al de las fábricas.

Cuando un vehículo carga materiales de una explotación, se descuenta de forma automática el precio de la cuenta del jugador. El precio de los bienes es el precio íntegro del material, sin modificadores.



Cuando un vehículo carga o descarga materiales en un puerto, se aplica la suma o resta correspondiente de forma automática a la cuenta del jugador. Adquirir bienes en un puerto es un 10% más caro que en una explotación. Vender bienes en un puerto proporciona el equivalente a venderlos en una ciudad con un -10% de margen. Los puertos aceptan cualquier producto en cualquier cantidad, y cuentan con un stock ilimitado de cualquier producto para vender al jugador.

La carga y descarga de materiales en explotaciones, fábricas, almacenes y puertos tarda siempre 0,5 segundos por cargamento y por operación. Por ejemplo, un tráiler que descarga dos unidades de madera y carga dos de muebles, tardará un segundo en descargar y otro segundo en cargar, es decir, un total de dos segundos.

Si un camión tiene como destino de descarga una ciudad, tardará 0,4 segundos por cada distrito que tenga que visitar, y 0,1 segundos por cada tienda que haya en él. Por ejemplo, si un camión tiene que repartir entre tres tiendas, en dos distritos diferentes, tardará 0,5 segundos ( $0,4 + 0,1$ ) en repartir a la tienda que se encuentra sola en un distrito, y 0,6 segundos ( $0,4 + 0,1 + 0,1$ ) en repartir a las dos tiendas que comparten distrito. Los bienes no aparecerán en el inventario de la tienda hasta que el camión no haya terminado de visitar todos los establecimientos de la ciudad, por lo que, para todas las tiendas del ejemplo, el tiempo real de descarga es de 1,1 segundos.

Si un camión intenta descargar un producto que no se encuentra en el catálogo de una tienda, la ignorará, y no se tendrá en cuenta para calcular los tiempos de descarga.

Las ciudades pueden seleccionarse en el mapa, mostrando su nombre, población y atractivo turístico. Si se accede a ellas, se abrirá un mapa de los distritos de la ciudad, representados por polígonos. Si se selecciona uno de los polígonos, nos mostrará su nombre, población, atractivo turístico, nivel adquisitivo y precio del suelo. Haciendo clic en el botón correspondiente, podemos abrir un nuevo establecimiento.

Al abrir un nuevo establecimiento, podremos escoger la superficie total y si construirlo en una zona concurrida o una zona tranquila. El coste base será la superficie escogida multiplicada por el precio del suelo. Construirlo en una zona concurrida añadirá un coste de un 20% al precio base, y aumentará la concurrencia en un 20%.

La superficie mínima son  $20\text{m}^2$ . Por cada  $10\text{m}^2$  adicionales, se podrá vender un nuevo tipo de producto. Por ejemplo, una tienda de  $40\text{m}^2$  podrá vender 3 tipos de productos, el primero por los  $20\text{m}^2$  mínimos, y otros dos por tener otros  $20\text{m}^2$ .

Las tiendas tienen un catálogo de productos. Para poder recibir y vender un producto, debe estar en el catálogo. Para añadir un producto al catálogo, se debe crear una oferta, estableciendo un precio. Al crear una nueva oferta, el margen por defecto es de 0, y se venderá al precio del bien. Se puede establecer un margen de entre -50% hasta un +100% en intervalos de 10%. Los productos con un margen negativo se venden a menor precio, generan una mayor demanda y aumentan la concurrencia de la tienda. Los productos con valores positivos se venden a mayor precio y generan una menor demanda.

### 3.3.3. Requisitos de la propuesta



A modo de resumen, y con el fin de esquematizar la propuesta elaborada, indicaremos los requisitos funcionales y no funcionales del juego.

### **3.3.3.1. Requisitos funcionales**

- Configuración del número de elementos en el mapa que se genere.
- Generación aleatoria del mapa, así como de los elementos iniciales que lo componen, que son las ciudades, las explotaciones y los puertos.
- Generación aleatoria del mapa de distritos de cada ciudad.
- Construcción de edificios como fábricas, depósitos de vehículos de carretera y de ferrocarril, y almacenes.
- Demolición de los edificios construidos por el jugador, excluyendo aquellos que se generan con el mapa.
- Capacidad de añadir líneas de producción en las fábricas seleccionando su output.
- Transformación de materiales en el inventario de entrada de una fábrica en productos de salida dependiendo de las líneas de producción activas.
- Posibilidad de eliminar líneas de producción de una fábrica.
- Compra de vehículos desde un depósito de vehículos seleccionando su modelo.
- Venta de vehículos desde el depósito de vehículos en el que se encuentran.
- Modificación de la ruta de un vehículo seleccionándolo.
- Posibilidad de añadir nuevos destinos a una ruta seleccionando la localización destino, y estableciendo en el menú correspondiente qué productos se cargarán y descargarán.
- Modificación de las operaciones de carga y descarga de un destino existente en una ruta.
- Capacidad de eliminar destinos de una ruta seleccionada.

### **3.3.3.2. Requisitos no funcionales**

- Uso de algoritmos ligeros y eficientes para poder ejecutarse en equipos viejos o de bajo rendimiento
- Interfaces de usuario fáciles de navegar e intuitivas de usar
- Escalabilidad para permitir la inclusión de un gran número de productos, vehículos y edificios sin impactar negativamente en el rendimiento.

## **3.4. Planificación del trabajo**



### 3.4.1. Cronograma

A continuación, se detalla un cronograma con la planificación propuesta para llevar a cabo las tareas que culminarán con la consecución de la propuesta realizada. Cada columna representa una semana, representada por su lunes para ubicarlas en el calendario. Las horas dedicadas de forma semanal se encuentran alrededor de las 30 durante los meses de abril y mayo, y llegan a ascender por encima de las 60 en las tres semanas de junio.

Tarea	22-abr	29-abr	6-may	13-may	20-may	27-may	3-jun	10-jun	17-jun
Formación Godot									
Práctica Godot									
Generación del mapa									
Pruebas									
Elementos del mapa									
Creación de clases									
Menús									
Interacción en el mapa									
Validación con usuario									

Tabla 2. Distribución temporal de las tareas. Elaboración propia.

El desarrollo comenzó con una concienzuda búsqueda de tutoriales en Internet y creación de juegos básicos para romper mano con el motor de juego. Una vez se tuvieron nociones básicas de su uso, se comenzó por programar la generación del mapa, y de cómo ubicar elementos sobre él a través de código. Con esto, se dio por acabada la primera etapa.

La segunda etapa consistió en la programación más propiamente dicha: se definieron las clases establecidas en el diagrama de clases, se crearon algunas estructuras de datos auxiliares para permitir la correcta gestión de, por ejemplo, los inventarios de los vehículos, y se generaron ficheros de recursos que establecían qué tipos de mercancías o modelos de vehículos existen.

Por último, la etapa más laboriosa, consistió en la creación de menús a partir de los mock-ups, y en la interacción de los elementos del mapa, tales como, por ejemplo, la carga y descarga en edificios, o la compra de vehículos.

### 3.4.2. Presupuesto

Contando con que durante seis semanas se trabajó a un ritmo de 30 horas semanales, y durante tres semanas la marcha de trabajo ascendió a las 60 horas semanales, se estima que, programando, se han invertido alrededor de 360 horas de trabajo.

Para calcular el coste por hora, se va a usar el valor que se percibió durante las prácticas, ya que el trabajo ha sido realizado por una persona sin experiencia previa, y podría considerarse labor de becario. Así, con un coste de 5€/h, el gasto en la mano de obra de este proyecto es de 1800€.

El resto del software utilizado es libre, o tiene versión gratuita, por lo que no existe coste de las herramientas software. Asimismo, el hardware utilizado data de 2014 y se puede considerar ya amortizado.

Por último, consideramos que no es necesario tener en cuenta el gasto en electricidad.

Por tanto, el coste total asciende a los 1800€.

## 4. Diseño

---

Dando por concluida la fase de análisis, y con la propuesta formulada, pasaremos a la etapa de diseño. En ella, plantearemos una serie de modelos conceptuales que servirán de guía a la hora de implementar el juego.

En primer lugar, dibujaremos el diagrama de clases, representando los diferentes elementos como abstracciones en código, y qué relaciones tienen entre sí, tales como la composición y la herencia.

En segundo lugar, realizaremos un diagrama de casos de uso. En él, plasmaremos las acciones que pueden realizar los usuarios y usuarias, qué acciones extienden o implementan, y cuáles son subacciones comunes a varias tareas diversas.

Por último, elaboraremos una serie de prototipos o *mock-ups* de las pantallas y menús necesarios para permitir que el jugador realice las acciones identificadas en el diagrama de casos de uso.

## 4.1. Diagrama de clases UML

A partir de la propuesta, se han identificado una serie de clases interrelacionadas. Ilustraremos este ecosistema de clases a través de un diagrama UML. Este diagrama ha sido creado usando la herramienta LucidChart. Los nombres de objetos, atributos y métodos se han traducido al inglés, pero las anotaciones se han mantenido en castellano por claridad.

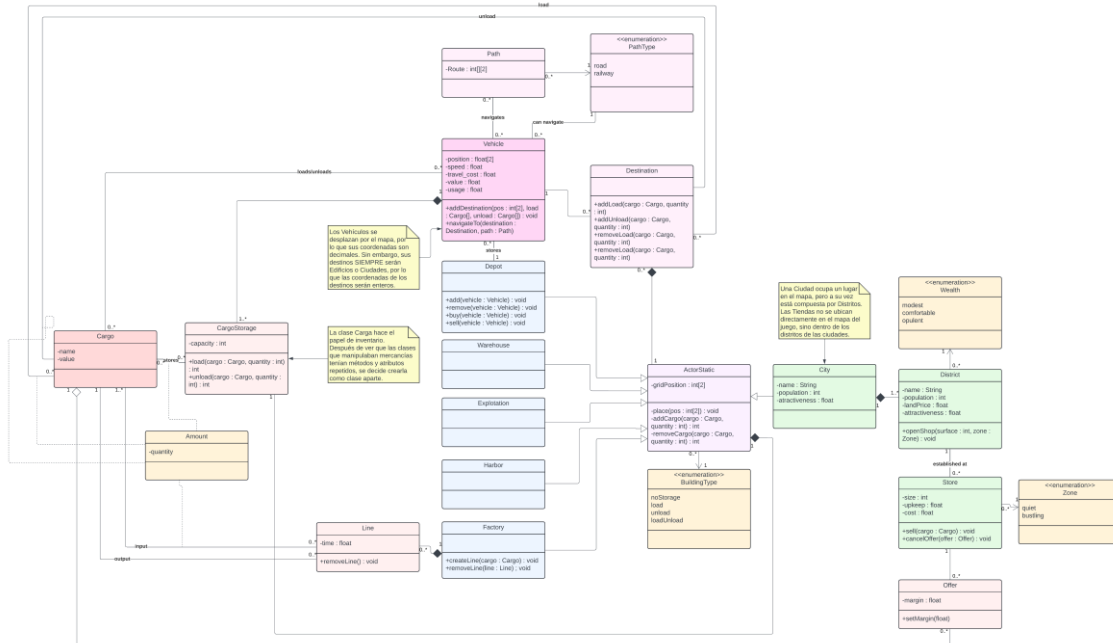


Ilustración 14. Diagrama de clases UML. Elaboración propia.

Podemos observar cuatro regiones destacadas por sus colores. En primer lugar, los cinco tipos de edificios, resaltados en azul: depósitos (Depot), almacenes (Warehouse), explotaciones (Exploitation), puertos (Harbor) y fábricas (Factory). Todas estas entidades heredan de Actor\_Static. Esta clase tiene como objetivo servir como base para todos los elementos estáticos del mapa, es decir, todos aquellos que no sean vehículos, y que tampoco sean carreteras ni vías de ferrocarril. La clase Actor\_Static incorpora, además, un elemento de tipo CargoStorage, que hace las veces de inventario de mercancía; implementa, además, los métodos addCargo y removeCargo, para añadir mercancía al inventario. Cabe destacar, sin embargo, que Depot no debería almacenar productos. Por simplicidad en el diagrama, se incluye como una clase hija más de Actor\_Static, cuya capacity para CargoStorage deberá ser de 0. Se ha añadido también el parámetro BuildType, precisamente para indicar qué tipo de edificio es, y si permite carga, descarga, ambas o ninguna.

La segunda región es la de ciudad (City), cuya familia ha sido marcada en verde. City desciende igualmente de Actor\_Static, por ser otro elemento estático en el mapa, y presenta la misma singularidad que Depot, ya que no debería poder almacenar mercancía. Una ciudad está compuesta por distritos (Districts), que pueden ser de tres tipos según el poder adquisitivo; en los distritos, se encuentran las tiendas (Store), en las que se pueden ofertar productos.

Esto nos lleva a la tercera región, la de producto o mercancía (Cargo), destacada de color salmón. Se trata de una de las clases angulares del esquema, relacionada con otros múltiples elementos. Muy frecuentemente, la mercancía viene acompañada de un número entero, representando las unidades de producto que se cargan, que se almacenan, que se venden, etc. Cargo, además, forma parte de CargoStorage que, como ya se adelantaba, hace las veces de inventario para edificios y vehículos.

Por último, nos encontramos con la clase vehículo (Vehicle). Se ha optado por no diferenciar entre camiones y trenes, sino definirlos como una clase genérica, con un atributo, PathType, que dicta en qué tipo de vía navegará el vehículo. A su vez, tenemos la clase vía (Path), que también recoge carreteras y vías férreas, diferenciándolas con el atributo PathType de nuevo. Las carreteras se representan con un array de posiciones enteras, que marcan las casillas por las que transcurre la carretera.

## 4.2. Diagrama de casos de uso

De la propuesta pueden extraerse, además de las clases, los casos de uso. Representaremos estos en un diagrama UML, creado con la herramienta LucidChart, y que podemos observar más abajo.

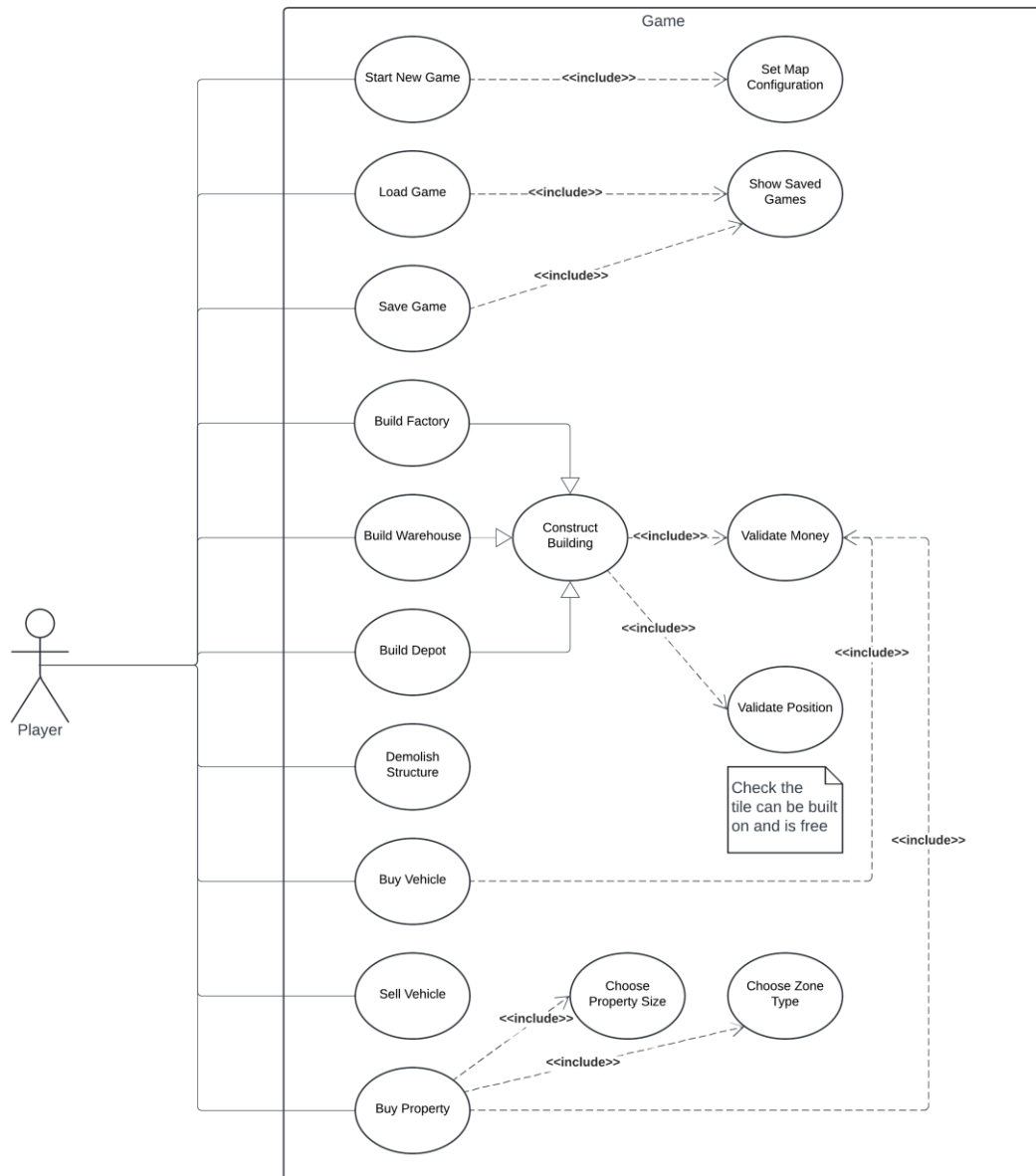


Ilustración 15. Diagrama de casos de uso, parte 1. Elaboración propia.



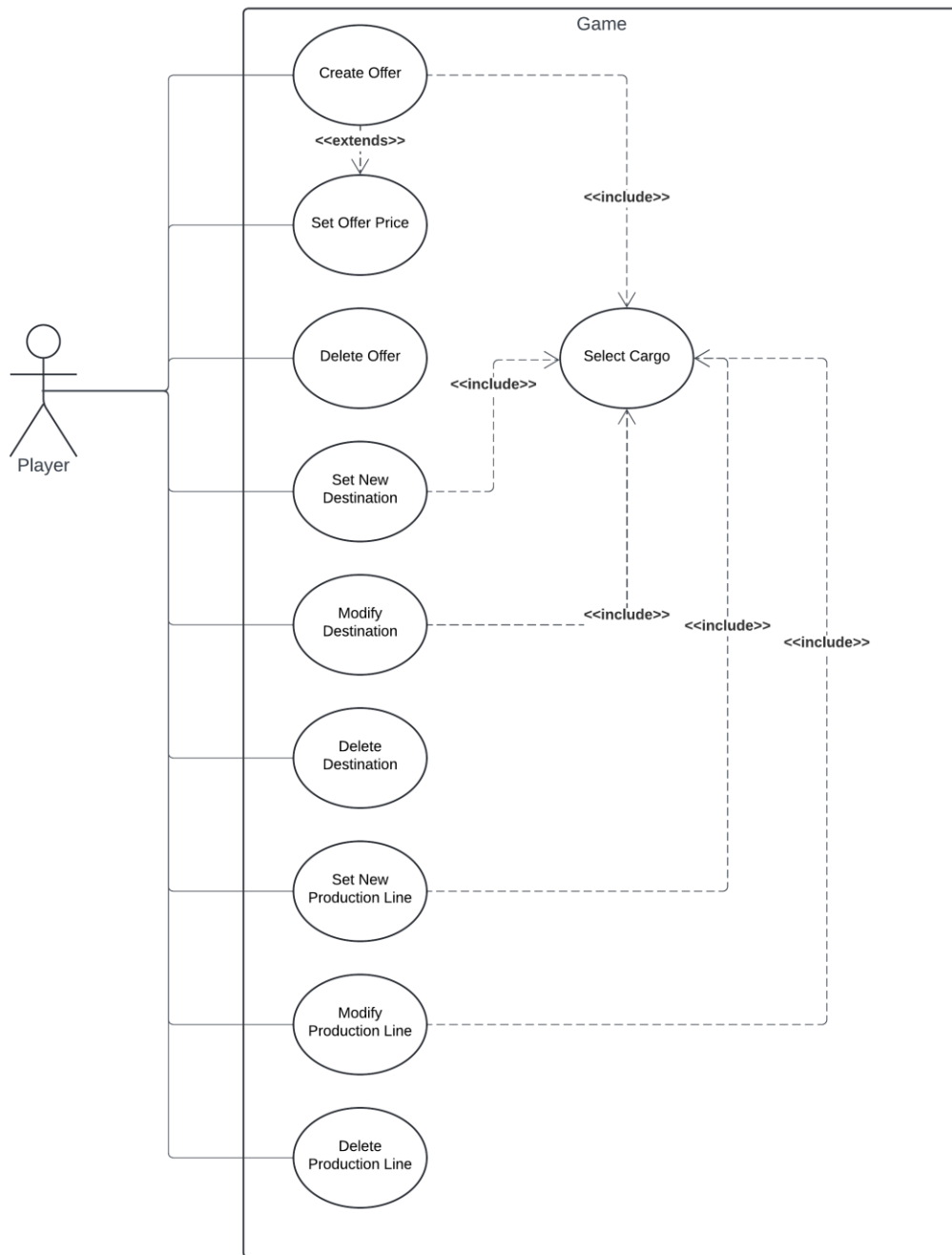


Ilustración 16. Diagrama de casos de uso, parte 2. Elaboración propia.

Los casos de uso se han dividido en dos diagramas, por comodidad a la hora de mostrarlos y de estudiarlos. A su vez, se han agrupado siguiendo un orden lógico.

Los tres primeros, los casos de uso de empezar un nuevo juego, cargar y guardar partida, tienen lugar fuera del transcurso del *gameplay*. Para empezar un nuevo juego, se debe seleccionar las características de la partida, tales como el tamaño del mapa o el número de elementos en este. Por otra parte, para poder guardar o cargar partida, se deben mostrar las partidas guardadas.

El segundo grupo de casos de uso es el de la construcción de edificios: construir fábricas, almacenes y depósitos. Todos ellos heredan de un caso de uso genérico, el de

construir edificio. Para realizar esta acción, se debe comprobar la disponibilidad de dinero, y que la casilla donde se va a construir sea válida, es decir, que no esté ni ocupada ni fuera del mapa. Relacionado con la construcción, tenemos otro caso de uso, el de demoler edificios.

A continuación, tenemos los casos de uso relacionados con los vehículos: comprar, para lo cual es necesario volver a comprobar los fondos disponibles, y venderlos.

El último caso de uso que requiere de la validación de dinero es el de comprar un local.

En el segundo diagrama, encontramos casos de uso sobre la configuración del tejido comercial, con acciones como crear oferta, que requiere seleccionar qué producto vender, y la posibilidad de cambiar el precio. También se permite eliminar ofertas.

La gestión de rutas se compone de tres casos de uso: seleccionar una nueva destinación y modificar el destino, ambas requiriendo elegir qué productos transportar; así como eliminar una parada de la ruta.

Por último, se pueden crear y modificar líneas productivas, seleccionando la mercancía que se obtendrá como *output*, así como eliminarlas.



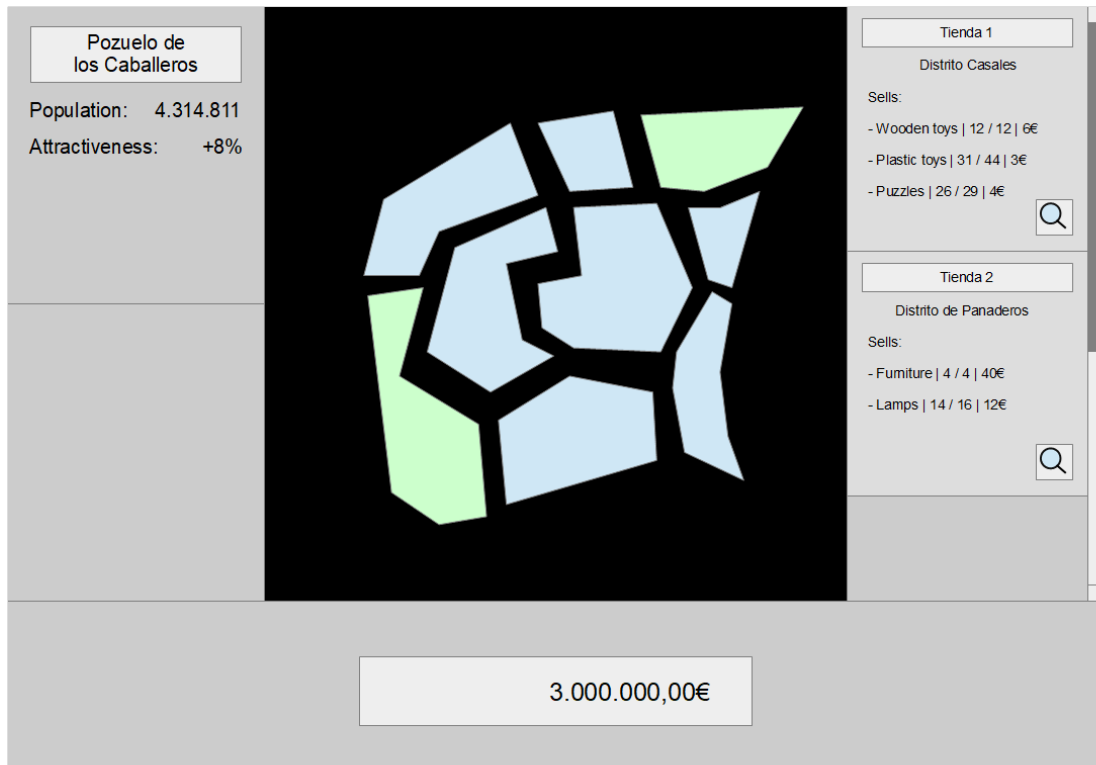


Ilustración 18. Mock-up del mapa de una ciudad. Elaboración propia.

En la siguiente pantalla, se observa el menú que permite gestionar las tiendas, añadiendo, modificando o eliminando ofertas. Además, en la parte superior se muestran las características de la tienda, a la izquierda información de la ciudad y del distrito, y a la derecha qué otras tiendas existen en el mismo distrito.

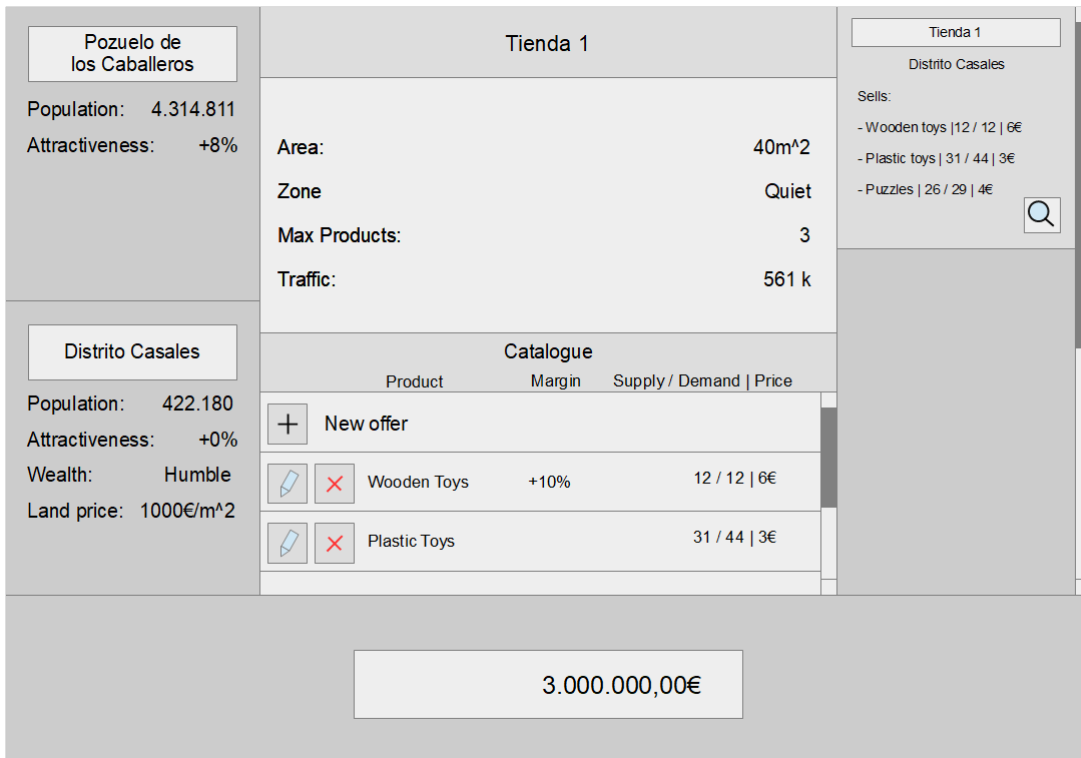


Ilustración 19. Mock-up del menú de las tiendas. Elaboración propia.

A continuación, pasamos al menú de las fábricas, en el que podemos gestionar las líneas de producción.

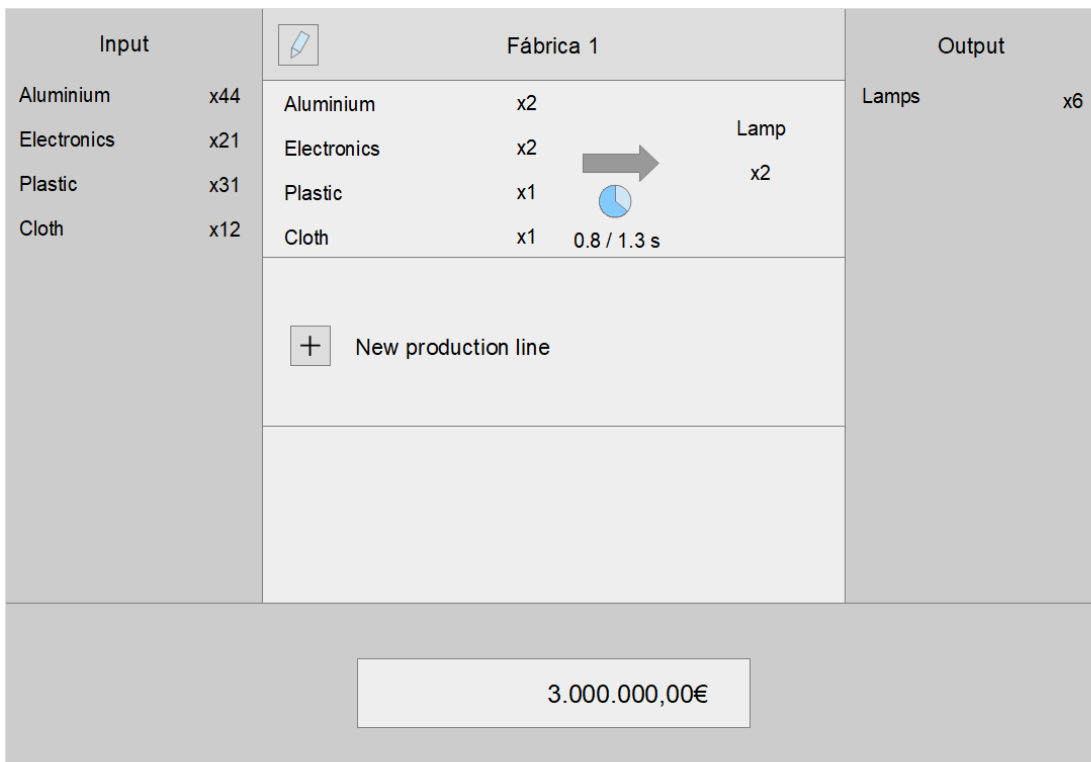


Ilustración 20. Mock-up de la pantalla de las fábricas. Elaboración propia.

Otra pantalla importante es la de los depósitos de vehículos. En ella, podemos visualizar nuestra flota actual, aumentarla, reducirla, o acceder al menú para modificar las rutas de un vehículo concreto.

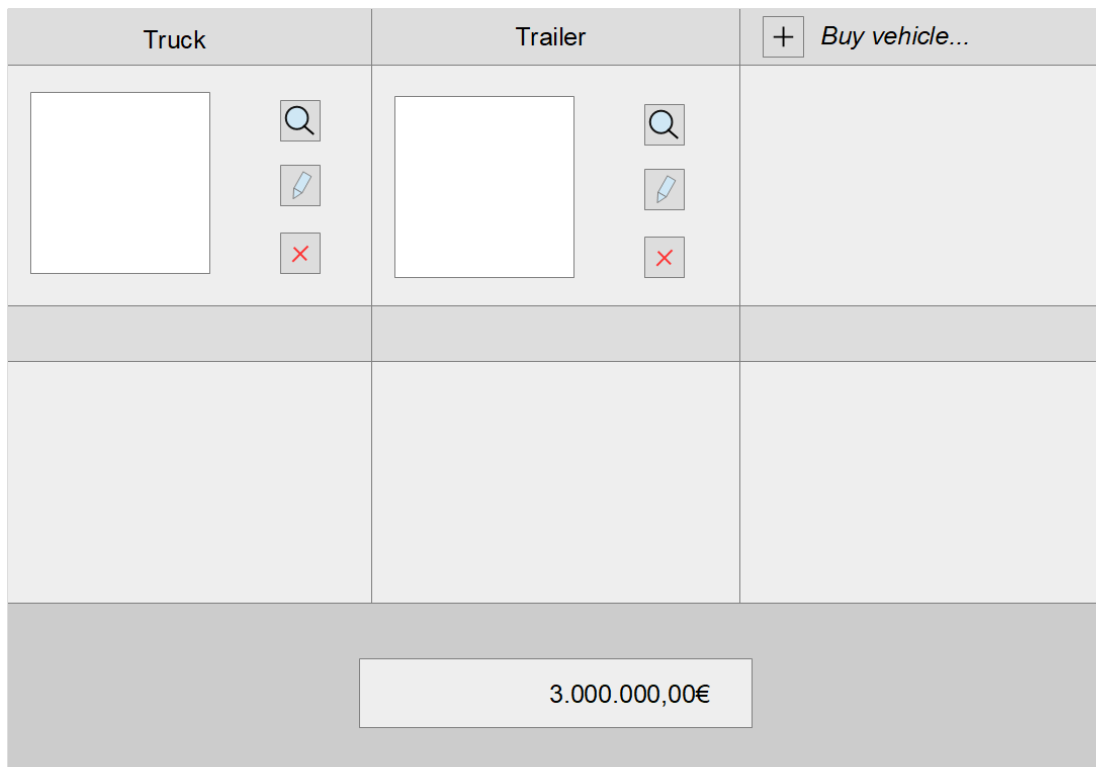


Ilustración 21. Mock-up de la pantalla de los depósitos. Elaboración propia.

Por último, una de las pantallas más complejas e importantes, la de selección y configuración de las rutas. Desde esta, podemos elegir una nueva destinación, editar una ya existente, o eliminarla, así como establecer qué mercancías se cargan y descargan en cada parada.

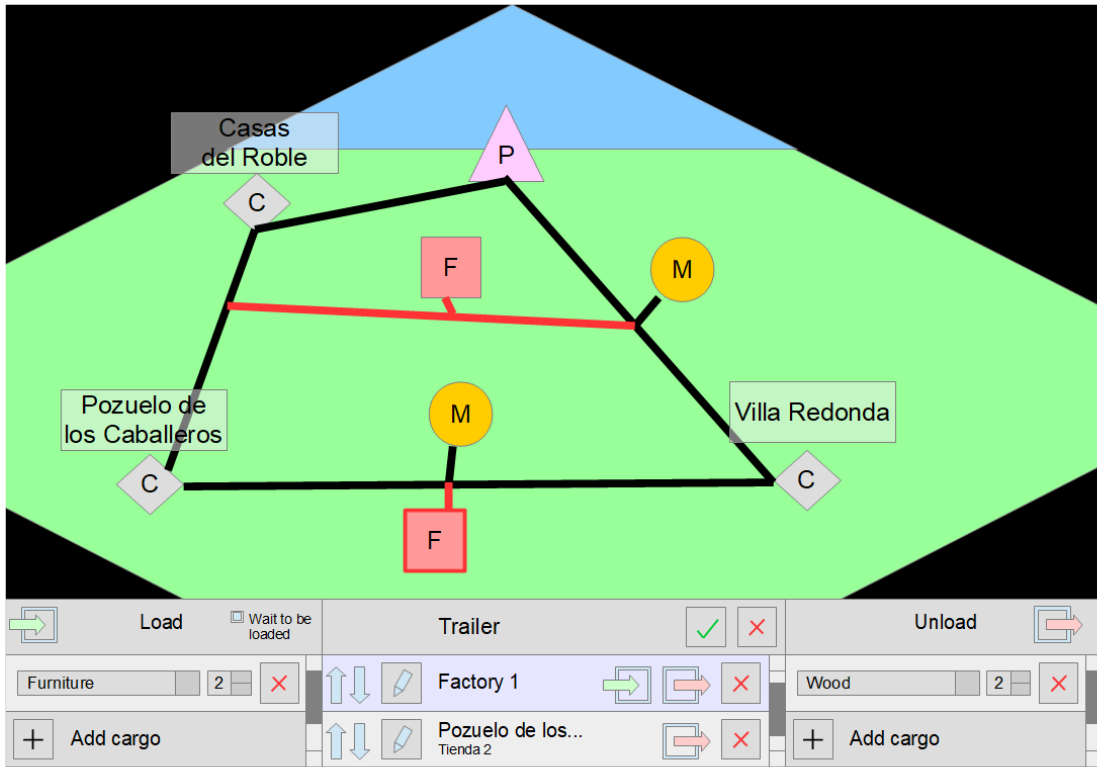


Ilustración 22. Mock-up de la pantalla de rutas. Elaboración propia.

## 4.4. Tecnologías empleadas

Como se adelantó en la sección del estado del arte, la herramienta principal para implementar la propuesta es el motor de juego Godot. Se trata de software abierto, altamente personalizable y de uso libre.

En los proyectos de desarrollo, suele ser muy recomendable utilizar herramientas de control de versiones. Aunque existen diversas, como *Subversion (svn)* utilizada durante las prácticas de empresa, en este trabajo se ha optado por *Git*.

Para la gestión del proyecto, y un uso más ágil del control de versiones escogidos, se ha usado *GitHub Desktop*.

Para la elaboración del arte, se ha empleado la herramienta *Piskel*. De nuevo, es un programa de software libre, sencilla pero completa para la elaboración de arte en bajas resoluciones (*pixel-art*). Otras alternativas que se tuvieron en cuenta fueron *Gimp* y *OpenOffice Draw*.

Todo el arte es de elaboración propia.

Aunque no se llegó a implementar, se consideró la incorporación de música de fondo al juego. La música sería creada usando, una vez más, software libre; en concreto, se habría usado el programa *Linux Multimedia Studio*, conocido por su abreviatura *LMMS*.

Los diagramas de clases y casos de uso se han elaborado usando la herramienta *LucidChart*.



## 5. Desarrollo

---

El desarrollo de un videojuego, aunque similar en rasgos generales a otros proyectos software, ofrece una serie de desafíos únicos para los que se ha tenido que desarrollar una serie de destrezas específicas.

En primer lugar, la representación visual de los elementos en el mapa supone tener en cuenta unas consideraciones que en otras aplicaciones tienen un peso secundario o nulo. Algunos ejemplos son las colisiones, la navegabilidad (*pathfinding*) o el posicionamiento de elementos, principalmente las carreteras, en la cuadrícula.

El lenguaje de programación de Godot, GDScript, toma elementos de Python y C#, pero ofrece numerosas particularidades. En concreto, no existen restricciones sobre las variables o funciones privadas, sino que se utiliza un convenio en su nomenclatura para demarcarlas como tal. Además, la herencia presenta a su vez una serie de restricciones vinculadas a su propósito de programación de videojuegos que han influido en su implementación.

La falta de experiencia en programación de videojuegos ha supuesto una serie de obstáculos que han impedido la implementación deseada, y se han tenido que optar por otras implementaciones alternativas. En ningún caso se ha dejado ninguna implementación pendiente por falta de conocimiento, aunque algunas son propensas de mejoras u optimización.

El motivo por el que se han tenido que cancelar algunas características en la propuesta ha sido el tiempo. La falta de experiencia en este tipo de desarrollos, la inexperiencia con la herramienta, y un alcance definido, tal vez, de forma poco realista, han llevado a la ausencia de algunas funcionalidades. La más notable, la ausencia de distritos y tiendas. También se han tenido que adaptar algunas características conforme el proyecto avanzaba, por falta de tiempo.



## 5.1. Diagrama implementado

Como se ha introducido, se han tenido que realizar una serie de adaptaciones de la propuesta inicial, motivadas por el límite temporal y la inexperiencia con el motor. A continuación, se muestra el diagrama de clases UML que finalmente se ha implementado, y que ha sido elaborado, de nuevo, con LucidChart.

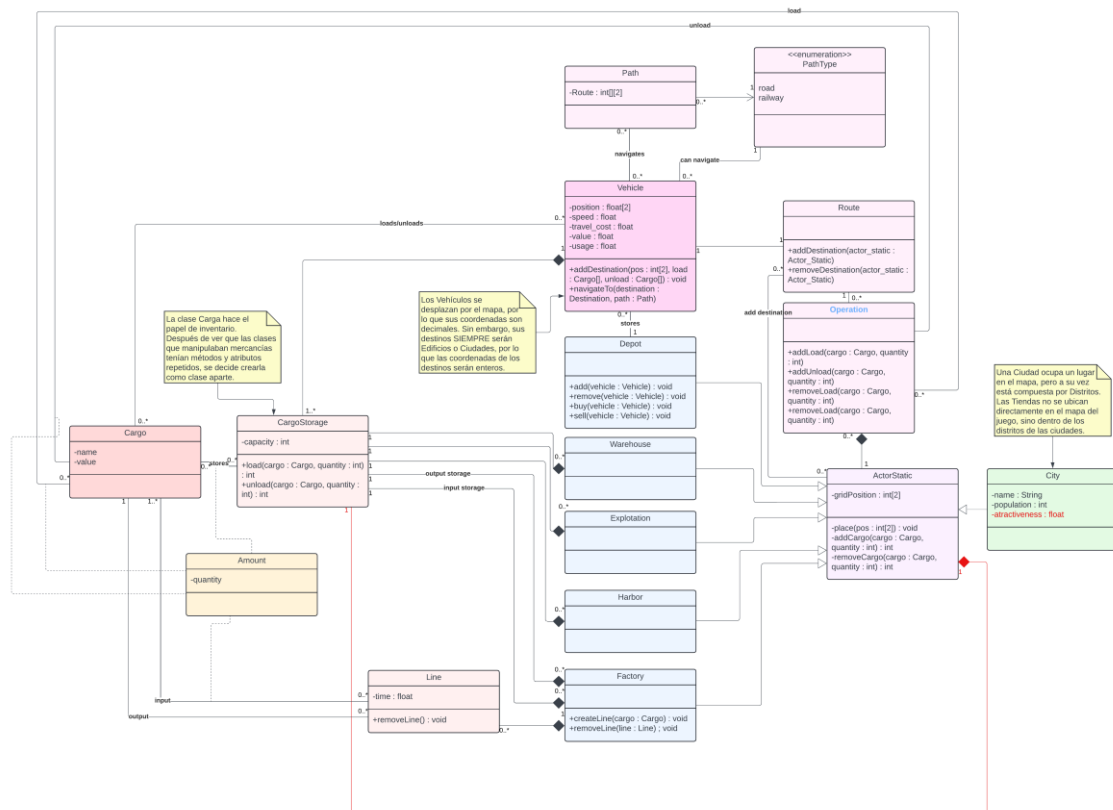


Ilustración 23. Diagrama de clases UML implementado. Elaboración propia.

El cambio más significativo es la ausencia de distritos, tiendas y ofertas de las tiendas. Se tomó la decisión de no implementar estas clases y su funcionalidad asociada durante las dos últimas semanas del proyecto, optando en su lugar por invertir el tiempo en pulir otras características básicas, reorganizar algunos menús para hacerlos más fáciles de navegar, y en optimizar y limpiar fragmentos de código, de forma que fueran más legibles y flexibles en caso de que se retomara el proyecto en un futuro y se desearan añadir nuevas características.

Otro cambio destacable es el de `ActorStatic`. Se ha eliminado su elemento de `CargoStorage`, y en su lugar se ha optado por implementar esta relación para cada uno de los edificios que lo usan. De esta forma, se evita la falsa herencia por parte de `Depot`, que no contaba con un inventario real, y se permite separar la capacidad de los inventarios de entrada y de salida de las fábricas. Esta separación permite que las fábricas puedan seguir recibiendo y almacenando materias primas aunque su almacén de productos terminados esté lleno; se ha considerado que esto era menos restrictivo para el jugador o la jugadora.

Un último cambio al que merece la pena hacer mención es el de la creación de la clase Route. Esta clase tiene como finalidad permitir un nivel de abstracción para Vehicle, y que ya no tiene la responsabilidad de gestionar las operaciones que ocurren en su ruta. Las operaciones de inserción, reordenación y de eliminación ahora las gestiona la clase ruta. Además, se ha renombrado la clase Destination por Operation. Aunque ambos nombres son acertados en diferentes contextos para la misma clase, se ha decidido que Operation tenía un significado más amplio y definía mejor la clase.



## 5.2. Estructura del proyecto

Para mantener el código organizado, se ha utilizado una estructura de carpetas para separar los diferentes elementos.

En el directorio raíz, se crearon tres carpetas: Assets, para el arte usado; Nodes, con el grueso del código y nodos de Godot usados; Scripts, en la que se encuentran scripts que no se vinculan a nodos; y Resources, con los ficheros en los que se podía encontrar la información de las instancias de las clases como Cargo y ProductionLine.

Dentro de Assets, se encuentran otras cinco carpetas: Buildings, con los *sprites* para los diferentes elementos del mapa, y una subcarpeta Explotations, con los diferentes *sprites* de los edificios de materias primas; Cargo, con los iconos de los diferentes tipos de mercancías; GUI\_Icons, con los iconos usados en interfaces, tales como lupas, cruces rojas, y flechas; TileSets, con las baldosas usadas para el mapa; y Vehicle\_Models, con los *sprites* para los diferentes tipos de vehículos.

En la carpeta Nodes, se encuentran la mayoría de nodos de Godot, así como sus scripts. Por la forma en que los scripts se vinculan a los nodos, se decidió no separar estos elementos en dos carpetas diferentes, para no aumentar la complejidad y evitar que los vínculos pudieran generar problemas. Dentro de Nodes, hay una subcarpeta, Menus. Dentro de ella, se encuentran los nodos y scripts de los diferentes menús que se usan. Debido a la forma en que están implementados, existen muchos submenús, representados por nodos aparte, por lo que se consideró separar todos estos nodos gráficos en su propio directorio.

En la carpeta Scripts se encuentran los scripts de clases no vinculadas a nodos, así como un fichero con constantes, otro con métodos globales usados por diferentes scripts, y el script encargado de crear los catálogos de mercancías y líneas de producción.

Por último, en la carpeta Resources, se guardan los catálogos con las diferentes instancias de Cargo y ProductionLine.

## 5.3. Adaptaciones al motor

Algunos de los cambios en la implementación se deben al motor de juego, que presenta algunas particularidades que condicionaron el desarrollo.

En primer lugar, su lenguaje de programación, GDScript, que incorpora elementos de Python y C#, destacó al inicio por no permitir la declaración de atributos ni métodos privados. En su lugar, existe la convención de nombrarlos comenzando con una barra baja ('\_') para denotar que no deben ser accedidos desde otras clases. La función de autocompletado (*Intellisense*) de Godot tendía a no mostrar sugerencias denotadas con este convenio, aunque se dieron ocasiones en las que sí. A pesar de todo, a lo largo de proyecto, se ha respetado esta nomenclatura.

Otra particularidad, que complicó notablemente el inicio del desarrollo, es el uso de las clases. El bloque de construcción en Godot es el Nodo. Existe un amplio abanico de estos en Godot, que se pueden especializar en función de su uso. Todos ellos heredan del nodo base Node, y se ramifican en función de si su uso será bidimensional, tridimensional, como elementos de interfaz, etc. Por ejemplo, los edificios en el mapa son StaticBody2D. Estos, a su vez, heredan de Node2D, que es descendiente directo de Node.

Los nodos pueden ser configurados a través del editor, pero no se pueden modificar por código directamente. Para incorporar código a un nodo, es necesario asociarle un script. El script debe heredar el mismo tipo de nodo al que está asociado. Cuando se define un `class_name` para un script, pasa a considerarse una clase.

Las clases se pueden instanciar usando la instrucción `.new()`, mientras que los nodos usan `.instantiate()` y deben incorporarse a la escena usando `.add_child()`. Esta distinción supuso cierta confusión al inicio, ya que, por ejemplo, crear una ciudad usando `City.new()` no permite ubicarla en el mapa. Los nombres de clases para nodos son útiles, entre otras cosas, para filtrar por tipo los hijos de un nodo. Por ejemplo:

```
for child in destinations_container.get_children():
    if child is DestinationMenu:
        child.queue_free()
```

Otro ejemplo: adición de una nueva línea de producción en el menú de las fábricas. Notar el uso de la variable privada `_index`:

```
func add_line(production_line : ProductionLine = null):
    var line_menu = LINE_MENU.instantiate()
    lines_container.get_child(_index).add_child(line_menu)
    line_menu.initialize(production_line)
    _index += 1
```



Este bucle sirve para vaciar los destinos en el menú de las rutas. Destinations\_container es un contenedor de tipo caja vertical (VBox) que contiene, aparte de los submenús con cada destino, un recuadro al final que explica cómo añadir nuevos destinos. Darles un nombre a los submenús permite que este bucle sea más sencillo de programar, y más intuitivo de leer, ya que se identifica claramente qué se está borrando. Así, variaciones de este bucle han sido utilizadas de forma extensiva en diferentes menús, en los que vaciar un contenedor de todos sus hijos no era deseable, sino que se requería de cierta habilidad para discriminarlos.

Esta es otra característica que merece la pena comentar. Los nodos de Godot presentan un alto grado de modularidad; y aunque los menús parezcan especialmente pensados para personalizarse usando el editor, sin necesidad de código, se ha optado por crear submenús, como el expuesto anteriormente para los destinos. De esta forma, tenemos la libertad de personalizar libremente el componente, siempre teniendo en cuenta las proporciones en las que se mostrará cuando se use, e insertar un número indefinido de copias del submenú en los contenedores.

Para las clases que no representan elementos del mapa, se crearon scripts aislados que heredan de Object directamente. Estas clases sí que se pueden instanciar con .new() de forma directa. Sin embargo, para algunas clases, tales como Cargo o ProductionLine, se deseaba que fueran consistentes entre sus instancias. Esto es, que todas las mercancías con nombre Plastic, tuvieran el mismo valor, y la misma imagen asociada.

La forma deseable de programar esta funcionalidad sería haciendo consultas a una base de datos, con una tabla que tuviera un registro por cada tipo de mercancía, y una columna para cada atributo, tales como el nombre, coste, imagen asociada, etc.

Sin embargo, Godot no ofrece esta opción. Como alternativa, permite la creación de recursos, que son ficheros que definen características y que se pueden consultar. Para la implementación de estos, era necesaria la creación de scripts que, en lugar de Object, heredaran de Resource; la creación de scripts cuya única funcionalidad es la de servir para almacenar las diferentes instancias de la clase (llamados en el proyecto como Catalogs, o catálogos), con métodos para su consulta, y que también debe de heredar de Resource; y un script que instancia las clases, les asigna un id, inicializa los diferentes parámetros, almacena cada instancia en su Catalog, y guarda el Catalog en un fichero .tres (text resource, o recurso en texto).

Declaración de un tipo de mercancía:

```
var catalog = CargoCatalog.new()

var cargo

cargo = Cargo.new()

cargo.id = 1

cargo.name = "Wood"

cargo.value = 275
```

```

cargo.img_path = "res://Assets/Cargo/"+cargo.name.to_lower()+".png"

catalog.cargos[cargo.id] = cargo

[...]

ResourceSaver.save(catalog, Constants.CARGO_CATALOG_PATH)

```

De esta forma, se pueden definir Cargo y ProductionLine en un único lugar, con todos los tipos de mercancías y líneas de producción, y posteriormente usarse en otros lugares sin que cambios en el catálogo afecten a otros lugares en el código.

Por último, otra particularidad de Godot es la herencia. No permite crear clases ni métodos abstractos. Toda clase debe implementar los métodos o funciones que use, y aquellas que los hereden, pueden sobrescribir los métodos de su clase padre.

En foros de Internet se aconseja, para métodos que se desea dejar sin implementar, que su contenido lance un error. De esta forma, se puede detectar cuándo una clase no ha implementado un método concreto, si bien requiere que se pruebe el código ejecutándolo.

La herencia se ha usado en Actor\_Static. Se definieron los métodos load() y unload(), cuya implementación en la clase padre es la de devolver un -1. Aprovechando esto, se ha añadido un condicional a los métodos de añadir o eliminar mercancía de los vehículos: si reciben un -1 en la primera operación que realice, cancela el resto de operaciones de carga o de descarga.

A continuación, se observan los métodos de carga y descarga de la clase padre Actor\_Static:

```

func unload_cargo(cargo : Cargo, quantity : int) -> int:

    print_debug("This building does not accept cargo: " + str(self.get_class()))

    return -1

func load_cargo(cargo : Cargo, quantity : int) -> int:

    print_debug("This building does not offer cargo: " + str(self.get_class()))

    return -1

```

El bucle de descarga de un vehículo. Notar cómo primero comprueba si tiene suficiente producto para descargar en su inventario, a continuación hace la petición de descarga y, si no recibe un -1 como resultado, elimina la cantidad descargada de su inventario, además de tacharla de la lista de descargas en ese destino.



```
var unload_dict = current_operation.get_unload_duplicate()

for cargo in unload_dict:

    var quantity: int = min(unload_dict[cargo], cargo_storage.get_quantity(cargo))

    var unloaded : int = destination.unload_cargo(cargo, quantity)

    if unloaded == -1: break

    cargo_storage.remove_cargo(cargo, unloaded)

    unload_dict[cargo] = quantity - unloaded
```

Implementación de carga y descarga en una fábrica, que acepta ambas operaciones. Notar que no es necesaria ninguna acción especial para sobrescribir los métodos de Actor\_Static.

```
func unload_cargo(cargo : Cargo, quantity : int) -> int:

    return input_storage.add_cargo(cargo, quantity)

func load_cargo(cargo : Cargo, quantity : int) -> int:

    return output_storage.remove_cargo(cargo, quantity)
```



## 5.4. Decisiones tomadas

Con motivo de algunos obstáculos encontrados, y de restricciones temporales, se han tenido que tomar una serie de decisiones sobre cómo afrontarlos.

### 5.4.1. Selección de elementos

El primer gran obstáculo fue la incapacidad de seleccionar edificios en el mapa usando sus elementos *CollisionShape*. Godot ofrece tres tipos de actores en elementos 2D: *StaticBody2D*, *PhysicsBody2D* y *CharacterBody2D*. Todos ellos tienen en común que pueden asociarse una imagen (*sprite*) y una caja de colisión (*hitbox* o *collision shape*). Están preparados, además, para ser fácilmente seleccionados si tienen una caja de colisión, marcando una casilla en el editor de Godot. De esta forma, el elemento envía una señal que puede ser conectada a un método que la gestione.

La implementación original detectaba esta señal exitosamente. Se creaba un mapa aleatorio con elementos ubicados en posiciones arbitrarias y, cuando se hacía clic en uno de ellos, recibía la señal del evento de entrada (*input*) y, a su vez, enviaba una señal con su propia identidad al mapa, configurado como nodo padre.

El problema apareció cuando se incorporó el mapa a la jerarquía de nodos que representa la pantalla principal, con los diferentes menús. Para mantener la funcionalidad del mapa sin que interfiriera de manera no deseada con los menús, se lo colocó en un componente llamado *SubViewport*. Sin embargo, esta implementación causó que los edificios del mapa no detectaran cuándo se hacía clic en ellos.

Se hizo una serie de pruebas. Se descubrió que el mapa sí que detectaba los clics, y que el elemento que estaba impidiendo el correcto funcionamiento era el *SubViewport*. Se consultó la documentación disponible, así como algunos tutoriales disponibles en *YouTube*; incluso se recurrió a un foro de ayuda de Godot, del cual no se obtuvo respuesta. Con todo, se experimentó con la configuración de los elementos, pero no se consiguió ningún cambio significativo.

Finalmente, se estudiaron dos alternativas. Por una parte, cabía la posibilidad de eliminar el componente *SubViewport*, con los problemas que ello podría traer, principalmente en detección de *inputs* y en los límites que usaba la cámara para detectar que debía moverse, y que deberían configurarse de forma manual.

La otra alternativa, y la que finalmente se escogió, fue usar la posición en el mapa clicada para encontrar el edificio que se deseaba seleccionar. Esto acarrea el problema de que, en perspectiva isométrica, casi la mitad del edificio aparece representado fuera de la casilla sobre la que se asienta, por lo que el jugador debe identificar esta peculiaridad y tener cierta precisión para clicar en la casilla deseada. Otra consecuencia de esta decisión fue que no se pudieran seleccionar camiones en el mapa; en su lugar, se delegó esta funcionalidad a los depósitos de vehículos.

### 5.4.2. Navegación

Otro problema acaecido fue el de la navegación (*pathfinding*). En el proceso de aprendizaje de Godot, se crearon juegos de prueba en los que la navegación se realizaba configurando qué tipo de casillas eran navegables, e incorporando un elemento de `NavigationAgent2D` al actor que debía recorrerlas.

Esta aproximación, sin embargo, no funcionó. En el juego final, se había optado por emplear diferentes capas en el mapa, destinadas al terreno, a las vías (carreteras y vías férreas), la previsualización de vías (para evitar que los vehículos intentaran navegar sobre vistas previas de carreteras), y otros actores. Además, se definieron diferentes capas de navegación, diferentes de las capas genéricas, una para carreteras, otra para ferrocarriles y otra para ambas. Esta última capa de navegación estaba destinada a ser usada por los edificios, de forma que fueran accesibles tanto por carretera como por ferrocarril.

Esta organización, si bien necesaria, resultó compleja. Cuando se intentó implementar la navegación de la misma forma que en los juegos de prueba, los camiones se quedaban atascados.

Se probó a eliminar las colisiones de edificios y camiones, ya que, por el problema resuelto anteriormente, habían dejado de ser necesarias. También se experimentó a dibujar una gran área de carreteras alrededor de los depósitos. No obstante, no se conseguía lograr que se movieran exitosamente por las carreteras, y se identificó que las destinaciones cambiaban a cada instante, “botando” de una esquina a otra de la casilla en la que se quedaban atascados.

Se creó una escena de prueba, cambiando la capa en la que los camiones se ubicaban, la capa en la que las carreteras se construían, y qué forma tenían las vías. Colocando las carreteras en la capa 0, la capa base, los camiones lograban navegarlas exitosamente. Sin embargo, no se logró encontrar qué configuración era necesario cambiar para adaptar el *pathfinding* a otras capas. Posteriores experimentos en esta escena de prueba arrojaron resultados confusos y que no se supieron interpretar.

Después de dos días, con más de quince horas invertidas en este problema, se optó por cambiar la forma en que los vehículos se movían por el mapa. Se consideró la opción de que se movieran libremente por el mapa, pero por restricciones temporales, finalmente se eligió que se “teletransportaran” de una ubicación a otra. Ambas soluciones suponían la pérdida de funcionalidad de las carreteras, con la consiguiente confusión y posible frustración de los jugadores y jugadoras. Sin embargo, se juzgó que era más deseable continuar con la implementación de otras características que seguir invirtiendo tiempo en este problema.

Para tener en cuenta la velocidad del vehículo, se usó un periodo de espera (*cooldown*) entre teletransporte y teletransporte. Este periodo se calculaba de forma inversamente proporcional a la velocidad del vehículo. De esta forma, se seguía utilizando, sin tener que cambiar su implementación.

### 5.4.3. Distritos y tiendas

Durante las últimas semanas de desarrollo, se detectó la posibilidad de que no hubiera el tiempo suficiente para implementar los mapas de distritos y las tiendas. Se planteó la opción de programarlos de forma básica y apresurada, pero finalmente se optó por invertir los últimos días en pulir algunos aspectos importantes.

Se cambiaron algunos valores por defecto en algunos menús, tales como los de selección de mercancía y cantidad; se ajustó la sensibilidad de la cámara a la hora de desplazarse sobre el mapa, si bien no se alcanzó el umbral deseado; se añadieron descripciones emergentes (*tooltips*) a la mayoría de botones, para especificar su uso; se rehicieron varios iconos para sustituir los provisionales (*placeholders*) que habían ocupado su lugar durante el desarrollo; y se reorganizó y limpió el código en algunas secciones en las que parecía propenso a comportamientos no deseados, concretamente la parte de selección de rutas.

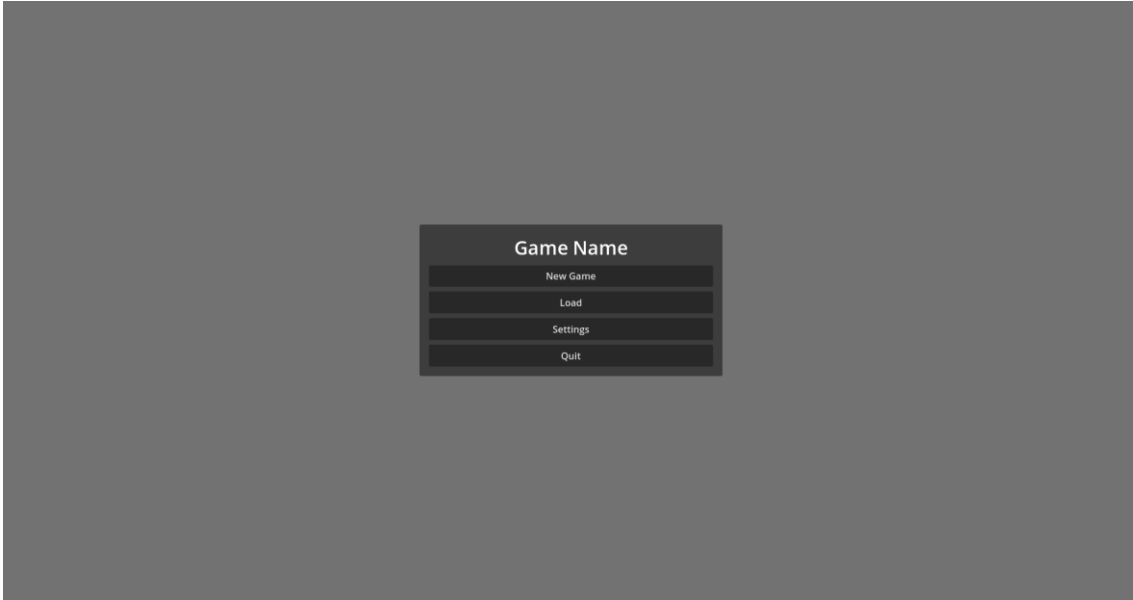
Dada la nueva funcionalidad de las tiendas, los puertos se consideraron redundantes, así que, aunque están implementados parcialmente, se han configurado para que no puedan aparecer en el mapa.



## 5.5. Aspecto final

A continuación, mostraremos algunas pantallas del videojuego resultante.

En primer lugar, tenemos la pantalla del menú principal y de selección del mapa.



*Ilustración 24. Menú principal.*



*Ilustración 25. Pantalla de selección de mapa.*

A continuación, podemos echar un vistazo a cómo ha quedado el mapa y los menús que aparecen en la parte inferior. Notar que no se han implementado botones para la construcción de almacenes ni depósitos de ferrocarriles, ni para guardar ni cargar partida, ni para cambiar los ajustes.

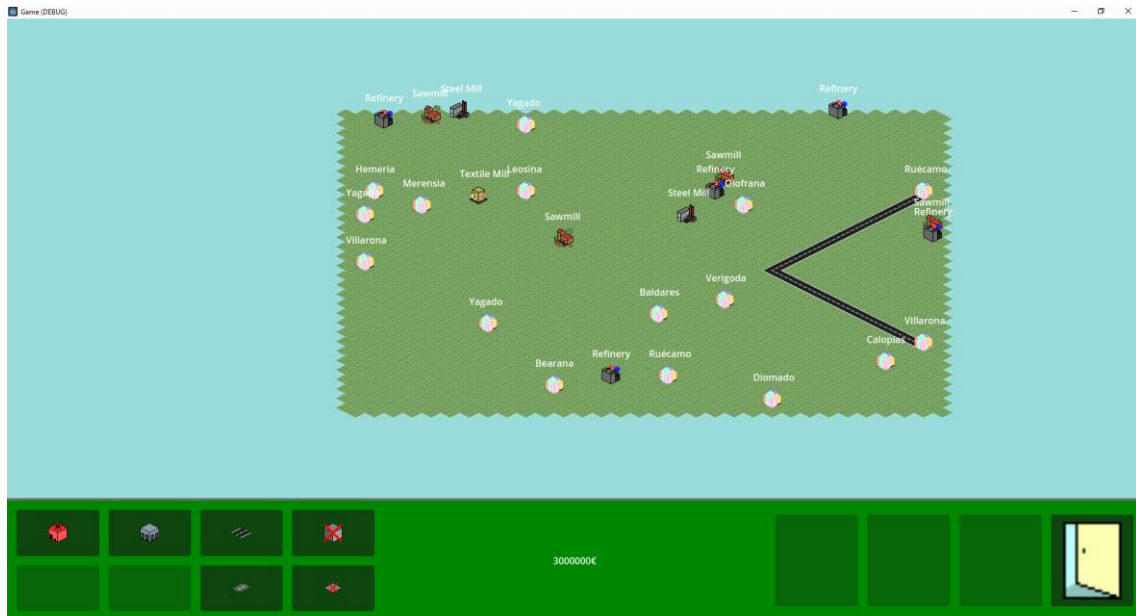


Ilustración 26. Mapa del juego.

Tenemos, también, el menú de los depósitos de vehículos, cuyo aspecto final es marcadamente similar al de su prototipo inicial. Desde este menú, se pueden comprar y vender camiones, así como acceder al menú para modificar sus rutas.

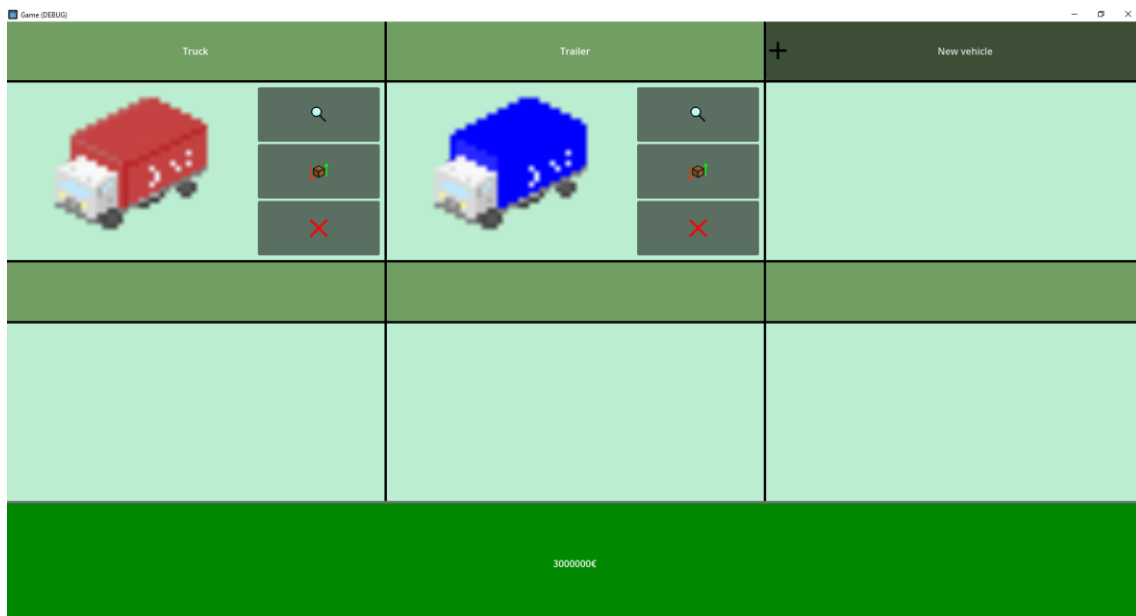


Ilustración 27. Menú de los depósitos de vehículos.

## Diseño y desarrollo de un videojuego de gestión empresarial y logística usando el motor de juegos Godot.

Es interesante ver también el menú de las fábricas. A ambos lados se cuenta con un espacio para enumerar los materiales en el almacén, y la sección central queda reservada para las líneas de producción, que pueden ser añadidas y eliminadas desde este menú. Notar cómo la primera línea de producción se encuentra destinada a los muebles y cómo, en el segundo espacio, el jugador se encuentra en proceso de crear una nueva línea, a punto de seleccionar qué bienes desea producir.

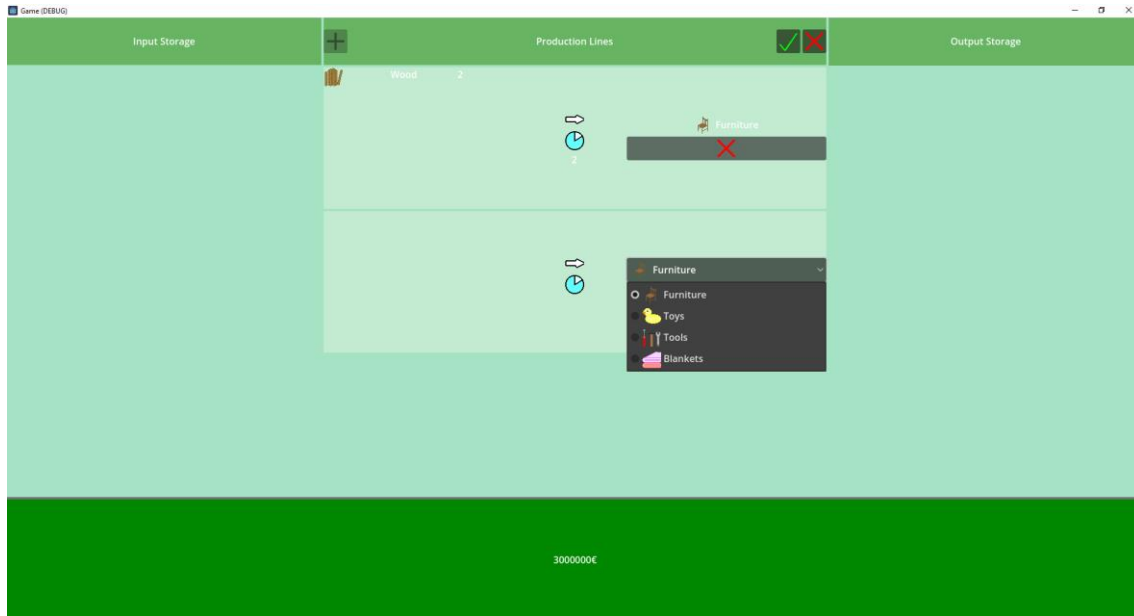


Ilustración 28. Menú de las fábricas.

Por último, nos gustaría mostrar el menú de la edición de rutas, que tiene como misión posibilitar la adición y eliminación de destinos de una ruta, así como definir qué operaciones de carga y descarga se realizarán en cada destino. Notar que en el ejemplo de la imagen, el camión tiene tres destinos asignados, y se está editando qué productos se cargarán en el segundo.



Ilustración 29. Menú de edición de rutas.

## 5.6. Implantación

Una vez finalizado el juego, después de pulir y dar los últimos repasos, se exportó a un fichero ejecutable.

Godot ofrece múltiples opciones de exportación para diferentes sistemas operativos: Windows, Linux, Mac, Android, iOS, e incluso para ser ejecutado en navegadores web.

Godot tiene la particularidad de no descargar por defecto las plantillas para la exportación, sino que deben ser descargadas en el momento en el que se finaliza el juego.

Este detalle obligó al visualizado de un último vídeo tutorial, de los muchos requeridos en este proyecto, para exportar exitosamente el juego. Después de tres intentos, se obtuvo el fichero .exe que abría el juego.

Este fichero se subió a un espacio de Google Drive, y se cambió su visibilidad a pública, para que pudiera ser probado por los usuarios y usuarias.



## 6. Validación con usuarios

---

Para conocer la recepción de los usuarios y usuarias, se preparó una encuesta usando la herramienta Google Forms. En ella, se adjuntó un enlace al juego, subido a un espacio de Google Drive, para que pudieran descargar el juego.

### 6.1. Preguntas de la encuesta

La encuesta se encuentra dividida en un total de seis secciones.

La primera de ellas contiene una breve introducción sobre el objetivo del estudio, las características del juego, cómo descargar la versión de prueba y cómo rellenar la encuesta. Esta parte no contiene ninguna pregunta.

La segunda sección sirve de introducción, y cuenta con dos preguntas sencillas para romper el hielo. La primera, inquiriere sobre el dominio con las nuevas tecnologías, para conocer si los problemas al navegar la interfaz más adelante pudieran estar relacionados con esta falta de conocimiento. La segunda, pregunta sobre la frecuencia de uso de videojuegos.

Ambas preguntas son de escala Likert, con valores del 1 al 5. En la primera, sobre el dominio de nuevas tecnologías, el 1 equivale a “me cuestan mucho” y el 5, a “las uso con naturalidad”. En la segunda, el 1 representa “casi nunca” y el 5, “todas las semanas”.

La siguiente sección, la tercera, invita al encuestado a probar nuestro juego. Se explican detalladamente las mecánicas de juego, el objetivo del jugador, y qué acciones se pueden realizar.

La única pregunta de esta sección es una genérica “¿cómo ha ido la experiencia?” para incitar a una breve reflexión, sin obligar al encuestado a pensar demasiado y encontrar la encuesta fatigante en este punto. La respuesta es una escala Likert, con el 1 equivaliendo a “no me ha gustado nada” y el 5, a “me ha gustado mucho”.

Es en la cuarta y quinta sección donde se realizan las preguntas más detalladas y relevantes sobre la interfaz de usuario y la jugabilidad, respectivamente. Se asume que a estas alturas de la encuesta, el usuario se encuentra comprometido con ella y se le puede pedir más información sin causar rechazo ni fatiga.

La sección sobre la interfaz gráfica, la cuarta, comienza preguntando en qué partes puede haberse quedado atascado o atascada. Se consideró definirla como de selección múltiple, para incitar a que se marcara cualquier obstáculo que se encontrara, no solamente los bloqueantes. Se intentaron describir las diferentes pantallas de forma que cualquier usuario las pudiera identificar, aun sin saber su propósito exacto. Las opciones eran:

- El menú principal
- La pantalla del mapa del mapa del juego
- El menú de los vehículos
- El menú para la ruta de los camiones
- El menú de las fábricas
- Otra (respuesta abierta)

A continuación, se realizaron dos preguntas de escala Likert para conocer la intuitividad y navegabilidad de los menús, respectivamente.

La primera preguntaba sobre la frecuencia en la que se encontraba un botón deseado en el lugar en el que se esperaba. El valor 1 se asocia con “casi nunca encontraba el botón que buscaba en el lugar en el que esperaba”, mientras que el 5 se corresponde con “casi siempre encontraba el botón que buscaba en el lugar en el que esperaba”. Se optó por descripciones tan largas para clarificar correctamente ambos extremos, aunque se suelen preferir frases más concisas.

La segunda pregunta tiene como enunciado “¿cómo de fácil te ha parecido navegar entre los menús?”, con ambos extremos representando “son muy confusos” o “son muy intuitivos”.

A continuación, se inquirió sobre la familiaridad de los menús. La pregunta es la siguiente: “¿cuál de las siguientes afirmaciones se corresponde mejor con tu experiencia con la interfaz?”, y las siguientes opciones:

- No tengo experiencia con juegos parecidos, pero la interfaz me ha parecido fácil de navegar.
- No tengo experiencia con juegos parecidos, pero la interfaz me ha parecido difícil de navegar.
- La interfaz es muy similar a la de otros juegos parecidos.
- La interfaz es diferente a la de otros juegos parecidos. Es igual o más fácil de navegar.
- La interfaz es diferente a la de otros juegos parecidos. Es más difícil de navegar.

En esta pregunta, se separan los casos en los que los encuestados y las encuestadas pueden haber probado o no juegos similares, y para todos los casos, se pregunta sobre su experiencia con la interfaz, a excepción de si la encuentran idéntica a la de otros juegos. Notar que, por no sobrecargar las respuestas, se agrupan los casos en los que la persona ha probado propuestas similares y encuentra la interfaz más fácil de navegar o igual de fácil. Se decidió agrupar estas dos para identificar de forma más sencilla casos en los que la interfaz haya resultado difícil de usar.

Para cerrar la sección, se añadió una pregunta de respuesta abierta sobre qué cosas se cambiarían de la interfaz, para poder capturar mejor qué elementos, o la ausencia de estos, causa frustración en un mayor número de usuarios y usuarias.

La quinta sección, sobre la jugabilidad, comienza con una pregunta directa: “¿qué te ha parecido la jugabilidad?”, con una escala Likert cuyos valores extremos son “muy aburrida” y “muy entretenida”.



A continuación, se inquiriere sobre cuánto tiempo duraría una sesión de juego, con los siguientes intervalos como respuestas:

- 5 – 25 minutos
- 30 – 55 minutos
- 1 – 3 horas
- Más de 3 horas

La siguiente pregunta trata sobre la dificultad, con una escala Likert cuyos extremos van desde “muy fácil” a “muy difícil”.

A continuación, se decidió inquirir, con una pregunta de selección múltiple, sobre qué características se podrían añadir para hacer el juego más disfrutable. Las diferentes opciones fueron las siguientes:

- Adversarios controlados por una Inteligencia Artificial que compiten contigo por los recursos del mapa
- Cadenas de producción más complejas, tener que fabricar componentes intermedios antes de fabricar el producto final
- Más opciones de comercialización, poder abrir tiendas y vender en función de la demanda y popularidad de nuestra marca
- Mayor dificultad, costes crecientes de las mercancías
- Mayor sensación de progreso, desbloquear industrias más caras conforme aumentan los ingresos
- Multijugador en línea
- Evolución del mundo y de las ciudades conforme se comercia con bienes
- Otra (respuesta abierta)

La sección finaliza con dos preguntas relacionadas con las mejoras. La primera, de escala Likert, inquiriere sobre la probabilidad de querer volver a jugar el juego si se incorporaran las mejoras marcadas, con valores del 1 al cinco asociados a “muy poco probable” y a “muy probable”, respectivamente.

La segunda pregunta, y con la que finaliza esta sección, tantea sobre cuánto se estaría dispuesto o dispuesta a pagar por el juego si incorporara las mejoras. Las opciones eran:

- Solo lo jugaría si fuera gratuito
- Entre 0,01 y 5,00€
- Entre 5,01 y 10,00€
- Entre 10,01 y 20,00€
- Entre 20,01 y 40,00€
- Más de 40,00€

Las preguntas de perfil se reservaron para la última sección de la encuesta, cuando el encuestado o encuestada puede mostrar mayores signos de fatiga y desea terminarla pronto. Así, se preguntó sobre el género, con las opciones “masculino”, “femenino” y “otro”, así como por la edad. Los intervalos de edad fueron los siguientes: “entre 1 y 8 años”, “entre 9 y 14 años”, “entre 15 y 19 años”, “entre 20 y 25 años”, “entre 26 y 35 años” y “más de 35 años”.

## 6.2. Respuestas obtenidas

El total de la muestra fue de cinco personas allegadas del autor. Dos chicas y tres chicos, de edades entre los 23 y 24 años, con conocimientos y hábitos de videojuego diversos. A pesar de esta riqueza de perfiles, el tamaño muestral es irrisorio, y se requeriría acceso a una mayor muestra para poder emitir un juicio más informado. No obstante, era el número disponible de voluntarios al realizar la encuesta, por lo que se estudiarán sus resultados aun a sabiendas de la limitada validez de la misma.

Por la manejabilidad del número de respuestas, y su poca representatividad, el análisis será superficial, y no se emplearán herramientas estadísticas, como RStudio.

La encuesta comienza preguntando por dominio de la tecnología y frecuencia de uso de videojuegos. El manejo de la tecnología oscila entre 4 y 5, mientras que la frecuencia de juego se encuentra repartida entre los diferentes valores, tal como se muestra en la siguiente imagen.

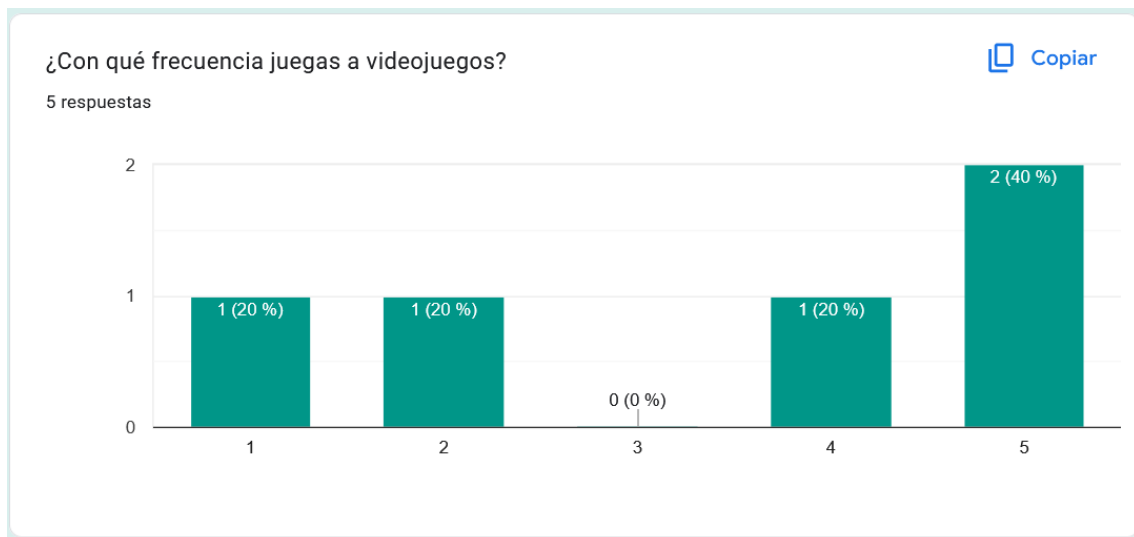


Ilustración 30. Frecuencia de uso de videojuegos.

Después de explicar el juego de forma resumida e invitar a los encuestados y encuestadas a experimentar con él, se obtiene, en la pregunta sobre la primera experiencia, puntuaciones de entre 4 y 5.

A continuación, en la sección sobre la interfaz de usuario, se pregunta, en primer lugar, si ha habido alguna pantalla en la que se hayan podido quedar atascados o atascadas. Destaca la pantalla del mapa, en la que tres de las cinco personas se quedaron atascadas.

La siguiente, sobre la intuitividad de los menús, pregunta la frecuencia en que, cuando se buscaba un botón, este se encontraba donde se esperaba. Los valores de las respuestas tienen una media de 4, como se puede observar en la ilustración más abajo. También se inquiriere sobre la facilidad de navegar por los menús, a lo que contestan con valores entre 3 y 4.

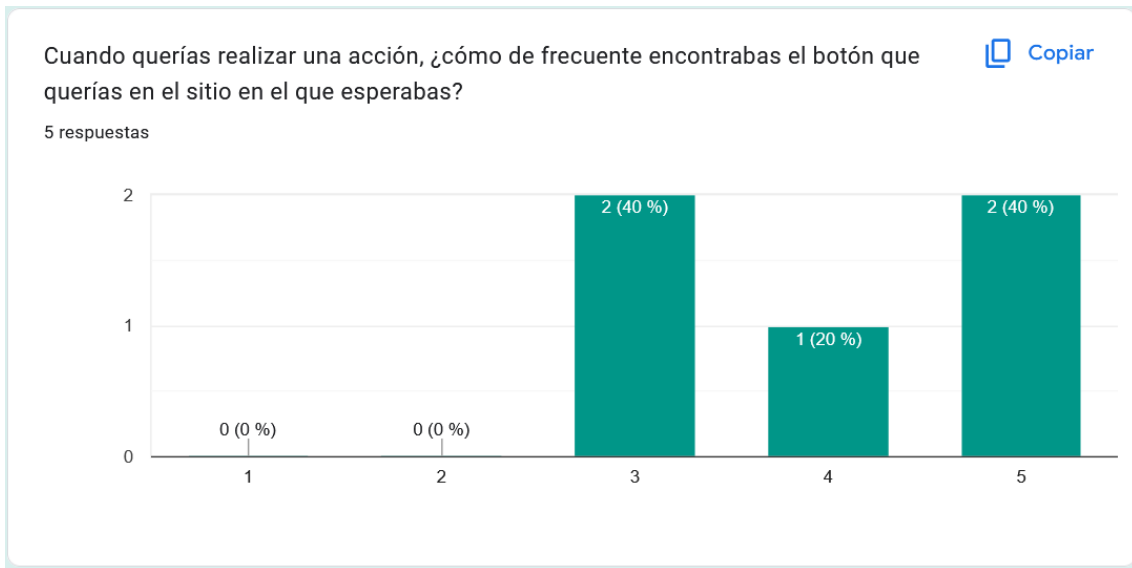


Ilustración 31. Pregunta sobre la intuitividad de los menús.

La penúltima pregunta, sobre la familiaridad de la interfaz, cuestionaba sobre cómo de parecidos eran los menús a los de otros videojuegos similares. Dos personas contestaron que la interfaz es muy similar a la de juegos parecidos, otra persona que no había jugado a juegos parecidos pero le parecía fácil de usar, y dos personas respondieron que no estaban familiarizados con juegos similares, pero les pareció difícil de usar.

Por último, en la pregunta de respuesta abierta para obtener sugerencias de cómo mejorar la interfaz, se obtuvieron las siguientes:

- Evitar el movimiento del mapa cuando el cursor se encuentra en el menú
- Destacar los edificios del mapa para indicar que son interactivables
- Permitir el movimiento de la cámara arrastrando con el ratón o con las teclas W, S, A y D.

La siguiente sección de la encuesta se centra en la jugabilidad. La primera pregunta plantea esta cuestión directamente, y se obtienen respuestas entre 3 y 5, indicando una buena acogida.

Cuando se les pregunta sobre cuánto tiempo pasarían en una sola partida, 3 personas contestan entre 5 y 25 minutos, una entre 30 y 55 minutos, y una última entre 1 y 3 horas.

A continuación, se presenta la pregunta de selección múltiple sobre qué características podrían hacer el juego más entretenido. Cuatro de los encuestados y encuestadas escogieron una mayor sensación de progreso, con industrias más caras desbloqueándose conforme los ingresos aumenten.

Otras tres opciones cuentan cada una con tres votos: mayor complejidad de la cadena de suministro, con productos intermedios que deben ser fabricados antes del producto final; más opciones de comercialización, usando variables como la demanda y popularidad de nuestra marca; y multijugador en línea. Únicamente dos personas escogieron oponentes controlados por la máquina.

Sobre cómo de probable era que, si el juego incorporase las mejoras, la persona encuestada quisiera volver a jugarlo, se obtuvieron tres respuestas con un valor de 5, una con un 4, y una con un 3, en una escala de Likert.

También se preguntó sobre qué precio se pagaría por el juego mejorado. Tres respuestas optaron por un rango de entre 5,01€ y 10,00€, otra por 10,01€ y 20,00€, y otra solo lo jugaría gratis.

Por último, se preguntó sobre género y edad. Como se ha explicado al inicio, la muestra constaba de 3 hombres y 2 mujeres, de entre 23 y 24 años de edad.

### Conclusiones de la encuesta

La encuesta ofrece una lectura positiva, con salvedades. La experiencia inicial, la interfaz gráfica y la jugabilidad obtienen buenas puntuaciones. La mayoría de los bloqueos ocurren en la pantalla de juego, es decir, en el mapa. Por su naturaleza, debe ser una pantalla desde la cual debe ser fácil acceder al resto, pero que no admite fácilmente elementos que puedan dar “pistas” sobre cómo acceder a todos ellos, por la sobrecarga visual que supondría. Así, se considera que la habituación del jugador a la pantalla compensará rápidamente estas situaciones iniciales de bloqueo.

No obstante, también se detectan bloqueos en el menú principal, en el menú de las fábricas, y a la hora de establecer las rutas de los camiones. Sería recomendable hacer encuestas futuras centradas en estos menús para identificar qué elementos en concreto causan frustración.

La navegabilidad e intuitividad obtienen puntuaciones altas, pero no lo suficiente como para poder afirmar que no son propensas de poder mejorarse.

Las respuestas obtenidas en la pregunta que cierra la sección delatan una insatisfacción general con la manera de mover la cámara, debiendo ser refinada para evitar desplazamientos indeseados, así como añadir opciones de movimiento presentes en juegos similares. También destaca la propuesta de resaltar los edificios interactivos.

Respecto a la jugabilidad, se obtienen de nuevo buenas puntuaciones, pero lejos de sugerir que es un apartado sin espacio para mejorar. La dificultad, por otra parte, se encuentra en un punto medio, sin ser excesiva ni demasiado fácil.

Una de las preguntas más interesantes es la de propuestas para mejorar la jugabilidad. Los elementos que mayor interés han causado han sido la capacidad de desbloquear nuevas industrias, la comercialización en tiendas en función de la demanda y la marca, y una mayor complejidad de la cadena de suministro. Todas ellas son características que, con los conocimientos actuales, serían sencillas aunque laboriosas de implementar, y representan un buen punto de partida para mejorar el juego, junto a las propuestas de mejora de interfaz.

Por último, hay una gran probabilidad de que los jugadores estuvieran interesados en volver a jugar, pagando un precio de entre 5,01 y 10,00€.



## 7. Conclusión

---

Llegando al final de este proyecto, es hora de hacer retrospectiva, reflexionar sobre los objetivos logrados, qué se ha aprendido, y qué cosas se podría haber hecho mejor.

Para este trabajo, nos habíamos propuesto una serie de objetivos. El primero de todos, era entender el sector de los videojuegos de gestión y simulación empresarial. Como ya hemos visto, en este sector hay algunos juegos similares a nuestra propuesta, si bien la oferta está desactualizada y tienden a ser juegos viejos, como la saga de Industry Giant. Aquellos más actuales o bien atraen poco interés, como es el caso de Rise of Industry, o buscan una experiencia de simulación más profunda, como Workers & Resources.

Otros juegos que tocan temas de logística de tema más tangencial son populares en el mercado y tienen una buena fuente de jugadores, como las últimas entregas de Trópico los juegos de Cities Skylines.

A partir de estas conclusiones, pudimos concretar algo más la propuesta, apostando por un mayor peso en la comercialización de productos, y cuidando la legibilidad y navegabilidad de los menús.

Antes de adentrarnos en la propuesta de forma más detallada, revisamos la literatura disponible sobre la psicología del juego. Estudiamos de forma superficial las teorías de diversos autores, como Huizinga, Caillois y Sutton-Smith, así como otros marcos teóricos desarrollados en el mercado de los videojuegos, como el de las 12 Motivaciones del Juego, de Quantic Foundry.

Con todos estos datos, elaboramos nuestra propuesta de forma detallada, para posteriormente analizarla y traducirla a un diagrama de clases y un diagrama de casos de uso, dentro del marco UML. Además, se elaboraron una serie de prototipos de las pantallas que usaría el juego.

A partir de la fase de diseño, conseguimos desarrollar un producto mínimo viable, que se compartió con allegados y allegadas del autor y cuyas opiniones se recogieron en una encuesta.

Con todo este proceso, podemos dar por cumplidos los objetivos propuestos al inicio del trabajo.

## 7.1. Lecciones aprendidas

Antes de proseguir, nos gustaría hacer una reflexión sobre qué se ha aprendido con este proyecto.

En primer lugar, se han trabajado todas las competencias relacionadas con la consecución de cualquier trabajo: planificación, establecimiento de objetivos, comunicación efectiva con el tutor, resolución de problemas, gestión del tiempo, autonomía, etc.

También se ha trabajado, al inicio y al final del proyecto, competencias relacionadas con la comercialización de productos. Se hizo un análisis de mercado superficial, al igual que una encuesta de pequeño alcance.

Otros conocimientos adquiridos han sido el uso del programa Godot, y de cómo se pueden desarrollar videojuegos, además del esfuerzo que involucran. Se han adquirido también destrezas en otras áreas, como la elaboración de arte de baja resolución.

Pero, en las que queremos hacer hincapié es en las lecciones más concretas aprendidas. La primera de todas es ser realistas con el alcance. Cuando se planteó este proyecto a principios-medios de abril, se consideró que su alcance era realista, incluso poco ambicioso; la realidad demostró lo contrario.

La sucesión de problemas ocurridos, desde los más importantes mencionados en el documento, pasando por puntuales bloqueos, lastraron este proyecto hasta el punto de haber tenido que eliminar algunos elementos. Ningún proyecto es fácil; todos los proyectos son propensos a sufrir contratiempos y contratiempos.

Esto es especialmente cierto en el desarrollo software, en el que a veces, las cosas no salen y no se entiende por qué. Se invierten horas en encontrar “qué pieza chirría” y, a veces, como es el caso de este proyecto, se debe llegar a una solución compromiso, no sin la frustración de los desarrolladores, que después de invertir tiempo, deben implementar una solución subóptima.

Ser realistas es importante, al igual que ser flexible y saber cuándo pasar a la siguiente tarea; es otra de las lecciones aprendidas en este proyecto, por los motivos indicados.

Otra competencia que se ha trabajado ha sido la creación de interfaces. Vistas las críticas a juegos similares por no tener pantallas claras y fáciles de navegar, nos pusimos como objetivo personal el de crear una interfaz sobresaliente, cómoda, intuitiva y fácil de usar.

Se pusieron en práctica cosas aprendidas en la carrera sobre interfaces. Se intentaron separar aquellas secciones destinadas a tareas distintas, a agrupar botones que servían funciones relacionadas, a dejar el espacio suficiente entre elementos cuyo clicado accidental podría causar frustración, a situar los componentes en zonas en las que un usuario o usuaria podría esperar que se encontraran, etc.





Todos estos retos son fáciles de escribir sobre el papel, pero muy complicados de cumplir de forma sistemática sin la experiencia ni el tiempo necesarios. A pesar de todo, estamos satisfechos con el resultado obtenido, pero somos conscientes, y la encuesta lo atestigua, de que hay cabida a mejoras.

Relacionado con esto, también hemos aprendido a cómo relacionar datos de una aplicación con la interfaz. En la carrera, frecuentemente nos hemos centrado en lo que ocurre “bajo el capó”, y muy raramente hemos llegado a programar interfaces, más allá de la asignatura destinada a ellas. Sin embargo, una correcta relación entre lo que se y lo que se almacena en el código es imprescindible para correcto desarrollo de la aplicación.

Vamos a poner un ejemplo concreto. A la hora de crear el menú con las rutas de los camiones, nuestra aproximación inicial fue la de almacenar, en el nodo del menú, una copia de la ruta de los camiones, y pasarle a cada submenú una de las destinaciones. Cada vez que se hacía un cambio en un submenú, se buscaba el índice del submenú en los hijos del menú, y se guardaba la destinación modificada en el elemento de la ruta que ocupara el mismo índice.

Esta forma de manejar los datos dio lugar a numerosas inconsistencias, y requirió de unos dos días, algo menos de veinte horas, de programar. Al tercer día, nos dimos cuenta de los problemas que generaba, y se rehízo totalmente. En lugar de trabajar con dos estructuras de datos simultáneamente, la ruta y la lista de submenús, se delegó totalmente la gestión de los destinos a los submenús, y no es hasta que se cierra el menú cuando se recupera la lista de destinos, se crea una nueva ruta con ellos y se le asigna al camión. De esta forma, siempre se encuentra en un estado consistente, y no hay ninguna modificación del usuario que pueda perderse en la traducción de la interfaz a la copia de la ruta almacenada.

Una forma “más correcta” que nos habría gustado implementar, y que pensamos hacia el final del proyecto, habría sido la de apuntar únicamente los cambios realizados, en lugar de rehacer completamente la ruta del camión. Tal como se ha hecho, el camión comienza su ruta de nuevo cada vez que se hace un cambio en ella. Un tratamiento más inteligente y sensible de los cambios realizados permitiría al camión seguir con la ruta sin verse afectado en gran medida.

Con todo, el proyecto se ha llevado a cabo y el resultado ha sido satisfactorio, pero somos conscientes de las cosas que podrían haberse hecho mejor, qué errores se han cometido que no deberían repetirse, y con qué lecciones nos quedamos para futuros trabajos.

## 7.2. Relación con los estudios

A lo largo de la carrera, hemos recibido la formación teórica y práctica en distintos campos de la informática, y que han fructificado en la realización de este proyecto.

Desde la asignatura de Introducción a la Informática y la Programación, con la que rompimos mano y descubrimos la programación orientada a objetos, pasando por asignaturas como Programación y Estructuras de Datos y Algoritmos, con las que adquirimos conocimientos más avanzados, poniendo el foco en la eficiencia espacial y temporal de los algoritmos, así como estrategias para elaborar software con el menor coste posible.

También es inevitable mencionar la asignatura Ingeniería del Software, centrada en la elaboración de aplicaciones con calidad. Se estudiaron y se entrenó en el uso de herramientas como los diagramas de clases y diagramas de casos de uso, así como en los tipos de pruebas para garantizar la máxima corrección posible del software.

A lo largo de la carrera, todas estas asignaturas nos han ofrecido una base sólida para realizar proyectos software, tales como el actual. Gradualmente, y curso tras curso, estos cimientos solidificaban y se añadía una nueva, con una asignatura tras otra expandiendo lo que las anteriores habían introducido.

Otra asignatura muy importante en este proyecto, que levantó nuestra curiosidad en el momento de cursarla, y que ha estado muy presente en la elaboración de este trabajo, es la Interfaces Persona Computador.

La habilidad de adaptar las herramientas al usuario y evitar que se tenga que dar el proceso inverso es muy importante. Como desarrolladores, e ingenieros en última instancia, es fácil dejarse llevar por las características del producto: querer crear la mejor aplicación, el software más eficiente, la herramienta más completa, el juego más divertido... Pero es importante tener en mente al usuario en todo momento, para que el producto final, sin importar su brillantez o mediocridad, pueda ser usado, y ofrezca valor y ayuda a las personas que lo usen. El hacha más afilada, sin un mango para agarrarla, no cortará ningún árbol.

En este proyecto, como se ha explicado en múltiples secciones anteriores, se ha dado gran relevancia a la legibilidad y navegabilidad de las pantallas, usando estrategias como la familiaridad, la agrupación de elementos comunes, la clara separación de elementos diferentes, y un diseño fácil de interpretar.

Dado el marco temporal disponible, estamos satisfechos con el resultado, pero nos habría gustado poder ir más allá. El mundo es diverso, y es de esperar que nuestros usuarios y usuarias lo sean. Existen personas con daltonismo, con discapacidades visuales, con autismo, y otras formas de diversidad funcional a las que es necesario tener en cuenta para poder presumir de una verdadera accesibilidad, por muy bien que estén hechos los menús o pantallas.



## 7.3. Competencias transversales

A lo largo de nuestra estancia en la Universitat Politècnica de València, hemos desarrollado y sido evaluados en una serie de competencias transversales, independientes a las asignaturas cursadas. Estas destrezas se aplican a un amplio marco de proyectos, muchas de ellas incluso a nuestra vida diaria, y nos aportan un mayor valor como profesionales, en comparación con otras personas con unos conocimientos similares.

A lo largo de la carrera, estas competencias transversales eran un total de treces, pero hace apenas dos meses, en el proceso de mejora continua educativa que lleva a cabo la UPV, fueron revisadas y reducidas a cinco competencias. Se relacionará el proyecto con este segundo grupo.

La más intuitiva, en el desarrollo de un videojuego, es la de Innovación y Creatividad. La elaboración de nuevos juegos, con sus reglas, mecanismos y dinámicas particulares requieren un cierto nivel de inventiva, y supone un ejercicio creativo para enriquecer la experiencia y hacerla lo más gratificante posible; estas mismas habilidades, que en este caso se han desarrollado y ampliado para la elaboración de un videojuego pueden servir, en un futuro, para ofrecer soluciones únicas e innovadoras a problemas y retos que se presenten. Este proyecto nos ha obligado a mirar más allá de lo que existe, y ponernos a pensar en lo que podría existir, de igual manera que exige el proceso de resolución de problemas.

Pero la que más se ha trabajado a lo largo del trabajo, ha sido la de Responsabilidad y Toma de Decisiones. Desde el momento en el que se escoge el tema de este proyecto, da comienzo una fase de trabajo autónomo que exige un alto nivel de autonomía, disciplina y gestión del tiempo. Todos ellos requieren una gran toma de responsabilidad para seguir avanzando.

Pero el avance no siempre era posible, o todo lo fluido que se quería. Ha habido múltiples instancias de bloqueo. Algunas más graves, como las que se han tratado anteriormente, y otras que solamente han supuesto retrasos de algunas horas. Era en esos casos cuando mayor compromiso ha sido necesario, para no tirar la toalla y seguir hasta el final.

Además, en las situaciones de bloqueo más críticas, ha sido necesario tomar decisiones que no siempre han sido satisfactorias; fue necesario realizar concesiones y optar por implementaciones menos vistosas a cambio de poder seguir con el transcurso del proyecto. Estas disyuntivas no han sido sencillas, y han obligado a adoptar una actitud más flexible para hacer frente a aquellos bloqueos que, por la naturaleza del trabajo, no se poseía suficiente experiencia para solucionar.

## 8. Trabajo futuro

---

Todo software es propenso a recibir mejoras. Los videojuegos están muy lejos de ser una excepción a esta regla. Desde el inicio se identificaron características deseables, pero cuya intención de implementación no habría sido realista. Concretamente, estamos hablando de adversarios controlados por la máquina que puedan rivalizar con el jugador humano, añadiendo un importante grado de competitividad a la experiencia de juego.

Sin embargo, la encuesta a usuario, aunque poco representativa, sugiere que esta adición no despierta tanto interés como, por ejemplo, el multijugador en línea. Esta mejora, sin embargo, también quedaba lejos del alcance realista de la propuesta, pero podría ser un desarrollo interesante, y potencialmente con una mejor acogida que invertir horas en programar adversarios controlados por la máquina.

Pero también hay un alto número de desarrollos con menor complejidad que han quedado en el tintero. En primer lugar, la característica que se echa más en falta es la apertura de tiendas en las ciudades, con diferentes niveles de demanda en función del distrito que puedan afectar al volumen de producto que se pueda vender exitosamente.

Atendiendo a la encuesta, el desplazamiento de la cámara con el ratón es susceptible de ser retrabajado, evitando movimientos indeseados por el mapa, así como añadir opciones de desplazamientos esperadas por el jugador. La encuesta ha destacado este punto como particularmente problemático.

Igualmente, se podría mejorar la interacción con los elementos del mapa: destacar elementos seleccionables para transmitir al jugador que son interactivables, así como cambiar la forma en que se detecta la selección de edificios en el mapa, con el fin de hacerla más intuitiva.

Los vehículos también son propensos a mejorar. En primer lugar, una correcta navegación por las carreteras y vías permitiría una mayor inmersión, y la posibilidad de seleccionarlos directamente en el mapa haría su gestión más sencilla y ágil.

Asimismo, la forma en que está programado el juego permite la sencilla adición de nuevos modelos de vehículos, tipos de explotaciones de materias primas, tipos de mercancía y líneas de fabricación, pudiendo enriquecer el juego y alcanzar niveles de complejidad logística relativamente elevados, tal como la encuesta sugiere que los jugadores y las jugadoras desean. También se podrían añadir fácilmente industrias más caras, desbloqueables conforme los niveles de ingresos aumentan.

Otras características, que se planeaban haber incluido pero, por motivos de tiempo se descartaron, fue la inclusión de música de fondo que acompañara la experiencia y la posibilidad de guardar y cargar partida.

Todas estas, así como muchas otras pequeñas mejoras en aspectos diversos ayudarían a mejorar la experiencia de juego y hacerla más gratificante.



## 9. Bibliografía y referencias

---

- [1] K. Sylva, J. S. Bruner, y A. Jolly, *Play: its role in development and evolution*. 1976. Accedido: 23 de mayo de 2024. [En línea]. Disponible en: [https://www.researchgate.net/publication/328486788\\_Play\\_its\\_role\\_in\\_development\\_and\\_evolution](https://www.researchgate.net/publication/328486788_Play_its_role_in_development_and_evolution)
- [2] T. Gautier, *Mademoiselle de Maupin*. Paris, 1868.
- [3] B. Wirtz, «Unreal vs Unity Engine: Pros and Cons», <https://www.gamedesigning.org/engines/unity-vs-unreal/>.
- [4] Starbreeze Studios, «PAYDAY 3», [https://store.steampowered.com/app/1272080/PAYDAY\\_3/](https://store.steampowered.com/app/1272080/PAYDAY_3/). Accedido: 23 de mayo de 2024. [En línea]. Disponible en: [https://store.steampowered.com/app/1272080/PAYDAY\\_3/](https://store.steampowered.com/app/1272080/PAYDAY_3/)
- [5] R. Dillet, «Unity CEO says half of all games are built on Unity», <https://techcrunch.com/2018/09/05/unity-ceo-says-half-of-all-games-are-built-on-unity/>.
- [6] L. Bellbrook, «Taking a Closer Look at Unity and Unreal Engine», <https://aircada.com/unity-vs-unreal-engine/>.
- [7] Digixart, «Road 96», [https://store.steampowered.com/app/1466640/Road\\_96/](https://store.steampowered.com/app/1466640/Road_96/). Accedido: 23 de mayo de 2024. [En línea]. Disponible en: [https://store.steampowered.com/app/1466640/Road\\_96/](https://store.steampowered.com/app/1466640/Road_96/)
- [8] Zenva, «Unity vs Godot – A Game Engine Analysis», <https://gamedevacademy.org/unity-vs-godot/>.
- [9] Blobfish, «Brotato», <https://store.steampowered.com/app/1942280/Brotato/>. Accedido: 23 de mayo de 2024. [En línea]. Disponible en: <https://store.steampowered.com/app/1942280/Brotato/>
- [10] J. Huizinga, *Homo Ludens*. 1938.
- [11] R. Caillois, *Les jeux et les hommes*. 1958.
- [12] B. Sutton-Smith, *The Ambiguity of Play*. Harvard University Press, 1997.
- [13] N. Ducheneaut y N. Yee, «Gamer Motivation Model», <https://quanticfoundry.com/#motivation-model>.
- [14] H. Jebens, «Are You the Next Industry Giant?», [http://headline.gamespot.com/news/98\\_04/29\\_imagic/index.html](http://headline.gamespot.com/news/98_04/29_imagic/index.html). Accedido: 25 de mayo de 2024. [En línea]. Disponible en:

[https://web.archive.org/web/20001005031630/http://headline.gamespot.com/news/98\\_04/29\\_imagic/index.html](https://web.archive.org/web/20001005031630/http://headline.gamespot.com/news/98_04/29_imagic/index.html)

[15] Metacritic, «Industry Giant», <https://www.metacritic.com/game/industry-giant/>. Accedido: 25 de mayo de 2024. [En línea]. Disponible en: <https://www.metacritic.com/game/industry-giant/>

[16] M. Oertel, «Der Industriegigant 2 (Taktik & Strategie) – Transportwesen mit Retro-Charme », [https://www.4p.de/test/der\\_industriegigant\\_2-2/82309](https://www.4p.de/test/der_industriegigant_2-2/82309). Accedido: 25 de mayo de 2024. [En línea]. Disponible en: [https://www.4p.de/test/der\\_industriegigant\\_2-2/82309](https://www.4p.de/test/der_industriegigant_2-2/82309)

[17] MobyGames, «Industry Giant II», <https://www.mobygames.com/game/8596/industry-giant-ii/>. Accedido: 25 de mayo de 2024. [En línea]. Disponible en: <https://www.mobygames.com/game/8596/industry-giant-ii/>

[18] QLOC, «Projects», <https://q-loc.com/projects/>. Accedido: 25 de mayo de 2024. [En línea]. Disponible en: <https://q-loc.com/projects/>

[19] Don VS Dodo, «Industry Giant 4.0», <https://donvsdodo.de/industry-giant-4-0/>. Accedido: 25 de mayo de 2024. [En línea]. Disponible en: <https://donvsdodo.de/industry-giant-4-0/>

[20] JoWood y QLOC, «Industry Giant», [https://store.steampowered.com/app/521090/Industry\\_Giant/](https://store.steampowered.com/app/521090/Industry_Giant/). Accedido: 25 de mayo de 2024. [En línea]. Disponible en: [https://store.steampowered.com/app/521090/Industry\\_Giant/](https://store.steampowered.com/app/521090/Industry_Giant/)

[21] R. Fancy Bytes, «Industry Giant 2», [https://store.steampowered.com/app/271360/Industry\\_Giant\\_2/](https://store.steampowered.com/app/271360/Industry_Giant_2/). Accedido: 25 de mayo de 2024. [En línea]. Disponible en: [https://store.steampowered.com/app/271360/Industry\\_Giant\\_2/](https://store.steampowered.com/app/271360/Industry_Giant_2/)

[22] Don VS Dodo, «Industry Giant 4.0», [https://store.steampowered.com/app/1129570/Industry\\_Giant\\_40/](https://store.steampowered.com/app/1129570/Industry_Giant_40/).

[23] SteamCharts, «Industry Giant», <https://steamcharts.com/app/521090>. Accedido: 25 de mayo de 2024. [En línea]. Disponible en: <https://steamcharts.com/app/521090>

[24] SteamCharts, «Industry Giant 2», <https://steamcharts.com/app/271360>. Accedido: 25 de mayo de 2024. [En línea]. Disponible en: <https://steamcharts.com/app/271360>

[25] Dapper Penguin Studios, «Rise of Industry», [https://store.steampowered.com/app/671440/Rise\\_of\\_Industry/](https://store.steampowered.com/app/671440/Rise_of_Industry/). Accedido: 26 de mayo de 2024. [En línea]. Disponible en: [https://store.steampowered.com/app/671440/Rise\\_of\\_Industry/](https://store.steampowered.com/app/671440/Rise_of_Industry/)



[26] SteamCharts, «Rise of Industry», <https://steamcharts.com/app/671440>. Accedido: 26 de mayo de 2024. [En línea]. Disponible en: <https://steamcharts.com/app/671440>

[27] A. Mochi, «Dapper Penguin Studios is no more - A passionate letter from a Game Developer», <https://store.steampowered.com/news/app/1492360/view/3972805574280463610>. Accedido: 26 de mayo de 2024. [En línea]. Disponible en: <https://store.steampowered.com/news/app/1492360/view/3972805574280463610>

[28] Kasedo Games, «Rise of Industry 2», <https://www.riseofindustry.com/>. Accedido: 26 de mayo de 2024. [En línea]. Disponible en: <https://www.riseofindustry.com/>

[29] 3Division, «Workers & Resources: Soviet Republic», [https://store.steampowered.com/app/784150/Workers\\_\\_Resources\\_Soviet\\_Republic/](https://store.steampowered.com/app/784150/Workers__Resources_Soviet_Republic/). Accedido: 25 de mayo de 2024. [En línea]. Disponible en: [https://store.steampowered.com/app/784150/Workers\\_\\_Resources\\_Soviet\\_Republic/](https://store.steampowered.com/app/784150/Workers__Resources_Soviet_Republic/)

[30] SteamCharts, «Workers & Resources: Soviet Republic», <https://steamcharts.com/app/784150>. Accedido: 25 de mayo de 2024. [En línea]. Disponible en: <https://steamcharts.com/app/784150>

[31] C. Fabricatore, «Gameplay and game mechanics design: a key to quality in videogames», en *OECD-CERI*, oct. 2007. doi: <http://dx.doi.org/10.13140/RG.2.1.1125.4167>.

[32] M. Lynch, «What is a gameplay loop in gaming?», The Tech Edvocate. Accedido: 26 de mayo de 2024. [En línea]. Disponible en: <https://www.thetechedvocate.org/what-is-a-gameplay-loop-in-gaming/>





## ANEXO

### OBJETIVOS DE DESARROLLO SOSTENIBLE

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenibles	Alto	Medio	Bajo	No Procede
ODS 1. <b>Fin de la pobreza.</b>				X
ODS 2. <b>Hambre cero.</b>				X
ODS 3. <b>Salud y bienestar.</b>				X
ODS 4. <b>Educación de calidad.</b>		X		
ODS 5. <b>Igualdad de género.</b>				X
ODS 6. <b>Agua limpia y saneamiento.</b>				X
ODS 7. <b>Energía asequible y no contaminante.</b>				X
ODS 8. <b>Trabajo decente y crecimiento económico.</b>			X	
ODS 9. <b>Industria, innovación e infraestructuras.</b>				X
ODS 10. <b>Reducción de las desigualdades.</b>				X
ODS 11. <b>Ciudades y comunidades sostenibles.</b>				X
ODS 12. <b>Producción y consumo responsables.</b>			X	
ODS 13. <b>Acción por el clima.</b>				X
ODS 14. <b>Vida submarina.</b>				X
ODS 15. <b>Vida de ecosistemas terrestres.</b>				X
ODS 16. <b>Paz, justicia e instituciones sólidas.</b>				X
ODS 17. <b>Alianzas para lograr objetivos.</b>				X





Reflexión sobre la relación del TFG/TFM con los ODS y con el/los ODS más relacionados.

Aunque este proyecto no incide profundamente en ninguno de los Objetivos de Desarrollo Sostenible, sí que puede relacionarse con varios de ellos por su naturaleza y los temas que trata.

El ODS con el que guarda una mayor relación es el cuarto, el de Educación de calidad.

El cuarto ODS promueve la educación inclusiva y de calidad, para todas las edades. Los videojuegos tienen, en este sentido, un gran potencial, ya que su finalidad lúdica permite tratar de una forma entretenida y amena temas complejos del mundo.

La idea de usar videojuegos para la educación no es nueva. Pero conforme la industria de los videojuegos ha crecido, es cada vez más común que juegos que no se desarrollaron con finalidades educativas contengan unos elementos de inmersión y realismo que el jugador pueda aprender.

Podría ser el caso de nuestro juego, que intenta simular la gestión logística de una empresa de producción y comercialización de mercancía diversificada.

La gestión logística, además, es un componente clave del comercio mundial, que ha convertido al planeta en una gran red de suministro interconectada. Este comercio global ha permitido la disponibilidad de alimentos y materias primas en los que, originalmente, no se podían encontrar.

Permite, por ejemplo, que se extraiga petróleo en el Golfo Pérsico, se refine a plástico en instalaciones en Croacia, se transporte hasta California para crear componentes electrónicos junto con cobre y elementos raros extraídos en Nigeria, para posteriormente ser incorporados en coches alemanes.

Esta complejidad, difícil de explicar de forma intuitiva, se vuelve un elemento a explorar y componente central de la jugabilidad de nuestra propuesta.

En la línea de lo anteriormente expuesto, un mejor entendimiento de la cadena de suministro y una familiarización con ella desde tempranas edades puede dar lugar a consumidores más concienciados con el esfuerzo, y la contaminación, requeridos para proporcionar los servicios y productos que demandan diariamente.

Una mayor concienciación de los consumidores dará lugar a un consumo más responsable, de forma que se busque colectivamente reducir las emisiones y comprar productos de cercanía, así como consumir alimentos locales.



UNIVERSITAT  
POLITÈCNICA  
DE VALÈNCIA



Todo esto nos permite intuir que este proyecto ayudará a trabajar por lograr el Objetivo de Desarrollo Sostenible número 12, el de Producción y consumo responsables.

Igualmente, el juego nos permite entrenar de forma amena actividades que nos podrían ser de utilidad en nuestra vida. Simulando la gestión de negocios y, más importante todavía, experimentando sin repercusiones económicas, se pueden desarrollar estrategias innovadoras potencialmente aplicables a escenarios reales. De esta forma, la propuesta se relaciona también con el Objetivo de Desarrollo Sostenible número 8, de Trabajo decente y crecimiento económico.



Escola Tècnica  
Superior d'Enginyeria  
Informàtica

**ETS Enginyeria Informàtica**  
Camí de Vera, s/n. 46022. València  
**T** +34 963 877 210  
**F** +34 963 877 219  
etsinf@upvnet.upv.es - www.inf.upv.es



# Anexo. Mock-ups.

---



*Ilustración 1. Pantalla del menú principal.*

## New Game

Map size:

16 x 16

Cities:

20

Production  
buildings:

12

Harbors:

10

Back

Start Game

*Ilustración 2. Pantalla de selección de mapa.*

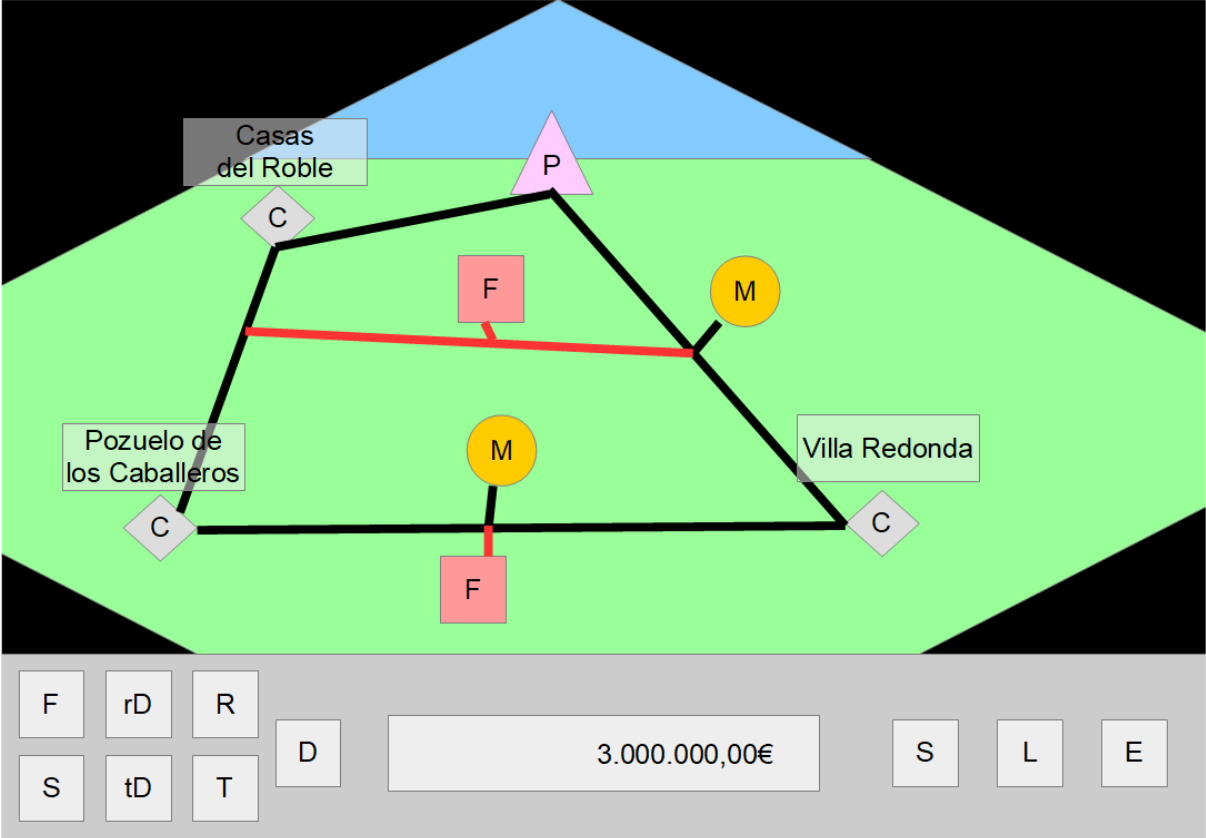


Ilustración 3. Pantalla del mapa,

# Load Game

Game 1 17:52h, 28/05/2024
Game 2 18:38h, 28/05/2024
Game 3 19:25h, 28/05/2024

Back	Load Game
------	-----------

*Ilustración 4. Pantalla de carga de partida.*

# Load Game

<i>New save...</i> 19:42h, 28/05/2024
Game 1 17:52h, 28/05/2024
Game 2 18:38h, 28/05/2024
Game 3 19:25h, 28/05/2024

Back      Save Game

*Ilustración 5. Pantalla de guardado de partida.*

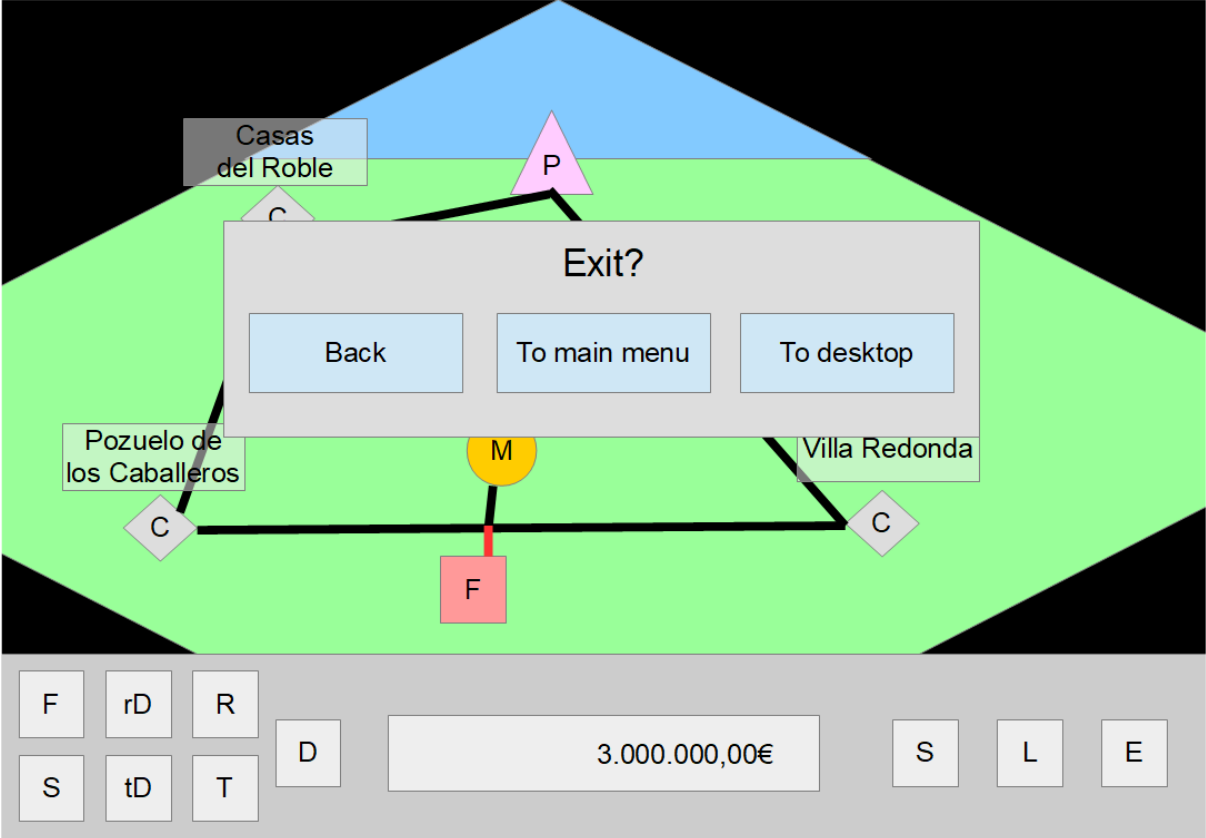


Ilustración 6. Pantalla de salida.



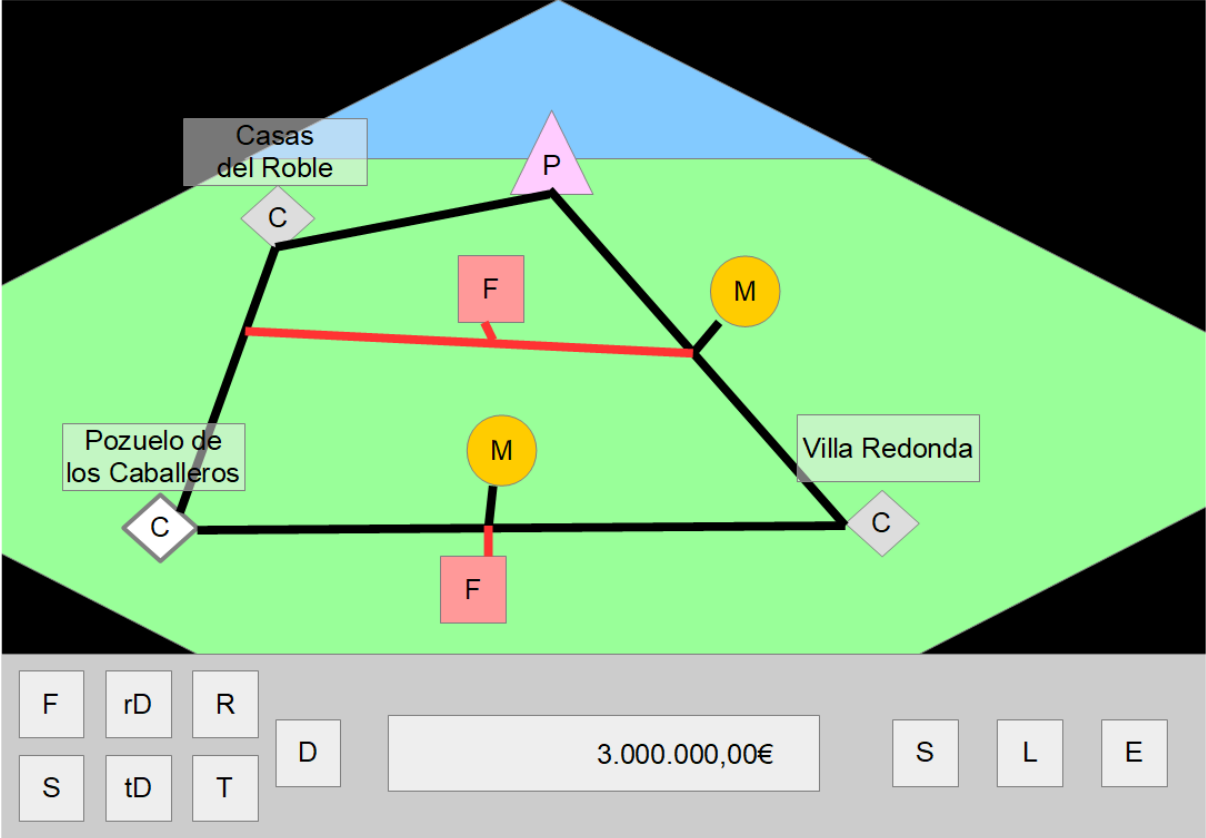



Ilustración 7. Pantalla del mapa con una ciudad seleccionada.

Pozuelo de los Caballeros

Population: 4.314.811  
Attractiveness: +8%



Tienda 1  
Distrito Casales

Sells:

- Wooden toys | 12 / 12 | 6€
- Plastic toys | 31 / 44 | 3€
- Puzzles | 26 / 29 | 4€

Tienda 2  
Distrito de Panaderos

Sells:

- Furniture | 4 / 4 | 40€
- Lamps | 14 / 16 | 12€

3.000.000,00€

Ilustración 8. Pantalla del mapa de distritos.

Pozuelo de los Caballeros

Population: 4.314.811  
Attractiveness: +8%

Distrito Casales

Population: 422.180  
Attractiveness: +0%  
Wealth: Humble  
Land price: 1000€/m<sup>2</sup>

+ New shop

Tienda 1

Distrito Casales

Sells:

- Wooden toys | 12 / 12 | 6€
- Plastic toys | 31 / 44 | 3€
- Puzzles | 26 / 29 | 4€

3.000.000,00€

Ilustración 9. Pantalla del mapa de distritos, con un distrito seleccionado en el que hay una tienda.

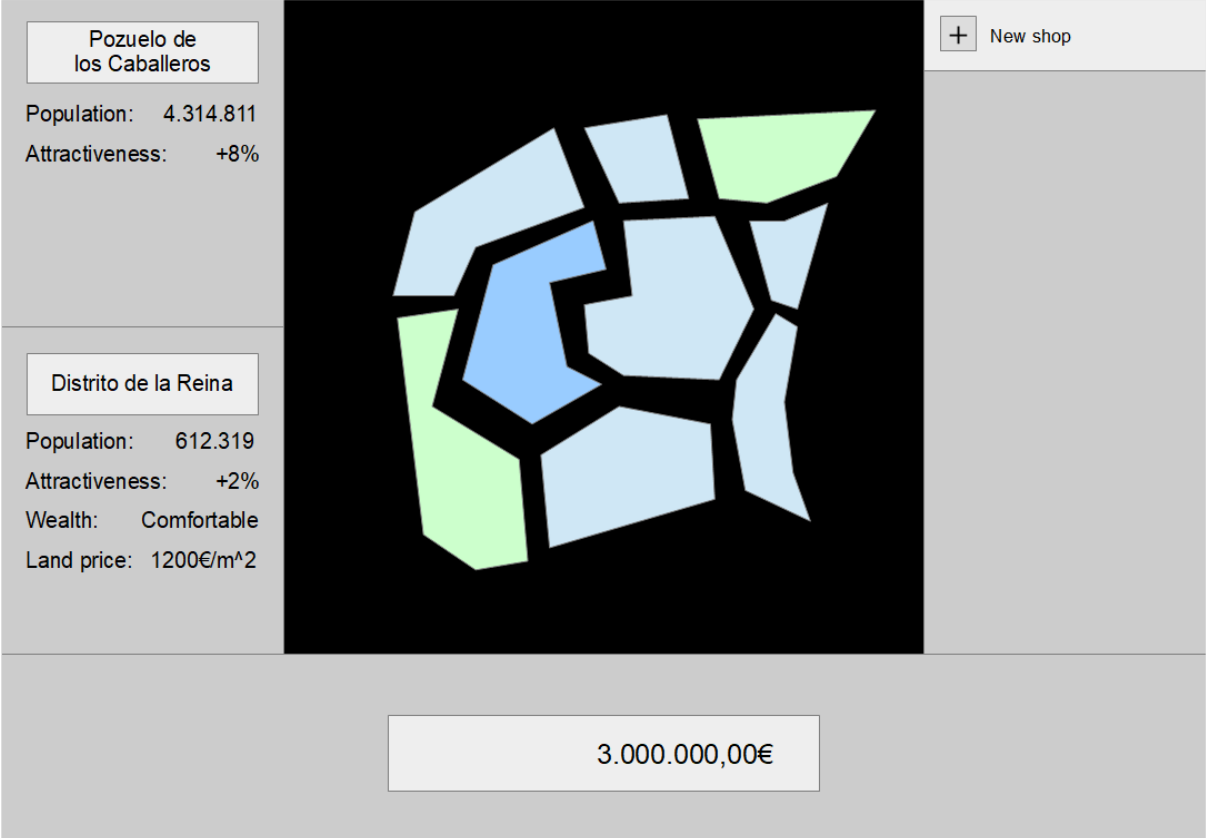


Ilustración 10. Pantalla del mapa de distritos, con un distrito seleccionado en el que no hay tiendas todavía.

**Pozuelo de los Caballeros**

Population: 4.314.811  
Attractiveness: +8%

**Tienda 1**

**Tienda 1**

Distrito Casales

Sells:

- Wooden toys | 12 / 12 | 6€
- Plastic toys | 31 / 44 | 3€
- Puzzles | 26 / 29 | 4€

**Distrito Casales**

Population: 422.180  
Attractiveness: +0%  
Wealth: Humble  
Land price: 1000€/m<sup>2</sup>

**Catalogue**

	Product	Margin	Supply / Demand	Price
<input style="width: 20px;" type="button" value="+"/>	New offer			
<input style="width: 20px;" type="button" value="✎"/>	Wooden Toys	+10%	12 / 12	6€
<input style="width: 20px;" type="button" value="✎"/>	Plastic Toys		31 / 44	3€

**3.000.000,00€**

Ilustración 11. Pantalla de las tiendas.

**Pozuelo de los Caballeros**

Population: 4.314.811  
Attractiveness: +8%

---

**Distrito Casales**

Population: 422.180  
Attractiveness: +0%  
Wealth: Humble  
Land price: 1000€/m<sup>2</sup>

**Tienda 1**

Area: 40m<sup>2</sup>  
Zone: Quiet  
Max Products: 3  
Traffic: 561 k

**Catalogue**

	Product	Margin	Supply / Demand	Price
<input checked="" type="checkbox"/>	Choose...	0%	0 / 0	0€
<input type="checkbox"/>	Wooden Toys	+10%	12 / 12	6€
<input type="checkbox"/>	Plastic Toys		31 / 44	3€

**Tienda 1**

Distrito Casales

Sells:

- Wooden toys | 12 / 12 | 6€
- Plastic toys | 31 / 44 | 3€
- Puzzles | 26 / 29 | 4€

3.000.000,00€

Ilustración 12. Pantalla de las tiendas cuando se clica en "Nueva oferta".

**Pozuelo de los Caballeros**

Population: 4.314.811  
Attractiveness: +8%

---

**Distrito Casales**

Population: 422.180  
Attractiveness: +0%  
Wealth: Humble  
Land price: 1000€/m<sup>2</sup>

**Tienda 1**

Area: 40m<sup>2</sup>  
Zone: Quiet  
Max Products: 3  
Traffic: 561 k

**Catalogue**

Product	Margin	Supply / Demand   Price
<input checked="" type="checkbox"/> <input type="checkbox"/> Robots	0%	0 / 6   14€
<input type="checkbox"/> <input type="checkbox"/> Wooden Toys	+10%	12 / 12   6€
<input type="checkbox"/> <input type="checkbox"/> Plastic Toys		31 / 44   3€

**Tienda 1**  
Distrito Casales

Sells:

- Wooden toys | 12 / 12 | 6€
- Plastic toys | 31 / 44 | 3€
- Puzzles | 26 / 29 | 4€

3.000.000,00€

Ilustración 13. Pantalla de las tiendas cuando se clica en "Nueva oferta" y se selecciona un producto.

<p>Pozuelo de los Caballeros</p> <p>Population: 4.314.811</p> <p>Attractiveness: +8%</p>	<p><u>Shop name</u></p>																
<p>Distrito de la Reina</p> <p>Population: 612.319</p> <p>Attractiveness: +2%</p> <p>Wealth: Comfortable</p> <p>Land price: 1200€/m<sup>2</sup></p>	<p>Size</p> <p>20 m<sup>2</sup></p>	<p>Zone</p> <p><input checked="" type="radio"/> Quiet <input type="radio"/> Bustling</p> <p><i>No bonus</i></p>															
<p>Cost: 24.000 €</p> <p>Max Products: 1</p> <p>Estimated Traffic: 970 k</p>																	
<p><u>Catalogue</u></p> <table border="1"> <thead> <tr> <th>Product</th> <th>Margin</th> <th>Supply / Demand</th> <th>Price</th> </tr> </thead> <tbody> <tr> <td><input type="button" value="+"/> New offer</td> <td></td> <td></td> <td></td> </tr> <tr> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>		Product		Margin	Supply / Demand	Price	<input type="button" value="+"/> New offer										
Product	Margin	Supply / Demand	Price														
<input type="button" value="+"/> New offer																	
<p>3.000.000,00€</p>																	

Ilustración 14. Pantalla de las tiendas cuando se crea una tienda.



<p>Pozuelo de los Caballeros</p> <p>Population: 4.314.811</p> <p>Attractiveness: +8%</p>	<p><u>Shop name</u></p>																
<p>Distrito de la Reina</p> <p>Population: 612.319</p> <p>Attractiveness: +2%</p> <p>Wealth: Comfortable</p> <p>Land price: 1200€/m<sup>2</sup></p>	<p>Size</p> <p>20 m<sup>2</sup></p>	<p>Zone</p> <p><input type="radio"/> Quiet <input checked="" type="radio"/> Bustling</p> <p>+10% cost, +10% traffic</p>															
<p>Cost: 26.400 €</p> <p>Max Products: 1</p> <p>Estimated Traffic: 1.067 k</p>																	
<p><u>Catalogue</u></p> <table border="1"> <thead> <tr> <th>Product</th> <th>Margin</th> <th>Supply / Demand</th> <th>Price</th> </tr> </thead> <tbody> <tr> <td><input type="button" value="+"/> New offer</td> <td></td> <td></td> <td></td> </tr> <tr> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>		Product		Margin	Supply / Demand	Price	<input type="button" value="+"/> New offer										
Product	Margin	Supply / Demand	Price														
<input type="button" value="+"/> New offer																	
<p>3.000.000,00€</p>																	

Ilustración 15. Pantalla de las tiendas cuando se crea una nueva y se selecciona la opción de "Zona ajetreada".





<p><b>Pozuelo de los Caballeros</b></p> <p>Population: 4.314.811 Attractiveness: +8%</p>	<p align="center"><u>Shop name</u></p>		<p>Tienda 1</p> <p>Districto Casales</p> <p>Sells:</p> <ul style="list-style-type: none"> <li>- Wooden toys   12   6€</li> <li>- Plastic toys   31   3€</li> <li>- Puzzles   26   4€</li> </ul> <p align="right"></p>								
<p><b>Districto Casales</b></p> <p>Population: 422.180 Attractiveness: +0% Wealth: Humble Land price: 1000€/m<sup>2</sup></p>	<p>Size</p> <p>20 m<sup>2</sup></p>	<p>Zone</p> <p><input checked="" type="radio"/> Quiet <input type="radio"/> Bustling</p> <p><i>No bonus</i></p>									
<p>Cost: 20.000 €</p> <p>Max Products: 1</p> <p>Estimated Traffic: 767 k</p>		<p align="center"><u>Catalogue</u></p> <table border="1"> <thead> <tr> <th>Product</th> <th>Margin</th> <th>Supply / Demand</th> <th>Price</th> </tr> </thead> <tbody> <tr> <td> New offer</td> <td></td> <td></td> <td></td> </tr> </tbody> </table>		Product	Margin	Supply / Demand	Price	 New offer			
Product	Margin	Supply / Demand	Price								
 New offer											
<p align="center">3.000.000,00€</p>											

Ilustración 16. Pantalla de las tiendas cuando se crea una nueva en un distrito en el que ya hay alguna tienda.





<p><b>Pozuelo de los Caballeros</b></p> <p>Population: 4.314.811</p> <p>Attractiveness: +8%</p>	<p><u>Shop name</u></p>		<p>Tienda 1</p> <p>Distrito Casales</p> <p>Sells:</p> <ul style="list-style-type: none"> <li>- Wooden toys   12   6€</li> <li>- Plastic toys   31   3€</li> <li>- Puzzles   26   4€</li> </ul> <p></p>								
<p><b>Distrito Casales</b></p> <p>Population: 422.180</p> <p>Attractiveness: +0%</p> <p>Wealth: Humble</p> <p>Land price: 1000€/m<sup>2</sup></p>	<p>Size</p> <p>20 m<sup>2</sup></p>	<p>Zone</p> <p><input type="radio"/> Quiet <input checked="" type="radio"/> Bustling</p> <p>+10% cost, +10% traffic</p>									
<p>Cost: 22.000 €</p> <p>Max Products: 1</p> <p>Estimated Traffic: 844 k</p>		<p><u>Catalogue</u></p> <table border="1"> <thead> <tr> <th>Product</th> <th>Margin</th> <th>Supply / Demand</th> <th>Price</th> </tr> </thead> <tbody> <tr> <td> New offer</td> <td></td> <td></td> <td></td> </tr> </tbody> </table>		Product	Margin	Supply / Demand	Price	 New offer			
Product	Margin	Supply / Demand	Price								
 New offer											
<p>3.000.000,00€</p>											

Ilustración 17. Pantalla de las tiendas cuando se crea una tienda en un barrio en el que ya hay alguna tienda y se selecciona la opción "Zona ajetreada".



Input		Fábrica 1		Output	
Aluminium	x44	Aluminium	x2	Lamp	x6
Electronics	x21	Electronics	x2		
Plastic	x31	Plastic	x1		
Cloth	x12	Cloth	x1		
		 			
		0.8 / 1.3 s			
		<input type="button" value="+"/> New production line			
<div style="border: 1px solid black; padding: 5px; width: fit-content; margin: 0 auto;">3.000.000,00€</div>					

Ilustración 18. Pantalla de las fábricas.

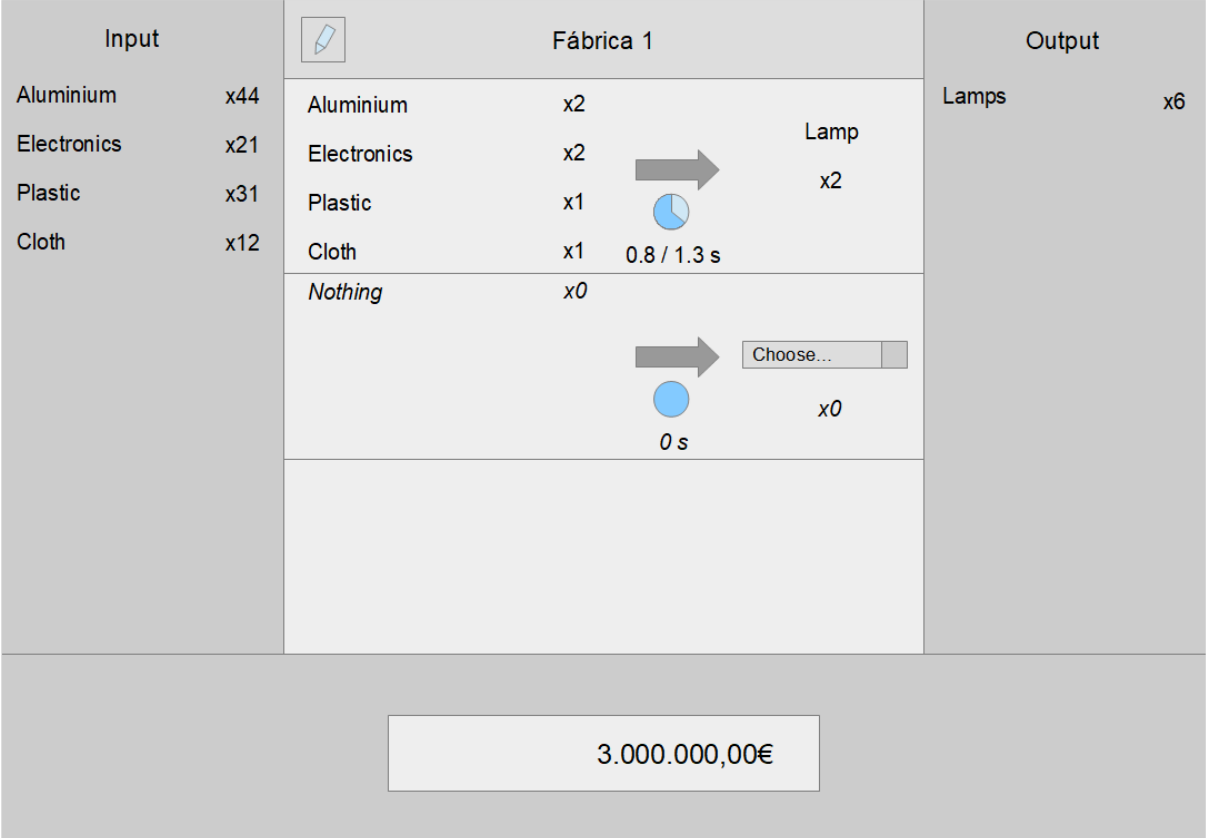


Ilustración 19. Pantalla de las fábricas cuando se clica "Nueva línea de producción".

Input		Fábrica 1		Output		
Aluminium	x44	Aluminium	x2	Lamp	Lamps	
Electronics	x21	Electronics	x2			x2
Plastic	x31	Plastic	x1	0.8 / 1.3 s		
Cloth	x12	Cloth	x1			
		Wood	x2			
				Furniture		
				3 s	x1	

3.000.000,00€

Ilustración 20. Pantalla de las fábricas cuando se clica "Nueva línea de producción" y se selecciona un producto.

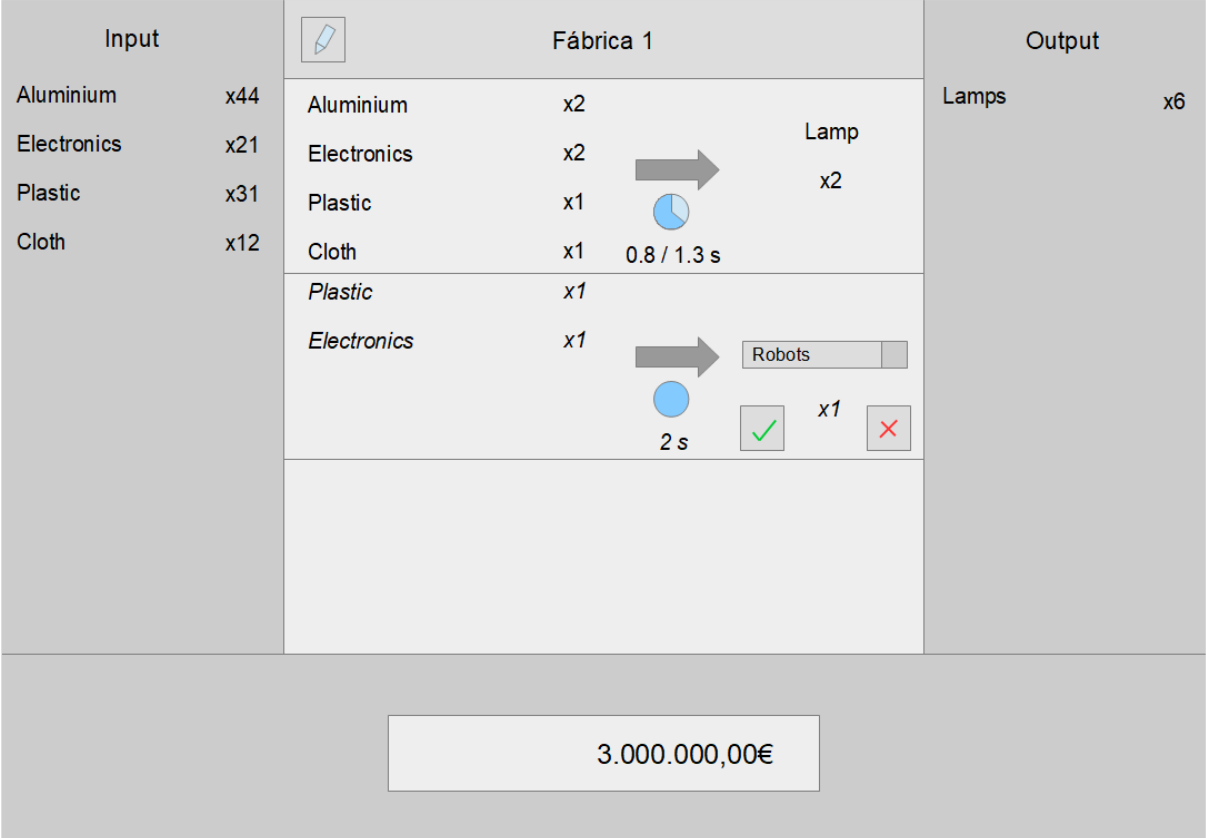


Ilustración 21. Pantalla de las fábricas cuando se clicca en "Nueva línea de producción" y se selecciona un producto con múltiples materias primas.









Truck	Trailer	+ Buy vehicle...
   	   	
<div data-bbox="574 905 1003 978" style="text-align: center;">3.000.000,00€</div>		

Ilustración 22. Pantalla de los depósitos de vehículos.






Truck		Trailer		<input type="button" value="✕"/> Buy vehicle...	
	Truck		Trailer		
	Capacity	1 unit	Capacity	2 units	
	Speed	90 km h	Speed	60 km h	
	Cost	10 €/km	Cost	15 €/km	
	Price	20.000 €	Price	35.000 €	
		<input checked="" type="checkbox"/>		<input checked="" type="checkbox"/>	
<input type="text" value="3.000.000,00€"/>					

Ilustración 23. Pantalla de los depósitos de los vehículos cuando se clica en "Comprar vehículo..."

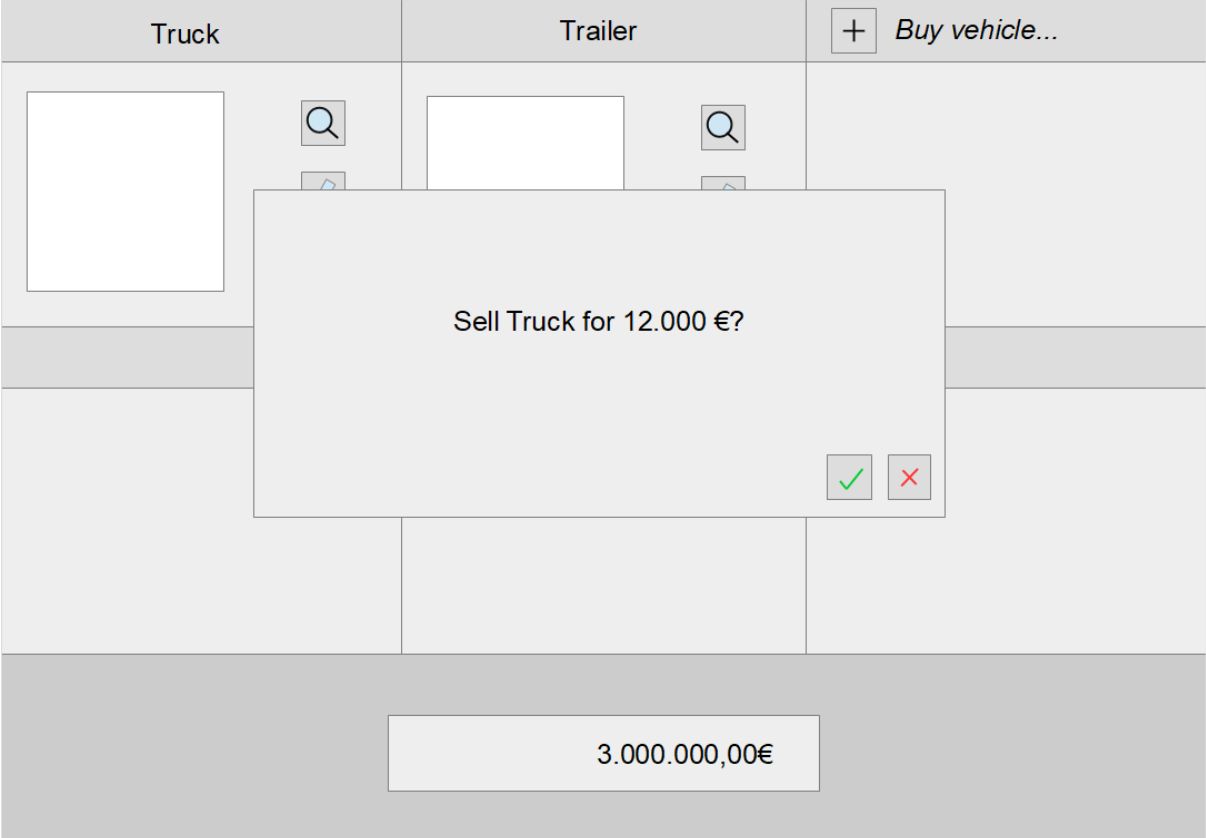


Ilustración 24. Pantalla de los depósitos de vehículos cuando se clicca en el icono de venta de un vehículo.

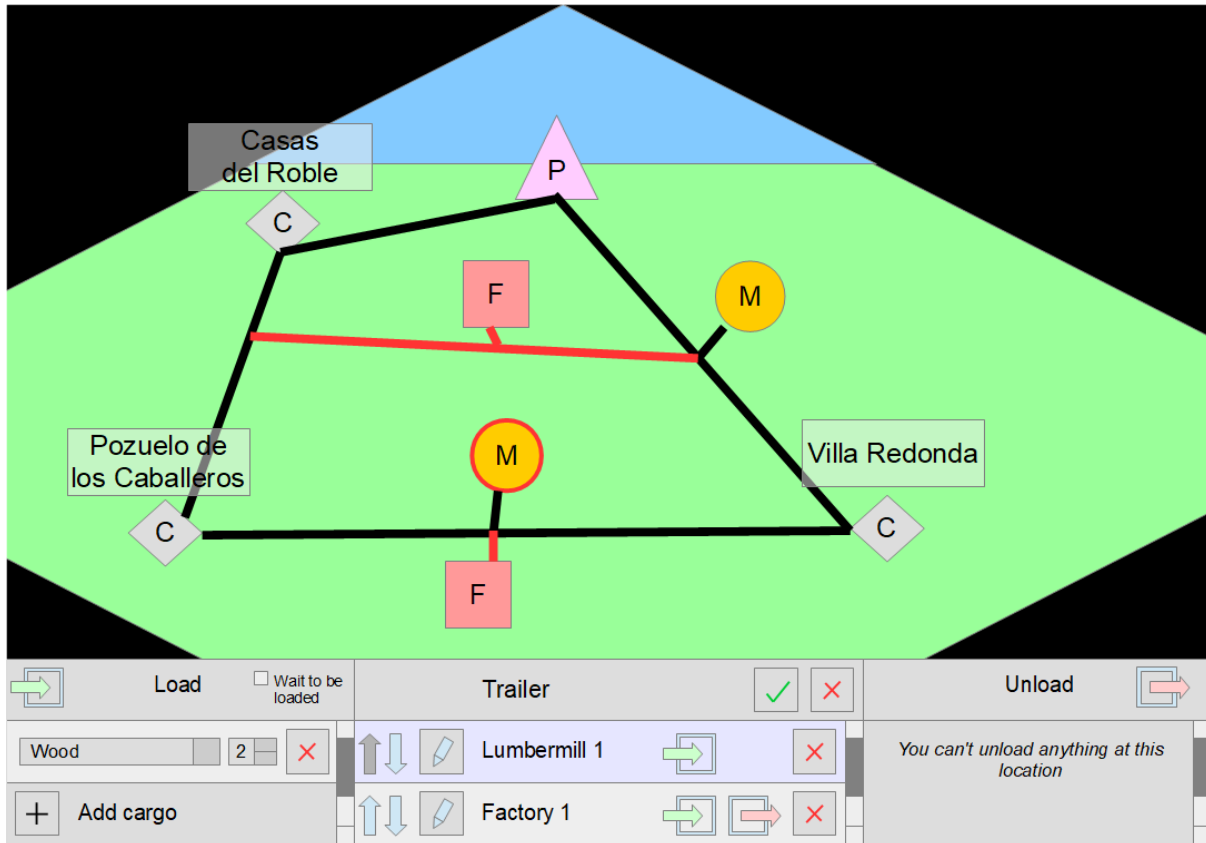


Ilustración 25. Pantalla de edición de rutas con un destino que no admite descarga seleccionado.

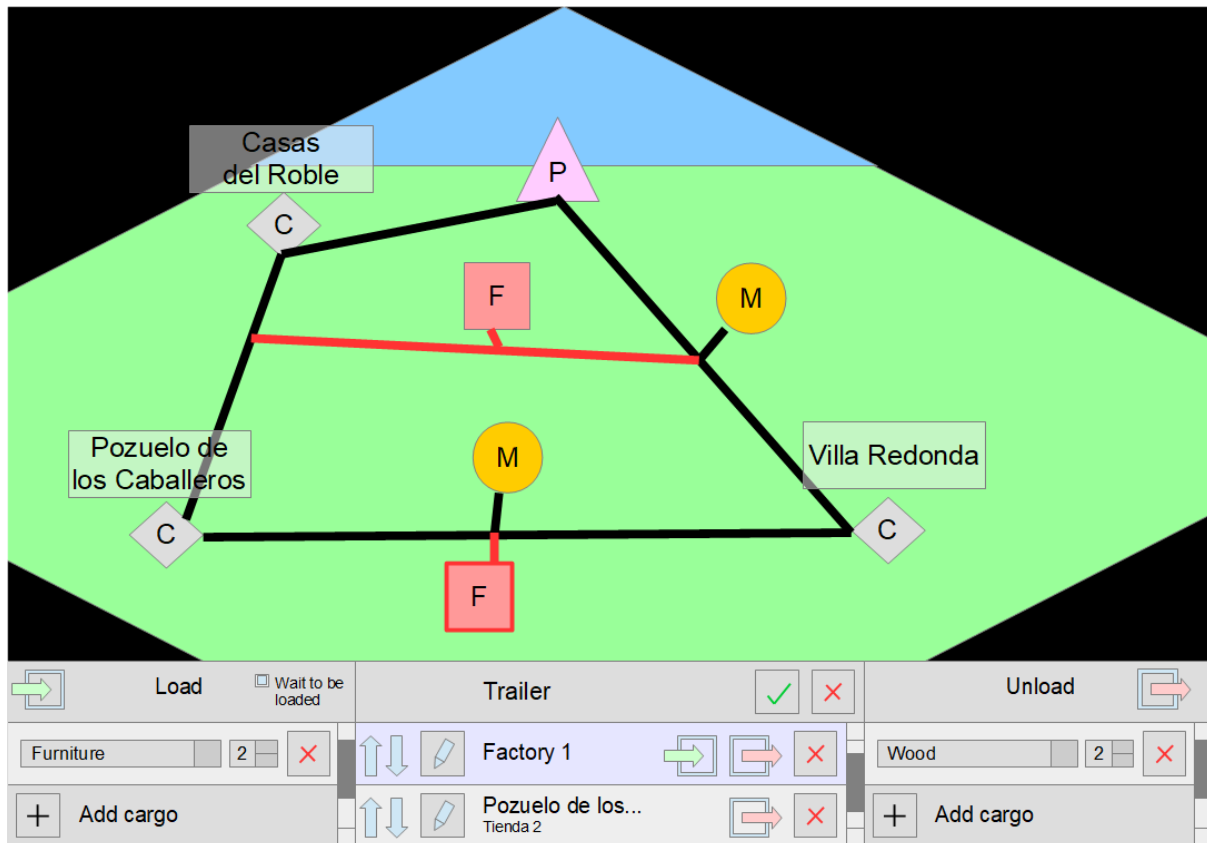


Ilustración 26. Pantalla de edición de rutas con un destino que admite carga y descarga seleccionado.

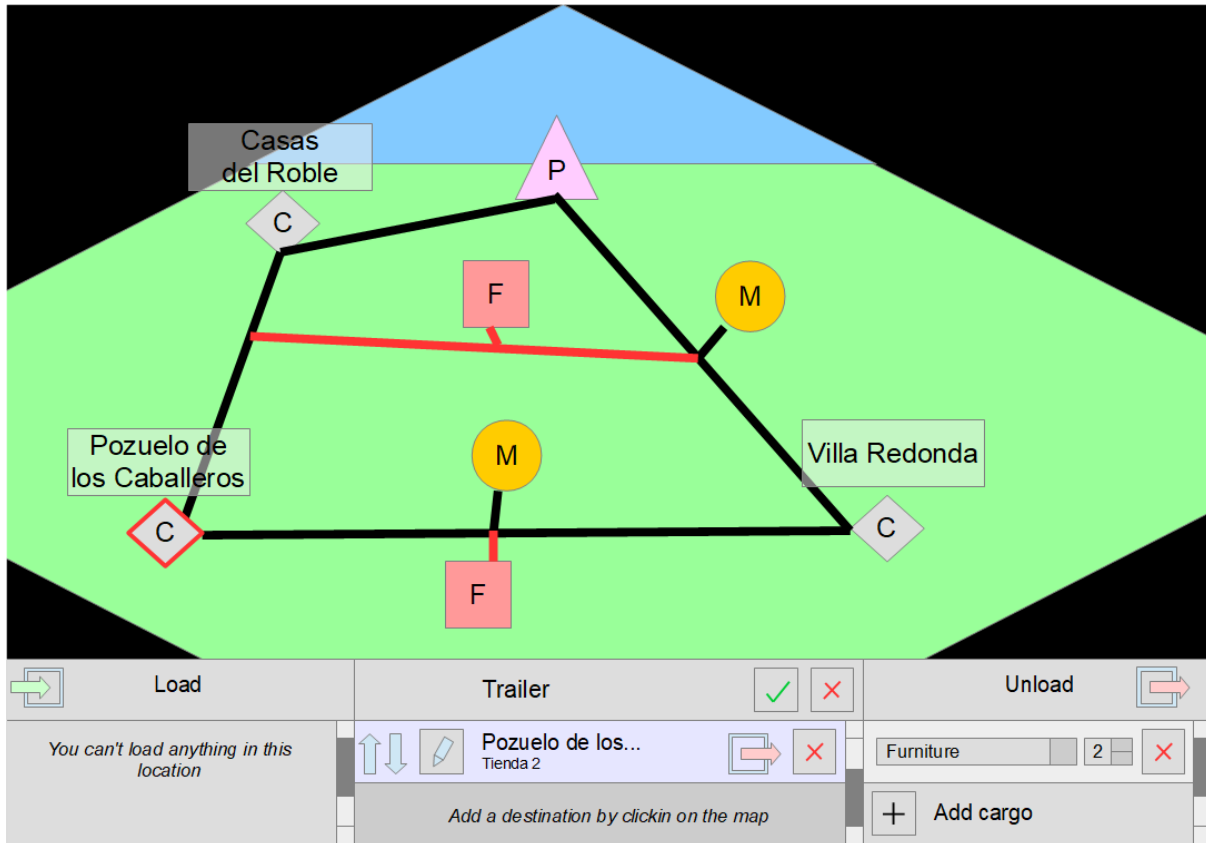


Ilustración 27. Pantalla de edición de rutas con un destino que no admite carga seleccionado, y que además es el último destino de la ruta.

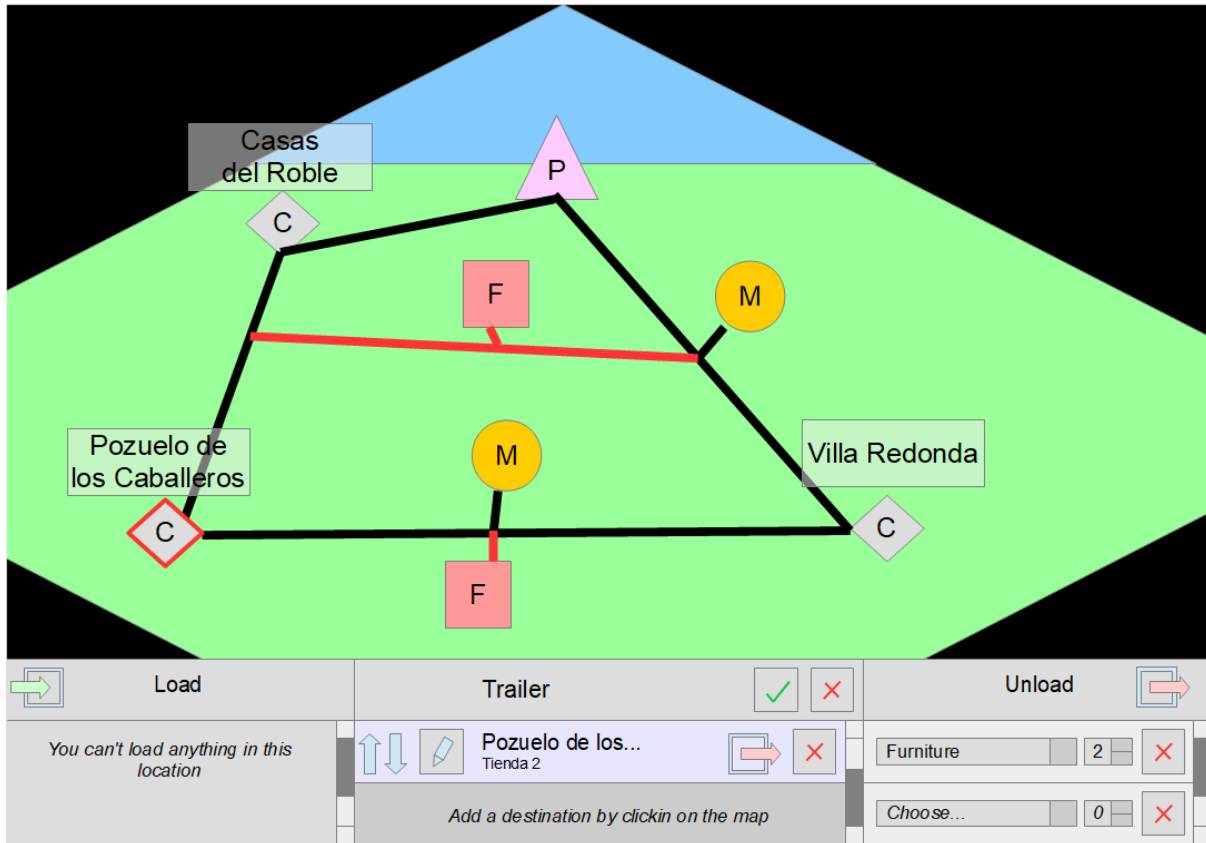


Ilustración 28. Pantalla de edición de rutas cuando se añade una operación de descarga.

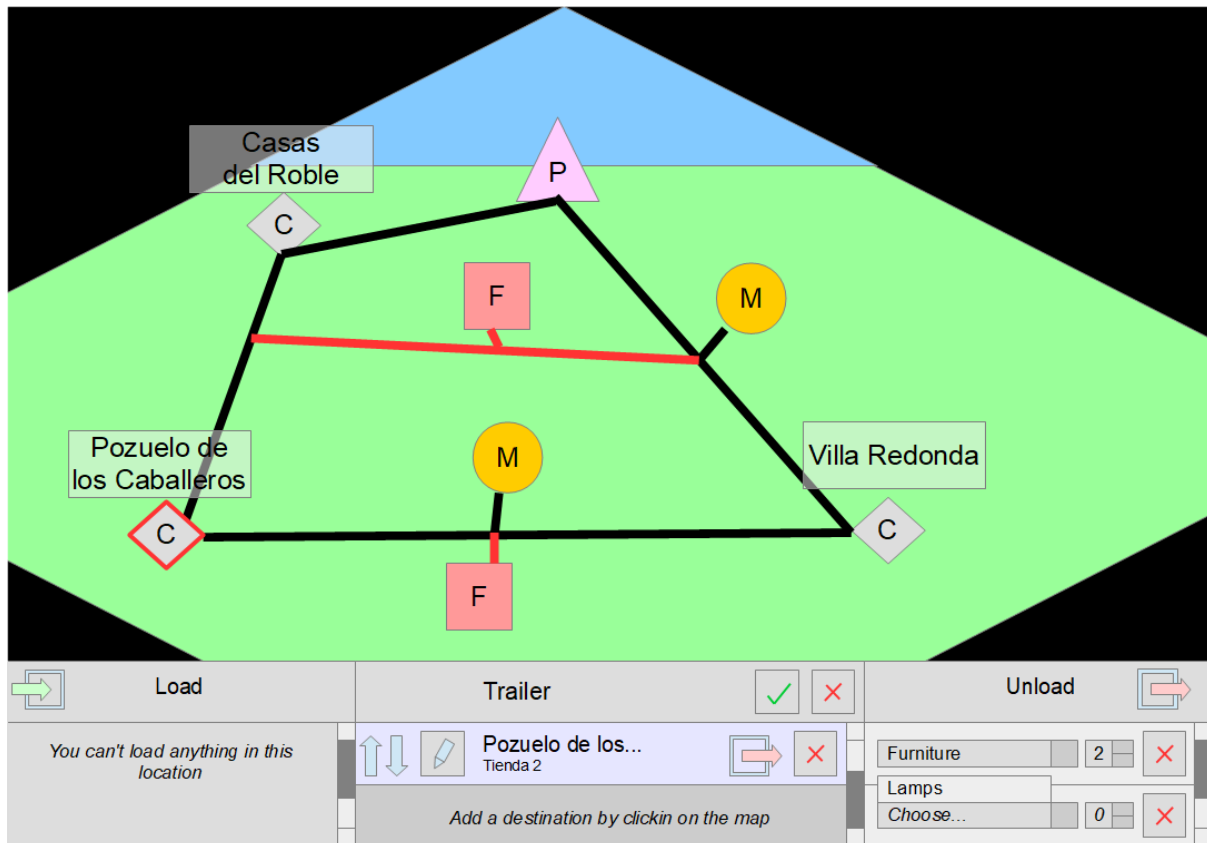


Ilustración 29. Pantalla de edición de rutas cuando se añade una operación y se clicca en el desplegable para ver la lista de productos que acepta el destino.

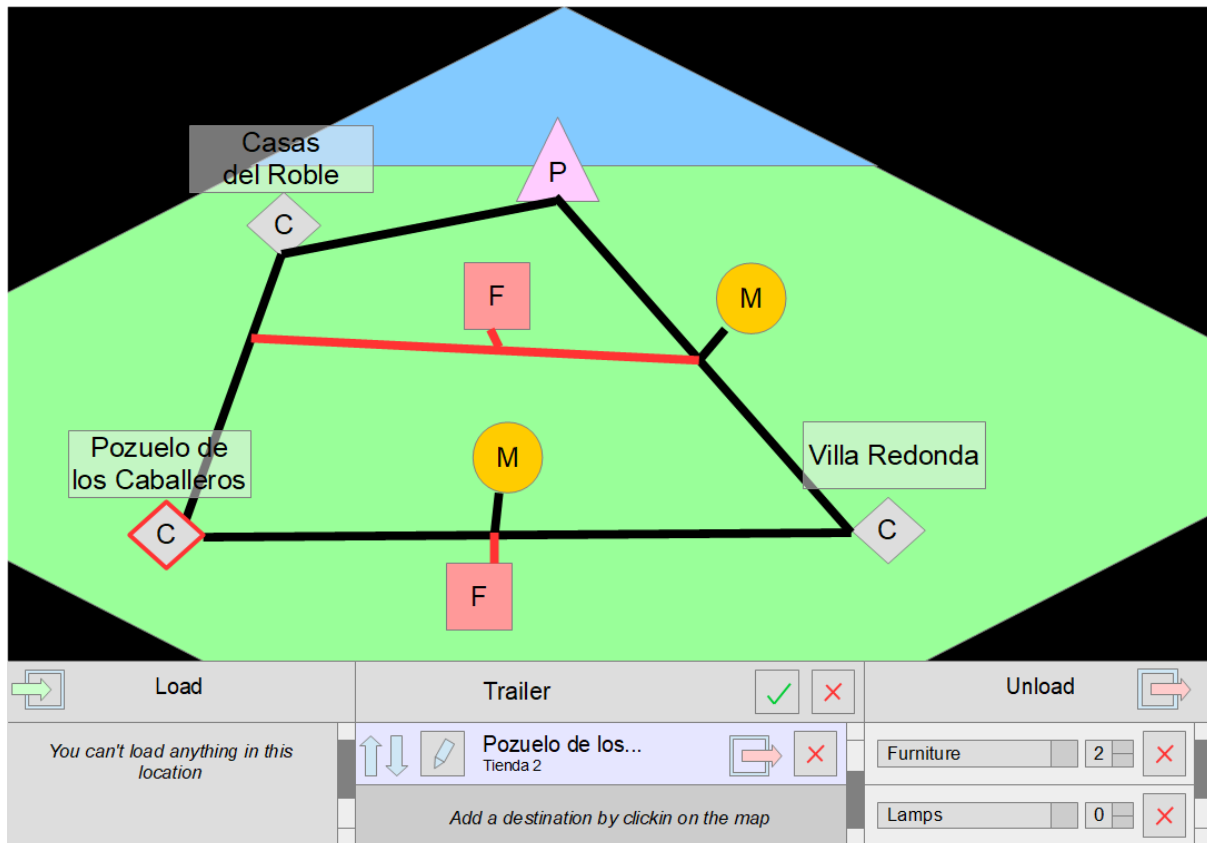


Ilustración 30. Pantalla de edición de rutas cuando se añade una operación pero se ha alcanzado la capacidad máxima del vehículo.



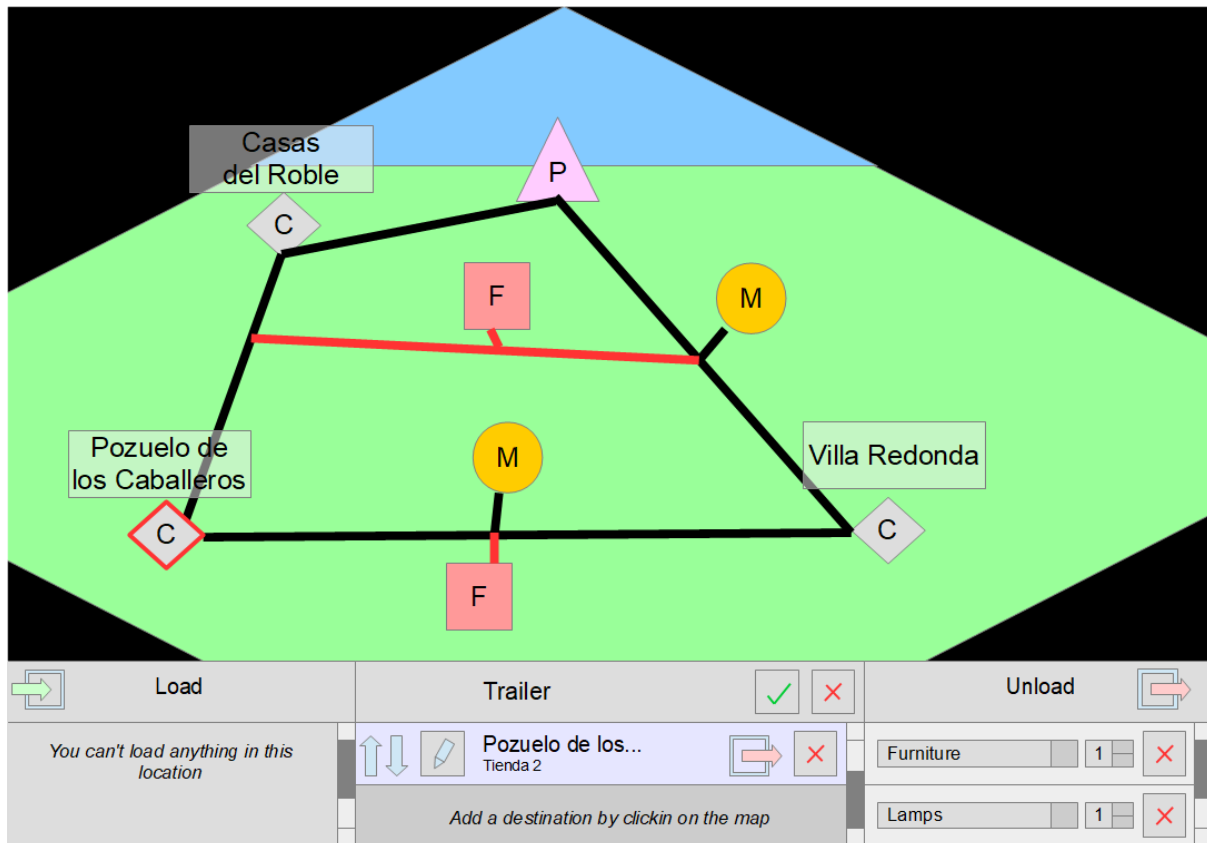


Ilustración 31. Pantalla de edición de rutas cuando se añade una nueva operación y se modifica la cantidad de descarga para poder descargar otro producto.

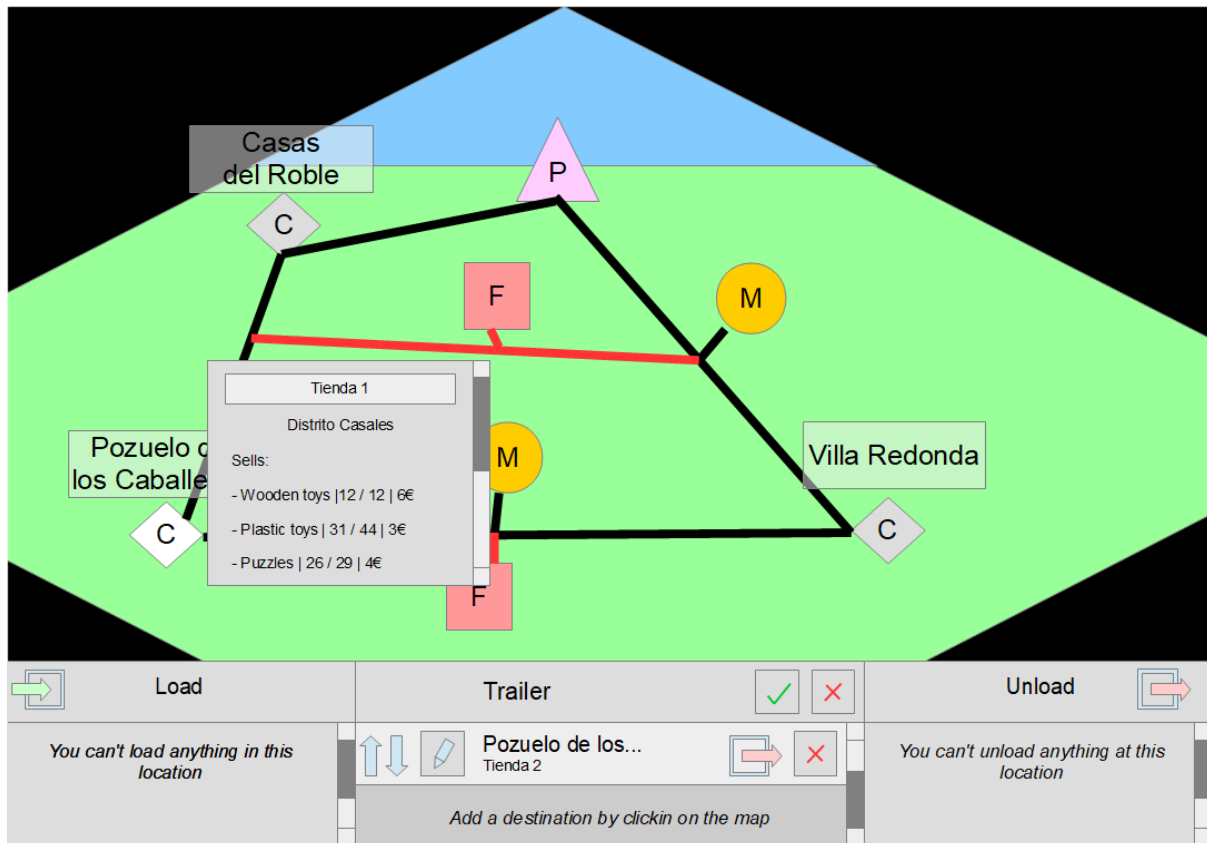


Ilustración 32. Pantalla de edición de rutas cuando se selecciona una ciudad como destino.

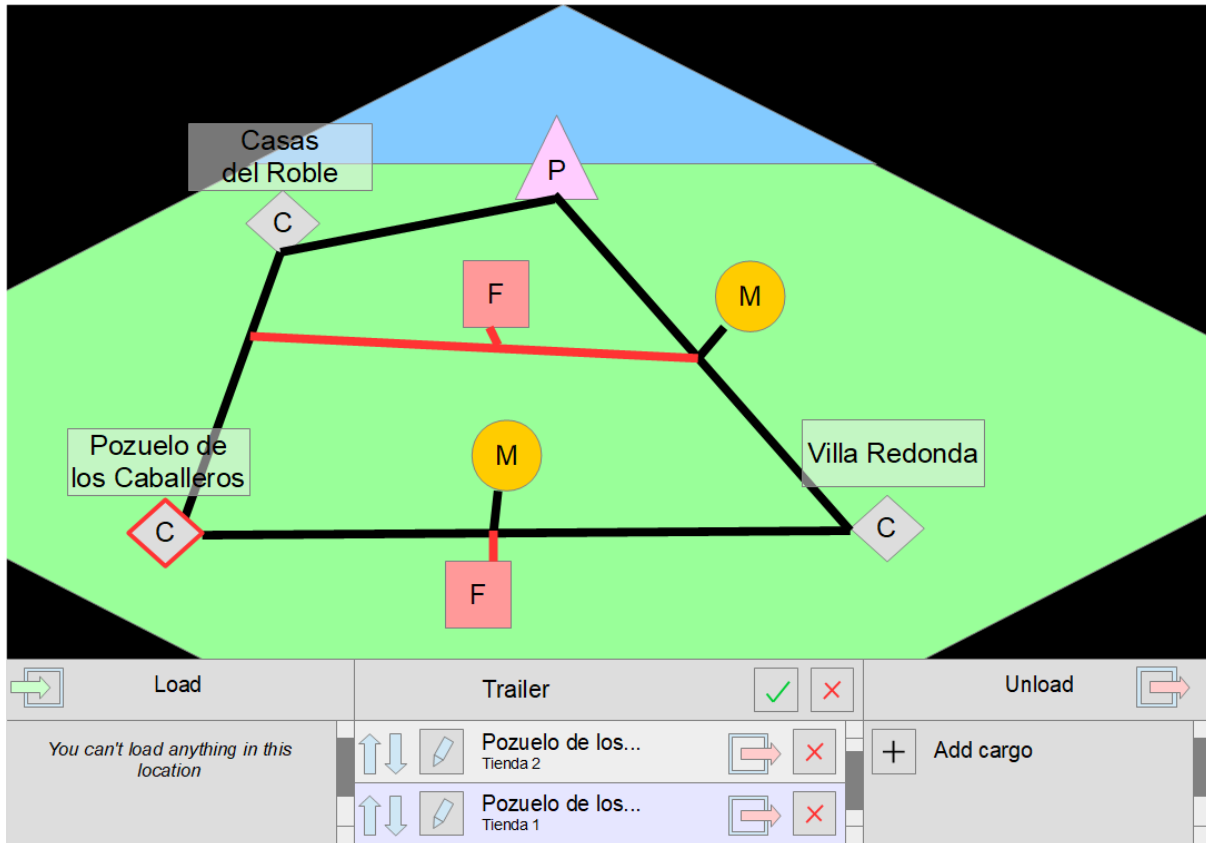


Ilustración 33. Pantalla de edición de ruta cuando se ha seleccionado una tienda de una ciudad como destino.

# Anexo. Encuesta.

---

# Encuesta Juego TFG Informática

¡Hola! Soy un alumno de último año de carrera de Ingeniería Informática, en la ETSINF de la Universitat Politècnica de València.

Me gustaría que probaras el juego que he desarrollado como proyecto final, para saber cómo es la experiencia de usuario y cómo se puede mejorar. **IMPORTANTE:** es una versión muy básica, y hay algunas acciones que no se pueden realizar todavía. Estas acciones suelen estar indicadas por un botón desactivado.

El juego se puede descargar en el siguiente enlace. Se ejecuta en Windows, por lo que deberás acceder desde el ordenador personal. Una vez lo descargues, se puede abrir haciendo doble click en él.

[https://drive.google.com/file/d/1KshO5kqvQSK1b29E4b8-Xt1Ev-k4g3l5/view?usp=drive\\_link](https://drive.google.com/file/d/1KshO5kqvQSK1b29E4b8-Xt1Ev-k4g3l5/view?usp=drive_link)

Por favor, sigue los pasos a continuación y contesta las preguntas con sinceridad. Este cuestionario es totalmente anónimo, y tus respuestas se usarán únicamente para conocer vuestra opinión sobre el juego.

¡Empecemos!

*\* Indica que la pregunta es obligatoria*

---

¡Pero antes que nada!

Me gustaría saber un poco de ti.

1. ¿Cómo te dominas con las nuevas tecnologías? \*

*Marca solo un óvalo.*

1   2   3   4   5

Me c      Las uso con naturalidad

2. ¿Cómo te dominas con los ordenadores, en concreto? \*

Marca solo un óvalo.

1 2 3 4 5

---

Me c      Los uso con naturalidad

3. ¿Con qué frecuencia juegas videojuegos? \*

Marca solo un óvalo.

1 2 3 4 5

---

Cas      Todas las semanas

### Juego a ciegas

Este juego nos permite ponernos en la piel de un empresario.

Abre el juego, e intenta jugar. Dedícale unos minutos. Investiga qué pantallas puedes abrir, qué botones hacen qué cosas, etc.

4. ¿Cómo ha ido la experiencia? \*

Marca solo un óvalo.

1 2 3 4 5

---

No r      Me ha gustado mucho

### Interfaz gráfica

5. ¿Ha habido alguna parte que te haya hecho quedarte atascado/a?

Selecciona todos los que correspondan.

- El menú principal
- La pantalla del mapa del juego
- El menú de los vehículos
- El menú para la ruta de los camiones.
- El menú de las fábricas.
- Otro: \_\_\_\_\_

6. Cuando querías realizar una acción, ¿cómo de frecuente encontrabas el botón que querías en el sitio en el que esperabas? \*

Marca solo un óvalo.

1 2 3 4 5

---

Casi siempre en el lugar en el que esperaba      Casi nunca en el lugar en el que esperaba

7. ¿Cómo de fácil te ha parecido navegar entre los menús? \*

Marca solo un óvalo.

1 2 3 4 5

---

Son      Son muy intuitivos

8. ¿Cuál de las siguientes afirmaciones se corresponde mejor con tu experiencia con la interfaz? \*

*Marca solo un óvalo.*

- La interfaz es muy similar a la de otros juegos parecidos.
- La interfaz es diferente a la de otros juegos parecidos. Es igual o más fácil de navegar.
- La interfaz es diferente a la de otros juegos parecidos. Es más difícil de navegar.
- No tengo experiencia con otros juegos parecidos, pero la interfaz me ha parecido fácil de navegar.
- No tengo experiencia con otros juegos parecidos, pero la interfaz me ha parecido difícil de navegar.

9. ¿Qué cosas cambiarías de la interfaz? ¿Añadirías algún botón? ¿Dónde? ¿Añadirías algún menú?

---

### Jugabilidad

10. ¿Qué te ha parecido la jugabilidad? \*

*Marca solo un óvalo.*

1   2   3   4   5

Muy      Muy entretenida

11. ¿Cuánto tiempo crees que podrías pasar jugando de forma seguida? \*

*Marca solo un óvalo.*

- 5 - 25 minutos
- 40 - 55 minutos
- 1 - 3 horas
- Más de 3 horas seguidas



12. ¿Qué te ha parecido la dificultad? \*

Marca solo un óvalo.

1 2 3 4 5

Muy      Muy difícil

13. ¿Qué crees que podría hacer el juego más entretenido?

Selecciona todos los que correspondan.

Adversarios controlados por una Inteligencia Artificial que compiten contigo por los recursos del mapa

Cadenas de producción más complejas, tener que fabricar componentes intermedios antes de fabricar el producto final

Más opciones de comercialización, poder abrir tiendas y vender en función de la demanda y popularidad de nuestra marca

Mayor dificultad, costes crecientes de las mercancías

Mayor sensación de progreso, desbloquear industrias más caras conforme aumentan los ingresos

Multijugador en línea

Evolución del mundo y de las ciudades conforme se comercia con bienes

Otro: \_\_\_\_\_

14. Si el juego incorporase todas las mejoras propuestas, ¿cómo de probable es que lo quisieras jugar?

Marca solo un óvalo.

1 2 3 4 5

Muy      Muy probable

15. ¿Cuánto pagarías por el juego mejoras?

*Marca solo un óvalo.*

- Solo lo jugaría si fuera gratuito
- Entre 0,01 y 5,00€
- Entre 5,01 y 10,00€
- Entre 10,01 y 20,00€
- Entre 20,01 y 40,00€
- Más de 40,00€

¡Ya casi hemos acabado!

16. ¿Cuál es tu género?

*Marca solo un óvalo.*

- Femenino
- Masculino
- Otro
- Prefiero no decirlo

17. ¿Qué edad tienes?

*Marca solo un óvalo.*

- Entre 1 y 8 años
- Entre 9 y 14 años
- Entre 15 y 19 años
- Entre 20 y 25 años
- Entre 26 y 35 años
- Más de 35 años

# Google Formularios

