



**UNIVERSIDAD CARLOS III DE MADRID
ESCUELA POLITÉCNICA SUPERIOR**

INGENIERÍA TÉCNICA EN INFORMÁTICA DE GESTIÓN

PROYECTO FIN DE CARRERA

**UN MODELO DE CONOCIMIENTO DEL JUEGO DEL
MONOPOLY**

Autor: Javier Aragón Zabalegui
Tutor: Carlos Linares López

Julio, 2006

UN MODELO DE CONOCIMIENTO DEL JUEGO DEL MONOPOLY

Tabla de contenidos

1	Introducción.....	6
2	Estado de la cuestión.....	10
2.1	Juegos	10
2.2	Juegos de N-agentes	14
2.2.1	Algoritmos de juegos para n-agentes	14
2.2.2	Algunos juegos de n-agentes	16
2.3	Monopoly	20
2.3.1	Reglas del juego.....	21
2.3.2	Servidor monopd.....	28
2.3.3	Atlantik	29
2.3.4	gtkAtlantik	29
2.3.5	Juegos comerciales.....	29
2.4	Conclusiones.....	33
3	Objetivos	36
4	Desarrollo.....	38
4.1	Análisis de requisitos.....	38
4.1.1	Gestionar partida.....	40
4.1.2	Mover ficha	42
4.1.3	Entrar en la cárcel	42
4.1.4	Coger tarjeta	43
4.1.5	Negociar.....	44
4.1.6	Declarar bancarrota	45
4.1.7	Salir de la cárcel.....	46
4.1.8	Pagar impuesto	48
4.1.9	Cobrar sueldo	49
4.1.10	Comprar propiedad.....	49
4.1.11	Hipotecar propiedad	50
4.1.12	Deshipotecar propiedad	51
4.1.13	Gestionar edificaciones de las propiedades.....	52
4.1.14	Alquilar propiedad	53
4.1.15	Consultar ayuda	54
4.1.16	Conclusiones sobre los casos de uso	55
4.2	Ontología	55
4.2.1	Diseño del tablero	57
4.2.2	Propiedades	60
4.2.3	El jugador en las casillas de tarjetas.....	62
4.2.4	Los propietarios	65
4.2.5	El papel de la banca.....	66
4.2.6	Las negociaciones entre jugadores.....	66
4.2.7	El cliente de IA	67
4.3	Discusión de las reglas del cliente de IA.....	68
4.3.1	Cliente de IA basado en reglas	68

4.3.2	Algoritmo de búsqueda sin información para procesos de hipotecas y construcciones	76
4.3.3	Cliente de IA basado en un comportamiento aleatorio.....	84
4.4	Generación aleatoria de números	87
4.5	Librería GNU Readline	88
4.6	Manual de usuario	88
4.6.1	Instalación de Street Master's	88
4.6.2	Ejecución	90
4.6.3	Jugar a Street Master's	90
4.7	Conclusiones.....	113
5	Resultados	117
5.1	Ontología	117
5.2	Evaluación de la eficacia del cliente de IA basado en reglas	117
5.2.1	Competición entre varios agentes de IA	117
5.3	Evaluación de la eficiencia del cliente de IA basado en reglas.....	134
6	Conclusiones	136
6.1	Ontología	136
6.2	Cliente de IA.....	137
6.3	Implementación	138
7	Líneas futuras	140
7.1	Cientes de IA	140
7.2	Conexión de Street Master's con el servidor Monopd	141
7.3	Interfaz gráfico	141
8	Bibliografía.....	144
8.1	Referencias bibliográficas	144
8.2	Referencias a direcciones de Internet	145
Anexos.....		148
A.	Tablero por defecto de Street Master's.....	148

Índice de diagramas

<i>Diagrama 1: Diagrama de casos de uso de Street Master's</i>	39
<i>Diagrama 2: Diagrama de clases de Street Master's</i>	56
<i>Diagrama 3: Diagrama de clases (diseño del tablero)</i>	57
<i>Diagrama 4: Diagrama de clases (diseño de las propiedades)</i>	60
<i>Diagrama 5: Diagrama de clases (diseño de las tarjetas)</i>	62
<i>Diagrama 6: Diagrama de clases (diseño de los propietarios)</i>	65
<i>Diagrama 7: Diagrama de clases (diseño de las negociaciones)</i>	66
<i>Diagrama 8: Diagrama de clases (cliente IA)</i>	67

Capítulo 1

Introducción

1 Introducción

Desde que en los años 1950 naciera el campo de la Inteligencia Artificial (IA) de la mano de Alan Turing¹, se ha convertido en un objetivo de los investigadores dotar a las máquinas de la capacidad de emular el funcionamiento del cerebro humano.

La IA ha pasado por fases de gran optimismo como la que se vivió en la conferencia de Darmouth en 1956 en la que se esperaban grandes avances a muy corto plazo, pero seguidamente han llegado fases de largos ocasinos tras la frustración de no cumplir las metas marcadas.

Actualmente, se trabaja en IA buscando soluciones a problemas de elevada complejidad. Estos problemas en muchas ocasiones se presentan en forma de juegos, ya que permiten crear situaciones idóneas para probar la eficacia de las nuevas ideas. Sin embargo, el estado actual es que el objetivo final de cumplir la famosa prueba de Turnig se encuentra tan lejos como cuando se formuló. Una de estas pruebas dice que *“existirá Inteligencia Artificial cuando no seamos capaces de distinguir entre un ser humano y un programa de computadora en una conversación a ciegas”*.

El problema que se trata en este proyecto se centra en la representación del conocimiento del famoso juego de Monopoly y el desarrollo de un cliente de IA basado en reglas que juegue de una forma competitiva usando dicho diseño.

Una propiedad muy deseable de la ontología creada, es que es fácilmente ampliable, lo cual permitirá definir nuevas casillas y partidas con cierta rapidez.

El Monopoly es un juego muy interesante en el que intervienen elementos de azar y procesos razonados en la toma de decisiones. Además, incluye problemas de cierta dificultad en IA como son las subastas y las negociaciones, lo cual abre una línea de trabajo muy importante.

¹ Turing inauguró el campo de la Inteligencia Artificial con su artículo “Computing Machinery and Intelligence” publicado en 1950

Asimismo, El juego creado, tiene la característica de ser un juego de n-agentes, un tipo de juego mucho menos estudiado que los de dos jugadores y por lo tanto con un amplio campo de estudio por delante.

El juego que se presenta en este trabajo incluye dos agentes de IA, pero sólo se espera de uno de ellos que razone de forma eficaz, ya que el otro tomará decisiones de una forma aleatoria y servirá para demostrar que el jugador basado en reglas es capaz de tomar decisiones acertadas.

El diseño creado permite definir nuevos clientes de IA usando otras técnicas más avanzadas. Estos nuevos clientes podrán competir con las ya definidos para medir su eficacia frente a los retos que se plantean en este juego.

Este proyecto se ha estructurado de la siguiente forma:

2. Estado de la cuestión: Se realiza un breve estudio sobre los juegos de n-agentes y las soluciones propuestas en IA, prestando especial atención al juego del Monopoly (reglas, distribuciones existentes e información de interés).
3. Objetivos: Se indica que se ha tratado de conseguir con la realización de este proyecto.
4. Desarrollo: Este apartado es el más relevante de la memoria e incluye toda la información importante sobre el desarrollo de la ontología y los clientes de IA presentados.
5. Resultados: En este capítulo se muestran los resultados obtenidos empíricamente después de finalizar el desarrollo de la sección anterior.
6. Conclusiones: Tras analizar todo el proyecto en su conjunto se emiten una serie de conclusiones que se recogen en este punto.
7. Líneas futuras de investigación y desarrollo de este proyecto.
8. Bibliografía: Referencias utilizadas a lo largo del proyecto, tanto en Internet como otras fuentes de tipo científico.

Capítulo 2

Estado de la cuestión

2 Estado de la cuestión

El objetivo de este capítulo es presentar el estado actual de las investigaciones y desarrollos basados en juegos de n-agentes (también conocidos como juegos multiagente) y discutir la conveniencia del estudio de juegos en el ámbito de la inteligencia artificial.

Se analizará el juego del Monopoly, exponiendo su origen, historia, instrucciones y relación con la inteligencia artificial.

Por último, se presentarán los juegos del Monopoly disponibles para ordenador, algunos de los cuales disponen de clientes de inteligencia artificial para competir.

2.1 Juegos

Desde mediados del siglo XX, cuando investigadores como Konrad Zuse, Claude Shannon y Alan Turing comenzaron sus investigaciones sobre diversos juegos como el ajedrez, la IA artificial ha tenido una gran evolución en este campo.

La investigación en el campo de los juegos resulta de gran utilidad en el área de la IA, tanto por su aplicación en diversas situaciones del mundo real, como en el propio desarrollo de los juegos. Algunos campos de aplicación de la teoría de juegos son la economía, la biología y las ciencias políticas [Binmore, 1994].

Además, los investigadores encuentran el tema muy atractivo por la naturaleza abstracta de los juegos, lo cual hace que se incrementen las investigaciones, y por lo tanto los resultados. Existen juegos muy complejos de resolver lo cual aporta un gran interés en la búsqueda de soluciones.

En ocasiones, la palabra “juego” se puede tomar como algo trivial, sin embargo, la teoría de juegos se utiliza en situaciones muy serias, como la adopción de contramedidas en situaciones de quiebra, la subasta del espectro radioeléctrico, la toma de decisiones en la fijación de precios y desarrollo de productos o la defensa nacional, situaciones en las que se

manejan grandes cantidades de dinero y ponen en juego vidas humanas [Russell y Norvig, 2003].

Una gran ventaja de los juegos, es que si no existe uno que se ajuste a las necesidades, se puede crear otro con las condiciones deseadas, apareciendo nuevos juegos que supongan nuevos retos para los algoritmos [Schaeffer, 2002], lo cual permite seguir mejorándolos y desarrollándolos.

Los juegos, por lo tanto son de gran utilidad en las investigaciones de IA. Una característica general muy apreciable de los juegos es el hecho de que se establezcan unas normas que deberán cumplir todos los participantes. De esta forma se puede asegurar que todos los agentes que intervienen en una partida de cualquier juego se van a regir por los mismos principios, lo cual permite intentar predecir el comportamiento de los adversarios.

En el mundo real, muchos de los problemas se solucionan gracias a la toma de decisiones por parte de la persona responsable. Esto ocurre también en el mundo de los juegos. En ocasiones, el juego presenta una gran cantidad de ramificaciones en la búsqueda de la solución óptima, lo que hace imposible llegar a ella. En estos casos, los juegos requieren la capacidad de tomar alguna decisión [Russell y Norvig, 2003]. Con frecuencia, esta decisión no será óptima, pero debe ser la mejor posible con los datos que se conocen.

Las aplicaciones en el mundo real de las investigaciones basadas en juegos son muy interesantes, pero además, existe un gran mundo de desarrollo enfocado a los videojuegos, una industria en expansión que ha crecido de forma imparable desde su invención a mediados del siglo XX.

Continuamente se intenta simular la inteligencia humana mediante máquinas, y los videojuegos son el entorno ideal para desarrollar la inteligencia humana [Schaeffer, 2002].

Además de lo ya mencionado, los juegos son muy útiles para refinar las técnicas de la IA [Nils J. Nilsson, 1998] y las investigaciones sobre ellos han generado nuevas ideas sobre cómo hacer un buen uso del tiempo. Los juegos castigan la ineficiencia con severidad [Russell y Norvig, 2003].

En cuanto al estado actual de la integración de la IA en los juegos, Jonathan Shaeffer se muestra contundente en sus valoraciones:

“El estado actual del arte en el desarrollo de caracteres realistas se puede describir como un ser primitivo, siendo la norma general el uso de sistemas de reglas simples y máquinas de estados finitos. Esto empieza a cambiar en algunas empresas que reconocen la importancia de la IA” [Shaeffer, 2002].

Caracterización de los juegos

La *teoría de juegos* clasifica los juegos en muchas categorías que determinan qué métodos particulares se pueden aplicar para resolverlos.

Las categorías más comunes incluyen:

- Juegos dinámicos

Un juego se considera dinámico si el fin perseguido por el resto de los oponentes resulta perfectamente conocido a cada uno de ellos. En otro caso no lo es.

- Juegos simétricos y asimétricos

Un juego es simétrico si la recompensa por jugar una estrategia determinada depende sólo de la estrategia que utilicen el resto de los jugadores, independientemente del jugador que realice la acción. Por lo tanto, para que un juego sea simétrico, deben poder intercambiarse los jugadores sin que esto afecte a las recompensas de las estrategias.

El caso contrario es aquel en el que la recompensa de usar una estrategia dependerá del jugador que la realice. Se conoce a estos juegos como juegos asimétricos.

- Juegos de suma nula

En un juego de *suma nula*, la suma de todos los beneficios y pérdidas de todos los jugadores debe ser cero, de manera que las pérdidas de un jugador, impliquen las ganancias de otro. Algunos ejemplos conocidos de juegos de suma nula son el ajedrez y el go.

Si no se cumple la condición indicada en el párrafo anterior, se estará hablando de un *juego de suma no nula*, tradicionalmente más difíciles de tratar y resolver. La mayoría de los ejemplos reales en negocios son de este tipo, ya que algunas decisiones tienen como desenlace resultados netos mayores o menores que cero.

- Juegos cooperativos

Los juegos cooperativos se caracterizan por la existencia de un contrato que se puede hacer cumplir entre jugadores o agentes.

- Juegos simultáneos y secuenciales

Los *juegos simultáneos* son aquellos en los que los jugadores ejecutan sus acciones a la vez, o sin conocer lo que han hecho el resto de los jugadores.

Si se debe mantener un turno, se tratará de un *juego secuencial*. Muchos juegos de mesa son de este tipo, mientras que los considerados en la disciplina matemática de "Teoría de juegos" son simultáneos [Binmore, 1994].

- Juegos de información perfecta

Los *juegos de información perfecta* son aquellos en los que no interviene el azar, como por ejemplo las damas y el ajedrez.

Si en un juego interviene el azar, se considera de *información imperfecta*.

- Juegos de información completa

Los *juegos de información completa* se caracterizan porque todos los jugadores poseen toda la información sobre las estrategias y las recompensas de los otros jugadores. De esta forma todos los jugadores tendrán la misma información sobre el juego.

El caso contrario se denomina *juego de información incompleta*.

Esta clasificación es muy completa pero usualmente no se utilizarán todas las características para definir un juego. La más normal es identificar

si el juego es o no de información completa, de suma nula y de información perfecta.

Por último, se considera importante identificar el número de jugadores (o agentes) que intervienen en un juego. Los juegos más investigados son aquellos en los que intervienen dos agentes, como son el ajedrez [Newborn, 1997], las damas [Shaeffer, 2002], el go [Muller, 1995] o el backgammon [Tesauro, 1995].

El siguiente apartado se ha dedicado en exclusiva a discutir los juegos de n-agentes.

2.2 Juegos de N-agentes

En un juego de n-agentes, además de las decisiones propias de un agente que sólo busca su beneficio personal, sin contar con nadie más, se pueden dar colaboraciones entre varios agentes que se denominadas *alianzas*. Una alianza entre varios agentes puede surgir por varios motivos: varios agentes colaborarán para debilitar a otro agente, alcanzar una meta común, etc.

En ambos casos los agentes se moverían por el interés común y la alianza surgiría de un comportamiento puramente egoísta [Russell y Norvig, 2003]. Existen complicaciones que se deben tener en cuenta al crear alianzas, como por ejemplo el hecho de romperla al alcanzar el objetivo, lo cual daría a entender que la alianza es frágil y podría causar dificultades para lograr nuevas alianzas en un futuro. No es el objetivo de esta memoria profundizar en este tema, simplemente se quiere mencionar el juego *FreeCiv* (o su versión comercial *Civilitation*) como un ejemplo de este posible comportamiento [López, 2004].

2.2.1 Algoritmos de juegos para n-agentes

Los algoritmos de juegos de n-agentes no contemplan las condiciones de negociaciones y alianzas, centrandó su objetivo en la búsqueda de una estrategia óptima en problemas de suma nula, información perfecta y completa en presencia de N agentes que persiguen el mismo objetivo.

Los algoritmos para los juegos de varios jugadores son una generalización de los desarrollados para dos agentes. En este apartado se muestra la evolución del algoritmo minimax y la poda Alfa – Beta para ser útil en juegos de más de dos jugadores.

Para poder aplicar los algoritmos que aquí se explican se debe establecer la condición de que los jugadores no podrán colaborar entre sí, de forma que no puedan formar alianzas para perjudicar a otros jugadores o alcanzar un fin común.

Luckhardt e Irani [Luckhardt et al., 1986] crearon una extensión del algoritmo minimax para juegos de múltiples jugadores que presentaron inicialmente para el juego del *Nim*: el algoritmo max^n . Para desarrollarlo, asumieron que los jugadores se turnan (juego secuencial) y que cada jugador tratará de alcanzar el rendimiento máximo en su estrategia devolviendo un valor máximo, y que dicho valor es independiente de las acciones de los otros jugadores.

A diferencia del algoritmo minimax para dos jugadores, max^n devuelve un vector de n -valores (*tupla*), donde n es el número de jugadores. De esta forma se conocerán las estimaciones de las acciones para todos los jugadores participantes.

El objetivo de este algoritmo es buscar el máximo valor para el jugador actual. Para ello se creará un árbol de búsqueda de forma que en cada nodo exista una tupla que integre los resultados de la función de evaluación aplicadas a cada jugador. Una vez conocidos todos los valores posibles, se deberá buscar el máximo valor para el jugador i en los nodos *hoja*, observando el valor que aparece en la posición i de la tupla.

Luckhart e Irani observaron que, para el jugador i , sólo es necesario evaluar la componente i -ésima de cada nodo. Sin embargo, esto no supone ningún ahorro computacional. Concluyeron precipitadamente que no se podrían hacer podas con más de dos jugadores sin realizar nuevas consideraciones.

Richard Korf explica como hacer podas con la única condición de que la suma de todos los pesos sea siempre igual a una misma cantidad [Korf,

1991]. Su trabajo ha sido extendido por su alumno Sturtevant [Sturtevant, 2000, Sturtevant, 2002].

Estos autores usaron los términos de poda profunda (*deep pruning*) y poda superficial (*shallow pruning*) para referirse a técnicas de evaluación parcial previas a las posibles podas.

2.2.2 Algunos juegos de n-agentes

Existen varios juegos de n-agentes que cuentan con algunas implementaciones. Entre ellos se pueden destacar los siguientes:

2.2.2.1 Risk

Risk es un juego de mesa comercial basado en turnos creado por la empresa Parker Brothers, una división de Hasbro.

Fue inventado a comienzos de la década del 1950 por el director de cine francés Albert Lamorisse.

Este juego comparte muchas características con los juegos de guerra, pero en comparación con otros juegos de guerra, es simple y abstracto. No pretende simular correctamente la estrategia militar, el tamaño del mundo, la logística de las campañas extensas o el azar del mundo real.

El Risk es un juego multiagente con intervención del azar, ya que se debe lanzar un dado y éste forma parte importante del éxito o fracaso de las estrategias, de información perfecta e información completa. Además es un juego de suma nula, ya que las pérdidas de tropas o territorios de un jugador, se convierten en una ganancia para otro.

No puede solucionarse de la mejor forma usando los algoritmos expuestos, ya que existe la posibilidad de que varios jugadores se alíen contra otros.

Existe una versión del juego (TGNet) que incluye, además, agentes inteligentes.



Imagen 1: TGNet, implementación del Risk

2.2.2.2 Póker

El juego de cartas de póker es el más popular del tipo de juegos denominado de apuesta, en el cual los jugadores con todas o parte de sus cartas ocultas hacen apuestas sobre una puja inicial, tras lo cual, la suma de apuestas es para el jugador o jugadores que tienen la mejor combinación de cartas.

Uno de los juegos de múltiples jugadores del que más investigaciones se realizan es el póker. Es un juego de información imperfecta en el que la toma de decisiones es la clave de la partida.

Jonathan Schaeffer dedica un capítulo de su libro *Chips challenging champions* a explicar los avances para jugar al póker desarrollados en el programa *Poki*.

Además de los métodos para evaluar las jugadas y elegir la estrategia adecuada al apostar, *Poki* usa técnicas de aprendizaje para construir modelos estadísticos de cada oponente, y se adapta dinámicamente para explorar los patrones y las tendencias observadas. El resultado es un programa que juega razonablemente bien al póker [Schaeffer, 2002].

Algunas formas del póker, sin embargo, se pueden considerar como juegos de 2 agentes.



Imagen 2: Poki, juego de poker

2.2.2.3 Bridge

El bridge es un juego de naipes para cuatro jugadores por parejas, habitualmente sentados alrededor de una mesa. El juego consiste en dos partes principales, la subasta y el carteo. Al Bridge se puede jugar de diferentes maneras. Pueden jugar 2 equipos de 4 jugadores una contra otra y también pueden organizarse competiciones por parejas que son las más habituales en los clubs de bridge, en las que un número indeterminado de parejas se enfrentan entre sí en lo que se conoce como *bridge duplicado*.

Matthew Ginsberg ha desarrollado un jugador de bridge que actualmente ostenta el título de campeón del mundo.

2.2.2.4 Sims

Existe un proyecto fin de carrera de la Universidad Carlos III de Madrid que se ha centrado en este juego. Éste proyecto es *AI-Live* [Roberto Adarve Afán de Rivera, 2006].

Los sims, un simulador de vidas humanas simplificadas, creado por Will Wright para Maxis en 2000, es el primer videojuego de este género. El juego consiste en diseñar personajes llamados *sims*, y dotarles de una casa que también hay que diseñar y posteriormente amueblar. Muchos *sims* vivirán en varias casas formando un barrio. El jugador humano controlará uno o varios *sims* que son suyos. Todos los sims que no son suyos, así como los que son suyos pero no quiere controlar, son controlados por un programa de inteligencia artificial que intenta jugar como lo haría un humano [Roberto Adarve Afán de Rivera, 2006].

Éste proyecto ha diseñado un motor de IA en el que se integra un sistema de producción de reglas (CLIPS) y otro en el cual se integra un

planificador de tareas (Prodigy) que son capaces de controlar un personaje creado dentro de la realidad simulada en el juego.



Imagen 3: *Los sims*

2.2.2.5 *Monopoly*

El proyecto actual se centra en el Monopoly, por lo que en este punto sólo se hará una ligera apreciación de la relación de este juego con la teoría de juegos aplicable a la IA. Más adelante se exponen las reglas y las explicaciones pertinentes sobre este juego.

Las características de Monopoly son:

- Juego de azar: El azar interviene al lanzar los dados y al tomar las tarjetas de los mazos. El éxito o fracaso de una estrategia dependerá de el resultado de estas acciones.
- Juego de información incompleta: Todos los jugadores conocen la situación de la partida, pero no pueden conocer las tarjetas que se encuentran en la cima del mazo.
- Es un juego de suma nula: En realidad existe la posibilidad de que la pérdida de un jugador no se convierta en beneficios para otro. Este caso se da al pagar (o cobrar) a la banca. Para convertir el juego en un juego de suma nula, se añade un jugador "ficticio" adicional, la banca. De este modo se considera el juego de suma nula.

Éste juego ofrece la posibilidad de que varios agentes colaboren, se alíen, con un objetivo común, por lo que al igual que ocurre con otros juegos como el Risk, no será posible resolverlo usando el método \max^n .

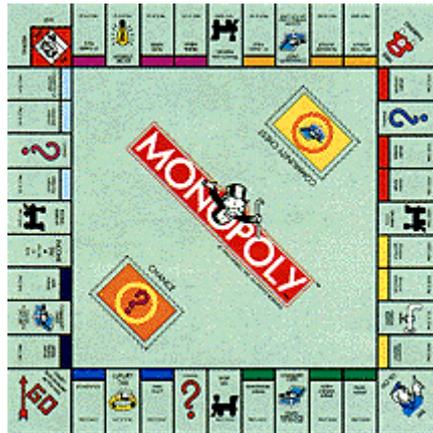


Imagen 4: Tablero del monopoly

2.3 Monopoly

Monopoly, también conocido en muchos lugares como *Turista*, es uno de los juegos más populares del mundo, habiéndose vendido más de 200 millones de ejemplares en 80 países de todo el mundo en 26 idiomas diferentes.

El origen del juego se remonta a inicios del siglo XX, cuando mucha gente creaba y modificaba diferentes juegos caseros como el *Landlord's Game* o el *Finance*, que sembraron las bases del actual Monopoly, patentado en Pennsylvania por Charles Darrow en 1933 durante la gran depresión económica. Darrow ambientó el juego en lugares de Atlantic City, fabricando a mano las 5000 primeras unidades. En 1935, cuando la demanda de juegos se disparó, tuvo que vender la patente a la empresa Park Bross (ahora propiedad de Hasbro) que continuó con la producción del juego hasta nuestros días.

Durante la expansión de Monopoly por el mundo se han personalizado versiones para todos los países donde se vende, adaptando tanto el idioma como las calles y la moneda con los que se juega. Por ejemplo, las últimas versiones de Monopoly en Europa ya están adaptadas al Euro.

El objetivo de Monopoly es convertirse en el jugador más rico a través de la compra, venta y alquiler de propiedades. A raíz del éxito de Monopoly han surgido otros juegos que comparten el mismo objetivo, como el *Palé* (muy conocido en España), juego idéntico al Monopoly del que sólo difiere

en las cantidades de dinero que se barajan; el Superpoly, idéntico en esencia y desarrollo al Monopoly; y el Hotel, que tiene un desarrollo muy diferente aunque la finalidad del juego es la misma.

Como es de esperar un juego tan popular ha servido de inspiración a empresas grandes y pequeñas de desarrollo software por lo se pueden encontrar varias versiones del juego en versión electrónica. Más adelante se comentarán aquellas más representativas.

2.3.1 Reglas del juego

La última revisión del reglamento de Monopoly data de 1994 a cargo de Parker Brothers. A continuación se expone integro el documento al que se hace referencia.

OBJETIVO: *El objetivo del juego es convertirse en el jugador más rico a través de la compra, alquiler y venta de propiedades.*

EL EQUIPO: *El equipo se compone de un tablero, dos dados, fichas, 32 casas y 12 hoteles. Hay barajas de cartas para los espacios de SUERTE y CAJA DE COMUNIDAD. Hay una tarjeta de escritura de propiedad para cada una de las fincas, y vales que representan dinero.*

PREPARACIÓN: *Se pone el tablero sobre la mesa colocando las cartas de suerte y caja de comunidad con la cara hacía abajo, en los espacios destinadas para las mismas en el tablero. A cada jugador se le entrega una ficha que representa a dicho jugador en sus viajes alrededor del tablero. A cada jugador se le dan también \$1500 divididos de la siguiente manera²:*

2 vales de \$500	2 vales de \$100
2 vales de \$50	6 vales de \$20
5 vales de \$10	5 vales de \$5
5 vales de \$1	

El resto del dinero lo recibe el banco.

EL BANQUERO: *Se elegirá como banquero un jugador que sea capaz de ser también un buen subastador. Si el banquero participa también en el juego, está obligado a mantener sus fondos personales aparte de los fondos del banco. Cuando juegan más de cinco jugadores, el banquero prefiere a veces limitarse a actuar tan sólo de banquero y subastador.*

EL BANCO: *El banco guarda, además del dinero del banco, las cartas de escritura de propiedad y las casas y hoteles antes de que los adquieran y*

² El documento se refiere a la versión estadounidense. En la versión española de Monopoly se reparten 150000 pesetas de la siguiente manera (versión previa a la entrada en vigor del euro):

2 billetes de 50000	2 billetes de 10000
2 billetes de 5000	6 billetes de 2000
5 billetes de 1000	5 billetes de 500
5 billetes de 100	

utilicen los jugadores. El banco paga los sueldos y gratificaciones, vende y remata propiedades a los jugadores y les entrega las cartas de escritura de propiedad de las mismas; les vende casas y hoteles a los jugadores y les presta, cuando esto es necesario, dinero sobre hipotecas de las propiedades.

Se le paga al banco todas las contribuciones, multas, préstamos e intereses y el precio de todas las propiedades que venda y remate.

El banco no “se arruina” nunca. Si se queda sin dinero puede emitir toda la moneda propia que se necesite, con sólo escribirla sobre cualquier papel corriente.

EL JUEGO: Empezando por el banquero, cada jugador echa los dados por turno. El jugador que logra el total mayor inicia el juego. Después de colocar su ficha en el ángulo marcado por la palabra SALIDA, echa los dados y hace adelantar su ficha en la dirección de la flecha el número de espacios que indican los dados. Cuando ha terminado la jugada le pasa el turno al jugador a su izquierda. Las fichas quedan en los espacios que han ocupado y siguen avanzando a partir de dicho punto cuando le toca el turno de nuevo a cada jugador. Pueden quedar a la vez dos o más fichas en el mismo espacio.

Según sea el espacio a que va a parar su ficha, el jugador tendrá derecho a comprar terrenos u otras propiedades, estará obligado a pagar alquiler, contribuciones, robar una tarjeta de suerte o de caja de comunidad, “váyase a la cárcel”, etc.

Si el jugador saca dobles, adelantará su ficha normalmente sea tantos espacios como puntos haya sacado, y quedará sujeto a los derechos o castigos que correspondan al espacio al que haya llegado. Después de esto volverá a echar los dados y adelantará su ficha del mismo modo. Si un jugador saca tres dobles seguidos, en vez de adelantar la ficha lo pondrá inmediatamente en el espacio marcado “En la cárcel” (véase LA CÁRCEL).

SALIDA (Salida): Cada vez que la ficha de un jugador va a parar o pasa sobre el espacio “SALIDA”, sin tener en cuenta si ha llegado a dicho espacio o pasado por el mismo como resultado de echar los dados o de robar una carta, el banquero le paga un “sueldo” de \$200³.

Sin embargo, los \$200 se pagan una sola vez por cada vuelta al tablero en la que se llegue al espacio “SALIDA” o se pase sobre el mismo. Si un jugador que pasa sobre “SALIDA” la primera vez que echa los dados va a parar a la caja de comunidad dos espacios más allá o a la casilla de suerte siete espacios más allá, si roba una carta “Avance hasta SALIDA”, recibe \$200 por pasar “SALIDA” la primera vez y otros \$200 por ir a “SALIDA” la segunda vez siguiendo las instrucciones de la carta.

COMPRA DE UNA PROPIEDAD: Cuando la ficha de un jugador va a parar a una propiedad sin dueño, este jugador tiene la opción de comprar dicha propiedad al banco por el precio que va impreso en la misma. El jugador recibe una carta de escritura de propiedad que lo acredita como dueño y que debe colocar boca arriba delante de sí mismo.

Si el jugador opta por no comprar, la propiedad es puesta inmediatamente a la venta en subasta por el banquero y es vendida al mejor postor. El comprador pagará al banco la cantidad estipulada en la licitación y recibirá la carta

³ En la versión española del juego el sueldo es de 20000 pesetas.

correspondiente de escritura de propiedad. Todos los jugadores pueden licitar, incluso el jugador que no aceptó la opción de compra al precio impreso en la propiedad. Cualquier precio puede servir de precio para iniciar la licitación.

LLEGADA A UNA PROPIEDAD CON DUEÑO: Si la ficha de un jugador llega a una propiedad que tiene dueño, el dueño de dicha finca le cobra el alquiler estipulado en la lista impresa en la tarjeta de escritura de propiedad correspondiente a la misma.

Si el solar está hipotecado no puede cobrarse alquiler. La hipoteca de la propiedad se indica volviendo boca abajo frente al dueño la tarjeta de escritura de propiedad que la representa.

Nota: Si en el solar hay una o varias casas el alquiler es mayor del que corresponde un solar sin construcción.

Es sumamente ventajoso poseer las escrituras de propiedad de todas las fincas de un grupo completo determinado color porque esto autoriza al dueño a cobrar doble alquiler por los solares sin construir de dicha propiedad. Esta regla se aplica a los solares no hipotecados, aún en el caso de que algún otro solar del mismo grupo esté hipotecado.

Es mucho más ventajoso ser dueño de casa y hoteles que de solares son construir porque los alquileres de los primeros son mucho más altos que los de los solares sin construir.

Si el propietario deja de exigir el pago del alquiler antes de que el segundo jugador que siga en turno eche los dados, no podrá cobrar dicho alquiler.

LLEGADA A LOS ESPACIOS DE SUERTE O CAJA DE COMUNIDAD: Cuando ello ocurre, el jugador roba la carta de encima de la baraja que se indique, y luego de seguir las instrucciones impresas en la misma, la vuelve a poner cara abajo debajo de dicha baraja.

La carta "Salir libre de la cárcel" se conserva hasta que se utilice y una vez utilizada se la vuelve a poner debajo de la baraja. Si el jugador que la posee no desea utilizarla, la puede vender en cualquier momento a otro jugador al precio que convenga.

IMPUESTO: Cuando el jugador llega al espacio de contribuciones tiene dos opciones: estimar su impuesto en la cantidad de \$200⁴ y pagarlo al banco o pagar el 10% de su patrimonio total al banco. El patrimonio total se calcula sumando el dinero en efectivo, el precio impuesto a las propiedades (hipotecadas o no) y el precio de costo de todos los edificios que posea.

El jugador debe decidir qué hacer antes de empezar a sumar el valor de sus bienes.

LA CARCEL: El jugador va a la cárcel cuando:

- (1) Su pieza va a parar al espacio marcado "váyase a la cárcel".
- (2) Roba una carta marcada "váyase a la cárcel".
- (3) Al echar los dados saca dobles tres veces seguidas.

Cuando se envía a la cárcel a un jugador no puede cobrar el sueldo de \$200 en dicho movimiento porque sin importar el lugar en donde se halle situada su

⁴ En la versión española los \$200 equivalen a 20000 pesetas.

pieza en el recorrido del tablero, tiene que ser trasladada directamente a la cárcel. Siempre que se le envía a la cárcel termina el turno de un jugador.

Si en el curso de una jugada normal la ficha de un jugador llega al espacio "Sólo de visita", el jugador no sufre ningún castigo, y sigue adelante de la manera normal cuando le toca de nuevo el turno.

El jugador puede salir de la cárcel de alguna de las siguientes maneras:

- (1) Sacando dobles en alguno de los tres turnos siguientes al de entrada a la cárcel. (Si consigue hacerlo, avanza inmediatamente la ficha por el número de espacios indicados al sacar los dobles. En este caso, a pesar de haber sacado dobles no usará el otro turno).
- (2) Usando la carta "salir libre de la cárcel gratis", si la posee.
- (3) Comprando la carta "salir libre de la cárcel gratis" a otro jugador y usándola.
- (4) Pagando una multa de \$50 antes de echar los dados en sus dos turnos siguientes⁵.

Si al llegar al tercer turno el jugador no ha sacado dobles, deberá pagar la multa de \$50. Entonces sale de la cárcel y deberá mover su ficha la cantidad de espacios que indique su tirada de dados.

Aunque esté en la cárcel el jugador puede comprar y vender propiedades, casas y hoteles y cobrar alquileres.

PARADA LIBRE⁶: El jugador cuya ficha va a parar a este espacio no reciben ningún dinero, propiedad ni premios de ninguna clase. Se trata simplemente de un lugar de descanso "gratis".

LAS CASAS: Cuando un jugador posee todas las propiedades de un grupo del mismo color puede comprar casas al banco y levantarlas en dichas propiedades.

Si compra una sola casa, la situará en cualquiera de las propiedades del grupo. La segunda casa que compre y levante tendrá que situarse en uno de los solares no ocupados de este grupo completo de un solo color o de algún otro de dichos grupos de su propiedad.

En la escritura de propiedad del solar se indica el precio que debe pagarle al banco por cada casa.

Aún sobre los solares sin construir pertenecientes a un grupo completo de un solo color de su propiedad puede cobrar doble alquiler a algún adversario cuya ficha llegue a los mismos.

El jugador podrá comprar y construir, de conformidad con las reglas que anteceden y en cualquier momento, todas las casas que juzgue conveniente y le permita su situación económica. Pero debe construir en partes iguales (es decir que no podrá construir más de una casa en un solar de un grupo de un determinado color hasta haber construido una casa en cada solar de dicho grupo. Cuando lo haya hecho podrá entonces empezar a construir la segunda fila de casas y así sucesivamente hasta llegar al límite de cuatro casas por

⁵ En la versión española el precio a pagar por salir de la cárcel es de 5000 pesetas.

⁶ PARADA LIBRE se corresponde con la casilla de PARKING GRATUITO de la versión española.

cada solar. Por ejemplo, no podrá construir tres casas en un solo solar mientras exista tan sólo una casa en otro solar del mismo grupo).

De la misma forma que hay que construir en partes iguales, también hay que deshacerse en partes iguales de las casas al volverlas a vender al banco. (Ver VENTA DE PROPIEDADES).

LOS HOTELES: Antes de poder comprar un edificio de hotel, el jugador ha de tener cuatro casas en cada solar de un grupo completo de un solo color. Cuando lo logre podrá comprar del banco un hotel para levantarlo en cualquier solar de dicho grupo de un solo color, entregándole al banco a cambio del mismo las cuatro casas allí existentes y el precio del hotel que indique la escritura de propiedad. En cada solar no puede construirse más de un hotel.

LA ESCASEZ DE EDIFICIOS: Cuando el banco no tenga casas para vender, los jugadores que deseen construir tendrán que esperar para hacerlo a que algún otro jugador devuelva o venda al banco sus casas. Si se dispone de una pequeña cantidad de casas y hoteles y dos o más jugadores desean comprar un número superior al que tiene el banco, las casas y hoteles tienen que venderse en pública subasta al mejor postor.

VENTA DE PROPIEDADES: Todo jugador que sea dueño de solares sin edificar, ferrocarriles y servicios públicos (pero no de edificios) podrá venderlos a cualquier otro jugador en una operación privada y por la cantidad que convengan ambos. No obstante, no podrá venderse ningún solar que pertenezca a un grupo de color en el que haya algún otro solar que contenga edificios. Antes de que el dueño pueda vender un solar de dicho grupo de color, tendrá que vender al banco dichos edificios.

Los jugadores pueden volver a vender las casas y hoteles al banco en cualquier momento y a la mitad del precio que pagaron por los mismos.

Todas las casas de un grupo de color deben venderse una a una, parejamente y a la inversa de la forma en que se construyeron.

Todos los hoteles de un grupo de color pueden venderse al mismo tiempo y enteros o de una casa por vez (un hotel equivale a cinco casas). Deben venderse parejamente y a la inversa de la forma en que se construyeron.

HIPOTECAS: Las propiedades sin construcción pueden afectarse en cualquier momento y sólo a través del banco. Antes de poder hipotecar un solar en construcción, todos los edificios construidos en dicho solar y en todos los demás solares pertenecientes al mismo grupo de color deben ser vendidos al banco a mitad de precio. El valor hipotecario de cada finca va impreso en la carta de escritura de propiedad de la misma.

Sobre los solares o servicios públicos hipotecados no puede cobrarse alquiler, pero sin embargo puede cobrarse sobre las propiedades no hipotecadas pertenecientes al mismo grupo.

Para levantar la hipoteca, el propietario tiene que pagarle al banco la cantidad de la hipoteca más un 10% de interés. Cuando todas las propiedades de un grupo de color dejen de estar hipotecadas, el propietario podrá volver a comprar la casa al banco por el precio total.

El jugador que hipoteca su propiedad retiene la posesión de la misma, de modo que ningún otro jugador podrá obtenerla levantando para ello la hipoteca que tiene el banco. No obstante, dicho propietario puede vender su propiedad

hipotecada a otro jugador a cualquier precio que convengan. El nuevo propietario, podrá, si lo desea, levantar inmediatamente la hipoteca, pero para esto tendrá que pagarle al banco el capital de la hipoteca más el 10% de interés. Si no desea levantar inmediatamente la hipoteca, tendrá que pagar al banco el 10% de interés en el momento que compre la propiedad de otro jugador y, si más tarde decide levantar la hipoteca tendrá que pagarle al banco la cantidad correspondiente a la hipoteca más otro 10 % de interés.

QUIEBRA: Un jugador entra en quiebra cuando debe más a otro jugador o al banco que lo que puede pagar. Si es deudor de otro jugador tiene que entregarle a dicho jugador todos sus bienes y retirarse del juego. Al efectuar dicha liquidación si posee casas u hoteles tiene que devolverlos al banco a cambio de dinero, por un cantidad igual a la mitad del precio que pagó por ellos, y entregarle el dinero al acreedor. Si posee propiedades hipotecadas también le entregará dichas propiedades a su acreedor, pero en este caso el nuevo propietario tiene que pagarle al banco inmediatamente los intereses del préstamo que son el 10% del valor de la propiedad. Luego de hacerlo, el nuevo propietario podrá, a su elección, pagar la cantidad correspondiente al capital y terminar la hipoteca o retener la propiedad y tomar la decisión de levantar la hipoteca más adelante. Si retiene de esta forma la hipoteca hasta un turno subsiguiente, cuando levante la hipoteca deberá pagar de nuevo intereses.

Si el jugador es deudor del banco, en vez de serlo de otro jugador, (debido a que no puede pagar las contribuciones o multas aún después de vender las construcciones y propiedades hipotecadas) le entregará todos sus bienes al banco. En dicho caso el banco venderá inmediatamente en pública subasta todos los bienes así adquiridos con excepción de los edificios. El jugador quebrado tiene que retirarse inmediatamente del juego. El último jugador que queda gana el juego.

MISCELÁNEA: Los jugadores sólo pueden tomar prestado dinero del banco mediante la hipoteca de sus propiedades. Ningún jugador puede pedir prestado o prestarle dinero a otro jugador.

REGLAS PARA UN JUEGO CORTO (60 a 90 minutos)

Hay tres diferencias en las reglas para este “juego corto”:

- 1. Durante la preparación el banquero mezcla el mazo de tarjetas de escritura de propiedad. El mazo es entonces cortado por el jugador a la izquierda y el banquero da las tarjetas de escritura de propiedad, dos a la vez a cada jugador (incluyéndose a sí mismo, si es a la vez jugador y banquero. Los jugadores que reciben las cartas deben pagar inmediatamente al banco el precio impreso de las dos propiedades así adquiridas. Se empieza a jugar entonces de la misma manera que en el juego común.*
- 2. En este juego corto sólo es necesario tener tres casas (en vez de cuatro) en cada solar de un grupo completo de un color para poder comprar un hotel.*

El alquiler recibido por un hotel es igual que en el juego común.

El valor de devolución de un hotel es siempre la mitad del precio de compra que, en este juego, es de una casa menos que en el juego común.

3. *FIN DEL JUEGO. El primer jugador en declararse en quiebra se retira del juego, al igual que en el juego común. Sin embargo, al producirse la segunda quiebra el juego se termina y el jugador en quiebra entrega a su acreedor todo lo de valor que posea, incluyendo edificios y otras propiedades, ya sea que el acreedor sea un jugador rival o el banco.*

Cada uno de los restantes jugadores calcula el valor de sus propiedades:

- (1) Dinero en efectivo*
- (2) Solares, servicios públicos y ferrocarriles de su propiedad, al valor impreso en el tablero.*
- (3) Toda propiedad que tenga hipotecada a mitad del precio impreso en el tablero.*
- (4) Casas, al precio de compra*
- (5) Hoteles, al precio de compra, incluyendo el valor de las tres casas entregadas en pago.*

El jugador más rico gana.

OTRO BUEN JUEGO CORTO

EL JUEGO CON LÍMITE DE TIEMPO: Antes de empezar, establézcase una hora de terminación, llegada la cual el jugador más rico en ese momento ganará. Antes de comenzar el juego se mezcla y corta el mazo de escrituras de propiedad y el banquero da dos escrituras de propiedad a cada jugador. Los jugadores pagan inmediatamente al banco el precio de las propiedades obtenidas.

2.3.1.1 Variaciones más populares en las reglas del juego

A pesar de que Monopoly tiene unas reglas oficiales establecidas, es muy común que cada jugador tenga sus propias reglas personalizadas por tradición, por desconocimiento de las reglas oficiales o simplemente porque de ese modo les resulta más divertido el juego. Sin embargo, las reglas oficiales del juego no han variado apenas desde que se inventó. Sólo se han modificado pequeños detalles como los valores de las casillas de impuestos o las cantidades de dinero a repartir al inicio del juego, pero sin modificar en ningún momento la esencia del juego. A continuación se exponen las modificaciones de las normas de Monopoly más extendidas.

COMPRA DE UNA PROPIEDAD: Un jugador cuya ficha alcance un solar que no sea propiedad de nadie podrá comprar dicho solar al banco por el precio que indique la misma. El jugador recibirá la escritura de propiedad de mano del banco. Esta variante de la regla no contempla la posibilidad de subastar la propiedad si el jugador en cuestión decide no comprarla, en cuyo caso el banco la guardará hasta que algún jugador caiga en esa casilla y decida comprarla.

PARADA LIBRE: En ocasiones, se decide que el pago de impuestos y los diferentes pagos que deban hacerse porque así lo indiquen las tarjetas de suerte o caja de comunidad, en lugar de ir al banco, se depositen en el centro

del tablero de juego. El objetivo de esta operación es que al caer sobre la casilla “parking gratuito” el jugador propietario de la ficha reciba todo el dinero acumulado en el centro del tablero, dando así una función a esta casilla que en las reglas oficiales sólo sirve como un descanso libre de pagos.

LAS CASAS: Una variante menos extendida que las dos mencionadas anteriormente implica que las casas no tengan que construirse de modo escalonado, aunque sí es necesario tener un grupo de propiedades del mismo color para poder construir. Siguiendo esta regla podría darse el caso de contar con un grupo de propiedades en las que un solar contenga tres casas y otra que no tenga ninguna.

En caso de que el banco no pueda abastecer con suficientes casas a los jugadores, en ocasiones se acepta la posibilidad de que un jugador compre o venda casas a otro. Las reglas oficiales del juego no especifican nada al respecto, no lo permiten explícitamente, pero tampoco lo prohíben.

LA CARCEL: En algunas ocasiones no se tiene en cuenta que un jugador deba pagar al salir de la cárcel después de tres tiradas en ella.

IMPUESTOS: Muchos jugadores de Monopoly olvidan la posibilidad de elegir pagar el 10% del capital y obligan a pagar 20.000 pesetas al caer en esa casilla. De esta manera se agilizará la partida evitando realizar engorrosos cálculos. En las aplicaciones desarrolladas para ordenador, esto no debe ser un problema ya que la propia aplicación realizará el cálculo.

2.3.2 Servidor monopd

Monopd es un proyecto de sourceforge⁷ que consiste en un servidor para jugar al Monopoly en red. Los clientes que se comunican con este servidor lo harán mediante comandos cortos y mensajes XML. Este servidor ha sido desarrollado para sistemas operativos Linux, ofreciéndose gratuitamente en Internet. Incluso existe un paquete Debian que incluye una versión de monopd. No se ha desarrollado ningún servidor de este estilo para otros sistemas operativos.

El servidor no consume una gran cantidad de recursos en la máquina en la que se lanza.

Esta aplicación se desarrolló con la idea de crear una gran red de jugadores de Monopoly. Actualmente existen dos importantes clientes para Linux que se conectan a este servidor: *atlantik* y *gtkAtlantik*.

Oligopoly, se presentó como el primer cliente para monopd en el sistema operativo Windows, pero actualmente está abandonado.

⁷ <http://sourceforge.net/projects/monopd/>

2.3.3 Atlantik

Atlantik⁸ es un cliente para KDE desarrollado por *Rob Kaper* que permite jugar a juegos de tablero similares al Monopoly en la red servida por *monopd*.

El propósito del juego de tablero atlantik es conseguir terrenos en las mayores ciudades de Norteamérica y Europa. Todos las modalidades de juego están servidos por *monopd*.

Uno de las modalidades de juego es muy similar al popular juego de tablero conocido como Monopoly.

Éste cliente permite personalizar las partidas según las preferencias de cada jugador.



Imagen 5: Cliente Atlantik para Monopd

2.3.4 gtkAtlantik

GtkAtlantik⁹ es un proyecto de software libre que aporta un nuevo cliente basado en GTK¹⁰ (GIMP toolkit) para conectarse al servidor *monopd*. Éste cliente cuenta con la propiedad de tener una versión para Microsoft Windows¹¹, lo cual permitirá jugar en la red *monopd* desde cualquier plataforma.

2.3.5 Juegos comerciales

Muchas de las versiones Monopoly para ordenador se distribuyen para el sistema operativo Microsoft Windows. A continuación se hace referencia a estos juegos, que en su mayor parte se distribuyen bajo licencia de pago.

2.3.5.1 Monopoly (Infogrames Interactive & Artech Studios)

Juego para ordenador creado por Infogrames Interactive y Artech Studios en 2002, con la colaboración de Hasbro. Ésta versión de Monopoly

⁸ Se puede conseguir en <http://kde-apps.org/content/show.php?content=10019>

⁹ Se puede descargar gratuitamente desde <http://www.robertjohnkaper.com/journal/20060303110521.html>

¹⁰ Es un grupo importante de bibliotecas o rutinas para desarrollar interfaces gráficas de usuario (GUI) para principalmente los entornos gráficos GNOME, XFCE y ROX de sistemas Linux. Es software libre, multiplataforma y parte importante del proyecto GNU. www.gtk.org

¹¹ Se puede descargar desde <http://libertonia.escomposlinux.org/story/2004/2/8/171431/1770>

para ordenador proporciona una versión completa del juego con las reglas oficiales del mismo y con la posibilidad de personalizar un gran número de ellas (parking gratuito, precio de los impuestos, número de casas por hotel, opción de partida corta...).

El juego también recoge la posibilidad de elegir entre los tableros de Monopoly existentes en los diferentes países en los que existe el juego con sus respectivas monedas (en España por ejemplo se puede elegir entre pesetas y euros).

Monopoly también ofrece la posibilidad de guardar partidas y datos de los jugadores así como jugar partidas on-line.

La implementación del juego soporta hasta 6 jugadores, pudiendo ser personas humanas o clientes basados en inteligencia artificial. En el caso de que los jugadores sean clientes de inteligencia artificial se permite incluso elegir entre 3 niveles de dificultad: comprador novato, empresario y magnate.

Todo ello está ambientado en un elaborado tablero tridimensional con fichas de los jugadores animadas y juego comentado y acompañado por voz y música.

Por lo tanto, el juego se resume en una fiel representación informática del juego de Hasbro perfectamente acompañado de graficos en 3D y sonido, permitiendo la posibilidad de jugar contra el ordenador gracias al uso de la inteligencia artificial.

2.3.5.2 *Monopoly party (PS2)*

Esta aplicación es una original versión del Monopoly original creada por Infogrames en 2002, que además de ofrecer un juego fiel al Monopoly original, ofrece una nueva opción: Monopoly party.

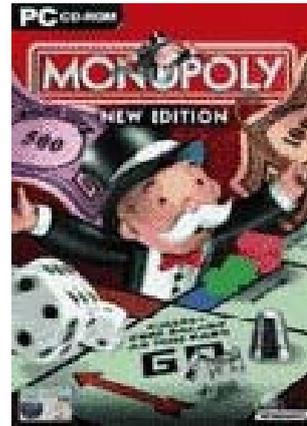


Imagen 6: *Monopoly (infogrames)*



Imagen 7: *Monopoly Party (PS2)*

Monopoly party consiste en un juego con las reglas oficiales del juego tradicional (aunque también podrán personalizarse), pero con la original idea de que todos los jugadores jueguen a la vez, evitando las aburridas esperas hasta que vuelva el turno a cada uno.

La versión normal del juego es semejante a la comentada anteriormente del Monopoly de ésta misma compañía.

2.3.5.3 *Billionaire II*

La compañía GameOn creó éste juego como una versión de Monopoly, pero más orientada a inversiones y negocios que el original de Hasbro. Por lo tanto, el juego no se ajusta a las reglas de Monopoly y tan solo tiene cierto parecido con él.



Imagen 8: *Billionaire II*

El juego ofrece la posibilidad de jugar en “el salón de la fama de Internet” o contra jugadores en el mismo computador, ya sean humanos o creados mediante inteligencia artificial, soportando un máximo de 4 jugadores.

Billionaire II tiene la posibilidad de jugar a 4 tipos de juego: jugar hasta que algún jugador sea billonario; hasta que algún jugador alcance una cifra establecida de dinero; hasta que se cumpla un tiempo establecido (ganando el jugador más rico); o jugar hasta la bancarrota de todos los jugadores menos el ganador, o hasta que éste llegue a \$100000 billones.

A diferencia de Monopoly, en Billionaire II se compran compañías en lugar de las típicas calles de Monopoly y se construyen infraestructuras (electricidad, agua...), apartamentos, tiendas y otros edificios como casinos en vez de las casas y hoteles del juego tradicional. Además, para jugar a este juego es necesario controlar las inversiones en bolsa, los préstamos del banco y las amortizaciones de las propiedades.

Por último, se concluye señalando que el tablero tiene, además de las propiedades, otras casillas que completan el juego con otros juegos alternativos, como los dados o las apuestas en carreras de coches.

2.3.5.4 *Moneylandia 2.00*

Este juego, creado por el programador David Eduardo Ramírez, es una versión gratuita y bastante fiel del juego Monopoly, pero con algunas variaciones en las reglas del mismo. Los cambios más importantes, a tener en cuenta, son tener que dar una vuelta completa al tablero para poder empezar a comprar propiedades, el hecho de no existir subastas en caso de no querer comprar una propiedad, y la ya mencionada variación de que, al caer en la casilla de parking gratuito (semáforo en este juego), el propietario de la ficha recoge toda la recaudación de impuestos pagados hasta ese momento.

El juego permite hasta 5 jugadores, dando la posibilidad de jugar contra la máquina gracias a la inteligencia artificial que ofrece el juego, si bien hay que decir que aún está en desarrollo por lo que comete algunos errores.

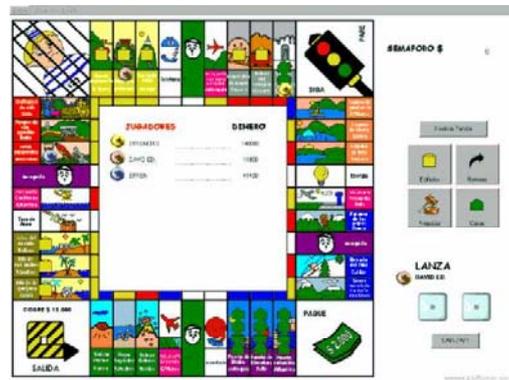


Imagen 9: *MoneyLandia 2.00*

MoneyLandia 2.00 aún no ha solventado todos los errores que tenía su predecesor (en ocasiones se confunde con los turnos, hace cosas extrañas o aparecen errores en tiempo de ejecución). Además, tampoco es posible negociar con otros jugadores a lo largo de la partida.

2.3.5.5 *Space Monopoly 1.0*

Versión estelar de Monopoly creada por Digital Lights en 2001. Esta versión del juego sigue casi por completo las reglas del juego tradicional, del que sólo difiere en que sustituye las calles por estrellas y estaciones espaciales. La única variación en las normas del juego que se ha detectado con respecto al juego original, es la inexistencia de subastas en las ocasiones que un jugador no quiere comprar una propiedad, algo que ocurre en bastantes versiones como se puede ver en este documento.

Incluye la posibilidad de jugar contra la máquina.



Imagen 10: *Space Monopoly 1.0*

2.3.5.6 Turbopoly 2.0

Turbopoly es un juego que ofrece al usuario la posibilidad de jugar al Monopoly en español con todas sus reglas oficiales con un máximo de 5 jugadores. No ofrece la posibilidad de jugar contra la máquina y, en cuanto a las subastas y negociaciones únicamente deberá indicarse su resultado final de modo que el juego sólo hará la transferencia de propiedades cuando sea necesaria.

2.3.5.7 Monopoly Here and Now

La agencia londinense DDB ha creado esta versión del juego que consiste en jugar a Monopoly en las calles reales de Londres. Para ello se cuenta con 6 taxis equipados con un sistema GPS que harán de fichas de los jugadores. Cuando un jugador transite por una propiedad de otro jugador deberá pagar el alquiler como ocurre con el juego tradicional.

El seguimiento del juego se hace mediante una página web¹² puesta en marcha por los promotores de la idea, donde se ofrece la posibilidad de jugar de forma gratuita.

El juego se creó con motivo de la celebración del 70 aniversario de Monopoly (2005), iniciándose el 20 de junio de dicho año con una duración de un mes. La experiencia promete repetirse gracias al éxito cosechado en la primera partida.

2.4 Conclusiones

Los juegos (formalizados por la teoría de juegos) aplicados a la IA permiten un rápido y óptimo avance en los algoritmos para la resolución de problemas del mundo real y de los juegos clásicos (ajedrez, damas, go,

¹² www.monopolylive.co.uk

póker...), del mismo modo que permite un rápido avance en la industria de los videojuegos que paulatinamente, y cada vez con más acierto, incluyen motores de IA.

Los juegos de 2 agentes han sido y son investigados por muchos de los investigadores de IA, y ya se conocen algoritmos competitivos unos derivados del *minimax*, y otros no.

Sin embargo, los juegos de n-agentes no han sido tan estudiados a lo largo de la historia, ya que su resolución presenta nuevos retos, al tener que contar con las posibles colaboraciones entre los agentes. Los algoritmos que se han expuesto no cuentan con la posibilidad de la colaboración entre los agentes, por lo que no será aplicable a una gran cantidad de juegos (como por ejemplo Risk y Monopoly).

El Monopoly es un juego muy interesante para el estudio de los algoritmos de distinta índole ya que es un juego de azar y de información incompleta en el que se dan lugar complejas cuestiones del mundo de la economía como las subastas y las negociaciones, y cuya implementación es, asimismo, un difícil reto.

Capítulo 3

Objetivos

3 Objetivos

El objetivo principal de este proyecto es diseñar, implementar y probar clientes de IA en el juego del Monopoly. Este diseño se deberá realizar siguiendo los principios del paradigma de representación de la información orientada a objetos.

- Este proyecto se ha desarrollado teniendo en cuenta las normas oficiales del monopoly, por lo que también se diseñarán e implementarán los procesos de subastas y negociaciones, que no se observan en otras implementaciones del juego (ver sección 2.3.5, página 29).
- El diseño se realizará en UML para que sea más fácilmente legible. Esta elección está fundamentada en la observación de que este modelo de diseño basado en objetos es muy cercano al modelo basado en marcos, con más sentido en la Ingeniería del Conocimiento. Sin embargo, puesto que se pretende implementar un sistema en C++ congruente con el diseño, UML resultará definitivamente más apropiado.
- Además, el juego se implementará usando C++ y las librerías STL (Standard Template Library) de dicho lenguaje. No se implementará un entorno gráfico, pero se utilizará la librería *GNU Readline* para la creación de la terminal del juego.
- Además, se probarán dos sencillos clientes de IA basados en reglas que jueguen utilizando este diseño. Éste agente no resolverá los problemas de negociación entre jugadores, dejándose como un nuevo campo de investigación para este proyecto, pero si resolverá el resto de cuestiones propias del juego, como la tima de decisiones para la compra, venta, hipotecas y deshipotecas, o un modelo de subastas.

Capítulo 4

Desarrollo

4 Desarrollo

A lo largo de este capítulo del proyecto se describirá de una forma completa como se ha resuelto el problema propuesto.

Se incluyen en este punto el análisis y diseño realizado representado gráficamente en UML, la solución propuesta para crear un cliente de IA basado en reglas y un manual de usuario en el que se profundiza en la aplicación desarrollada explicando qué ofrece y cómo se utiliza.

4.1 Análisis de requisitos

Street Master's debe cubrir todas las necesidades que se requieren para completar una partida del popular Monopoly. Este juego se caracteriza por permitir una gran variedad de acciones y funcionalidades a los jugadores que van desde la simple tirada de los dados hasta procesos más complejos como la construcción de las casas y hoteles en las calles o las negociaciones entre los jugadores.

Esta versión del juego se ha basado en las reglas oficiales del Monopoly descritas en el Estado de la Cuestión¹³ de éste mismo proyecto por lo que, además de las reglas más populares del juego, se han incluido otras funcionalidades menos conocidas como las subastas de propiedades.

Además de las funcionalidades propias del juego, es necesario que el jugador pueda cargar y guardar partidas y obtener toda la información sobre el juego (desde ver el tablero hasta ver la información de un jugador).

A continuación se muestra el diagrama de casos de uso que representa los requisitos potenciales de la aplicación y se realiza una descripción completa de cada uno de ellos.

¹³ 2.3.1. Reglas del juego, página 21.

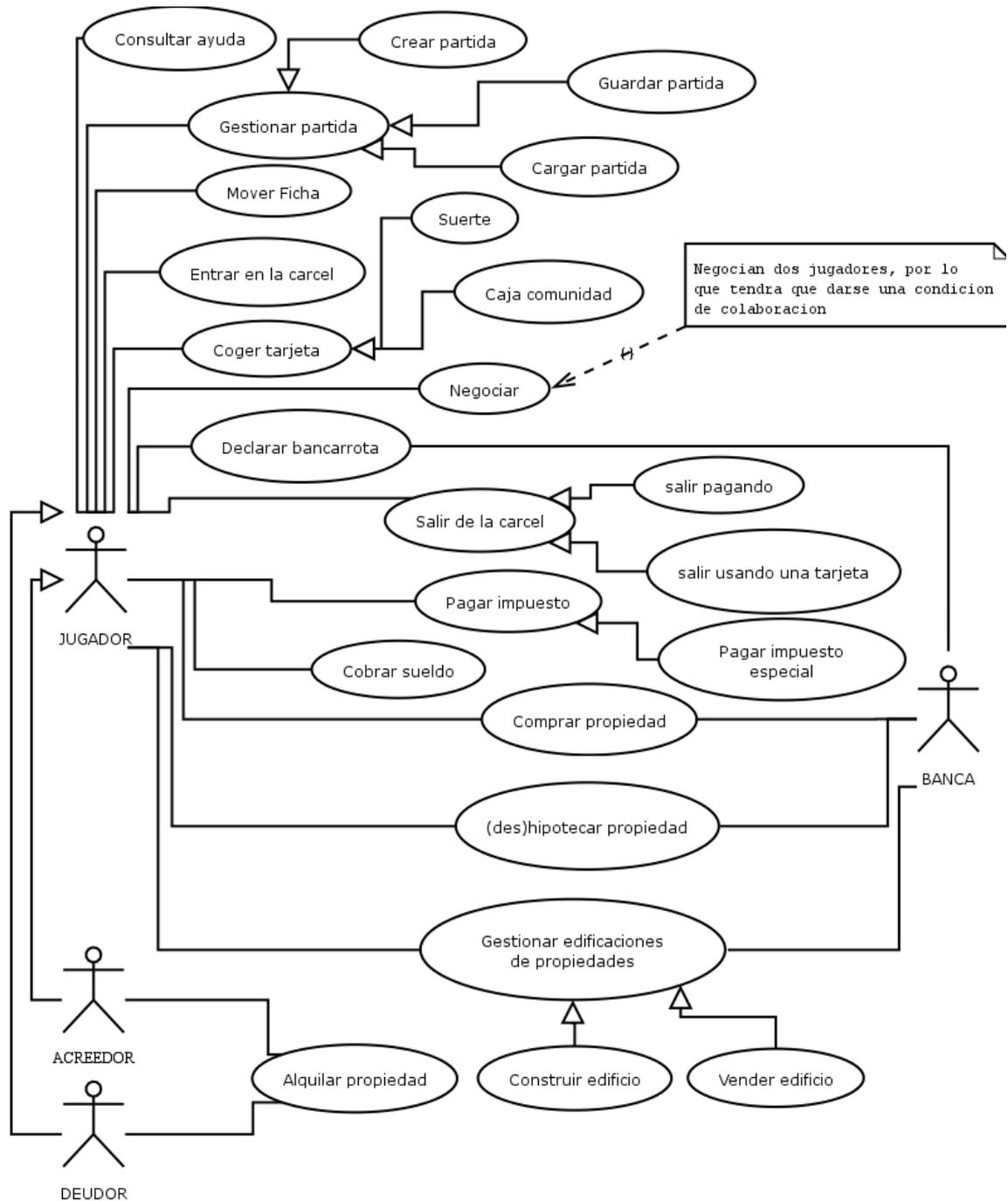


Diagrama 1: Diagrama de casos de uso de Street Master's

En el juego se identifican dos roles que interactúan con el sistema: la *banca*, que interviene en ciertas transacciones económicas con los jugadores y reparte la riqueza entre ellos; y los propios *jugadores*, que son los actores principales del juego, ya que es sobre ellos sobre quienes recaerán la mayor parte de las acciones que se muestran en el diagrama adjunto.

En algunos casos es necesario que varios jugadores colaboren entre sí para cumplir con una funcionalidad completa. Por ejemplo, para que se pueda dar un proceso de negociación o alquiler de una propiedad es necesario que intervengan dos jugadores, ya que de otro modo resultaría imposible. También se pueden dar condiciones de colaboración entre los jugadores y la banca.

A continuación se describen cada uno de los casos de uso, explicando qué funcionalidad cubre cada uno de ellos.

4.1.1 Gestionar partida

El caso de uso general *gestionar partida* se descompone en otros más concretos que se describen a continuación: *crear partida*, *guardar partida* y *cargar partida*.

Crear partida
<p>Descripción:</p> <p>El sistema creará una partida nueva a partir de los datos que le proporcione el usuario y recoja de los ficheros donde se almacena información sobre el tablero de juego y las tarjetas de 'suerte' y 'caja de comunidad'.</p>
<p>Actores:</p> <p>- Jugador</p>
<p>Precondiciones:</p> <p>- No debe existir ninguna partida abierta en el sistema</p>
<p>Escenario básico:</p> <ol style="list-style-type: none"> 1. Cargar la configuración por defecto de Street Master's 2. Cargar las tarjetas de suerte desde el fichero que se indica en la configuración del juego. 3. Cargar las tarjetas de caja de comunidad desde el fichero que se indican en la configuración del juego. 4. Cargar las casillas desde el fichero que marca la configuración como origen de las mismas. 5. Se indicarán el número de jugadores. 6. Se introducen los datos de los jugadores para crearlos en la partida. Se ofrecerá la posibilidad de que el jugador sea controlado por el ordenador.
<p>Escenario alternativo:</p> <ol style="list-style-type: none"> 2. El jugador puede elegir una configuración alternativa introduciendo sus preferencias. 5. Si se produce algún error en la carga de las casillas, se le deberá ofrecer la posibilidad al jugador de cancelar la carga de las mismas o continuar eliminando las erróneas.

Poscondiciones:

- Tablero creado
- Casillas creadas e insertadas en el tablero.
- Jugadores creados e incluidos en el tablero.
- Tarjetas de suerte y caja de comunidad creadas y asignadas a su mazo correspondiente.

Guardar partida**Descripción:**

Durante el desarrollo de una partida de Street Master's cualquier jugador debe poder guardar el estado actual de la partida en un fichero para una futura carga.

Actores:

- Jugador

Precondiciones:

- Debe existir una partida abierta en el sistema

Escenario básico:

1. Introducir el nombre con el que se guardará de la partida.
2. Guardar los datos generales de la partida.
3. Guardar los datos de los jugadores (y los clientes de IA)
4. Guardar el estado de las propiedades (hipotecadas, número de casas y hoteles, ...)

Escenario alternativo:

No aplicable

Poscondiciones:

- Partida guardada.

Cargar partida**Descripción:**

El usuario debe poder cargar una partida previamente salvada. De esta forma el juego se restaurará en el mismo estado que quedó en el momento de ser guardada.

Actores:

- Jugador

Precondiciones:

No aplicable.

Escenario básico:

1. Introducir el nombre del fichero guardado.
2. Cargar los datos generales de la partida.
3. Cargar los datos de los jugadores (y los clientes de IA)
4. Cargar el estado de las propiedades (hipotecadas, número de casas y hoteles, ...)

Escenario alternativo:

1. Si no existe el fichero introducido se devolverá un error.

Poscondiciones:

- Partida abierta.

4.1.2 Mover ficha

Mover ficha
Descripción: El jugador lanza los dados y el juego moverá la ficha por el tablero hasta llegar a la casilla destino.
Actores: - Jugador
Precondiciones: <ul style="list-style-type: none"> - Debe existir una partida abierta. - Debe mover ficha el jugador que posee el turno.
Escenario básico: <ol style="list-style-type: none"> 1. El jugador lanza los dados. 2. Se mueve la ficha desde la casilla origen hasta la casilla destino. 3. Ejecutar la acción correspondiente a la casilla. 4. Cambiar el turno al siguiente jugador.
Escenario alternativo: <ol style="list-style-type: none"> 2. Si el jugador hace una tirada doble por tercera vez, envía al jugador a la cárcel. 2. Si el jugador está en la cárcel debe hacer una tirada doble para salir o que sea su tercer turno encarcelado. 2. Si el jugador pasa por la casilla de “salida”, deberá cobrar el sueldo. 3. Si el jugador no puede hacer frente a una deuda se declara en bancarrota. 4. Si es una tirada doble no se cambia de turno.
Poscondiciones: <ul style="list-style-type: none"> - Jugador colocado en la nueva casilla. - Establecido nuevo turno.

4.1.3 Entrar en la cárcel

Entrar en la cárcel
Descripción: El jugador se debe desplazar desde su posición actual hasta la casilla de cárcel del tablero.

Actores:
- Jugador
Precondiciones:
- Debe existir una partida abierta. - El jugador se encuentra en la casilla “Vaya a la cárcel” o haber realizado tres tiradas dobles consecutivas.
Escenario básico:
1. Mover al jugador a la casilla de tipo “cárcel”. 2. Indicar al jugador que debe permanecer encarcelado durante los turnos establecidos.
Escenario alternativo:
No aplicable.
Poscondiciones:
- Jugador situado en la casilla de cárcel. - El jugador debe tener el estado <i>encarcelado</i> .

4.1.4 Coger tarjeta

El juego dispone de dos mazos de tarjetas de las que se obtendrán las tarjetas durante una partida en función del tipo de casilla de tarjetas en la que se encuentre el jugador: tarjetas de *suerte* y *caja de comunidad*. El proceso y los requisitos son los mismos en ambos casos, y tan sólo difiere en el mazo del que se recogerán las tarjetas.

La información sobre las acciones que se pueden dar en una tarjeta se explicará con más detalle en el siguiente punto de esta memoria¹⁴.

Coger tarjeta
Descripción:
El jugador que ha caído sobre una casilla de tarjetas debe recoger la tarjeta que se encuentre en la posición más elevada del mazo y realizar la acción que se indique en ella.
Actores:
- Jugador
Precondiciones:
- Debe existir una partida abierta. - El jugador se encuentra en la casilla de tarjetas (“suerte” o “caja de comunidad”).

¹⁴ Ver sección 4.2 Ontología

Escenario básico:

1. El jugador recoge la tarjeta que está en la cima del mazo correspondiente.
2. Se ejecuta la acción que ordena la tarjeta.

Escenario alternativo:

2. En el caso de que se pueda elegir entre varias acciones, se debe seleccionar una antes de ejecutarse.

Poscondiciones:

- Acción de la tarjeta ejecutada.

4.1.5 Negociar

El proceso de negociación implica una colaboración entre dos jugadores, en el que uno de ellos propone una oferta y el otro deberá rechazarla o aceptarla según considere oportuno.

Esta funcionalidad sólo estará disponible si no se juega contra jugadores manejados por el ordenador.

Negociar
Descripción: <p>En cualquier momento cualquier jugador puede proponer un negocio a otro. La oferta formulada puede incluir dinero, propiedades y tarjetas para salir de la cárcel si se posee alguna.</p>
Actores: <ul style="list-style-type: none"> - Jugador (dos jugadores colaborando)
Precondiciones: <ul style="list-style-type: none"> - Debe existir una partida abierta. - No deben existir clientes IA.
Escenario básico: <ol style="list-style-type: none"> 1. El jugador que inicia el proceso selecciona quien va a ser el receptor de su oferta. 2. El emisor de la oferta indicará el dinero, las propiedades y las tarjetas para salir de la cárcel (si dispone de ellas) por este orden. 3. A continuación debe indicar lo que pide a cambio (siguiendo el mismo proceso). 4. Por último el jugador receptor deberá decidir si desea aceptar o rechazar la oferta. 5. Reasignar las propiedades que intervienen en la negociación.
Escenario alternativo: <ol style="list-style-type: none"> 5. Si no se acepta la oferta, no se realiza este paso.
Poscondiciones: <ul style="list-style-type: none"> - Jugadores actualizados (dinero y propiedades)

4.1.6 Declarar bancarrota

Cuando un jugador no tiene dinero para cubrir sus deudas (situación de quiebra) se debe declarar en bancarrota. Hasta que esto no ocurra el juego no podrá continuar.

Existen tres posibles consecuencias al aplicar esta funcionalidad en función de quien sea el acreedor.

Si el acreedor es un único jugador las propiedades del jugador arruinado se deberán asignar directamente al acreedor. Si son varios los acreedores se deberán subastar las propiedades entre todos ellos.

Cuando el acreedor es la banca, se debe realizar una subasta de todas las propiedades entre todos los jugadores.

Declarar bancarrota
<p>Descripción:</p> <p>Cuando un jugador de Street Master's no tiene suficiente dinero como para seguir en el juego, debe declararse en bancarrota.</p>
<p>Actores:</p> <ul style="list-style-type: none"> - Jugador - Banca
<p>Precondiciones:</p> <ul style="list-style-type: none"> - Debe existir una partida abierta.
<p>Escenario básico:</p> <ol style="list-style-type: none"> 1. Un jugador se declara en bancarrota 2. Se resuelve el reparto de las propiedades: <ol style="list-style-type: none"> 2a. Si el acreedor es un jugador, se asignan las propiedades del deudor al acreedor. 2b. Si son varios los acreedores, se subastarán las propiedades entre ellos. 2c. Si el acreedor es la banca, se subastan las propiedades entre todos los jugadores 3. Se continúa la partida
<p>Escenario alternativo:</p> <ol style="list-style-type: none"> 3. Si sólo queda un jugador en la partida, se considera concluida, y dicho jugador será el ganador
<p>Poscondiciones:</p> <ul style="list-style-type: none"> - Jugador en quiebra eliminado.

4.1.7 Salir de la cárcel

Existen varias posibilidades para salir de la cárcel, por lo que se mostrará la explicación de todas ellas por separado: *salir de la cárcel por cumplir el máximo de turnos*, *salir de la cárcel pagando la multa* y *salir de la cárcel utilizando una tarjeta de suerte o caja de comunidad*.

Salir de la cárcel
<p>Descripción:</p> <p>Un jugador sólo podrá permanecer encarcelado durante un número determinado de turno, una vez que se cumplan dichos turnos, se debe liberar al jugador de la cárcel.</p>
<p>Actores:</p> <ul style="list-style-type: none"> - Jugador
<p>Precondiciones:</p> <ul style="list-style-type: none"> - Debe existir una partida abierta. - El jugador debe estar encarcelado (situado en la casilla de cárcel y con estado <i>encarcelado</i>) - El jugador debe cumplir en el turno actual el número máximo de turnos que puede estar encarcelado. - Debe ser el turno de juego del jugador interesado.
<p>Escenario básico:</p> <ol style="list-style-type: none"> 1. Se cambia el estado del jugador a <i>desencarcelado</i>.
<p>Escenario alternativo:</p> <p>No aplicable.</p>
<p>Poscondiciones:</p> <ul style="list-style-type: none"> - Jugador desencarcelado.

Salir pagando
<p>Descripción:</p> <p>Antes de cumplir el número máximo de turnos que puede estar un jugador en la cárcel, se le debe ofrecer la posibilidad de pagar una multa para salir con anterioridad. Si el jugador acepta pagar para salir de la cárcel, se debe desencarcelar al mismo.</p>
<p>Actores:</p> <ul style="list-style-type: none"> - Jugador

Precondiciones:

- Debe existir una partida abierta.
- El jugador debe estar encarcelado (situado en la casilla de cárcel y con estado *encarcelado*)
- El jugador debe tener dinero suficiente para pagar la multa.
- Debe ser el turno del jugador interesado.

Escenario básico:

1. Se descuenta el dinero indicado en la multa al jugador encarcelado.
2. Se cambia el estado del jugador a *desencarcelado*.

Escenario alternativo:

No aplicable.

Poscondiciones:

- Jugador desencarcelado.
- Dinero del jugador actualizado

Salir usando una tarjeta**Descripción:**

Si un jugador se encuentra encarcelado y dispone de una tarjeta que permite salir de la cárcel, dicho jugador puede usarla para salir de la cárcel antes de cumplir el número máximo de turnos que puede permanecer encarcelado..

Actores:

- Jugador

Precondiciones:

- Debe existir una partida abierta.
- El jugador debe estar encarcelado (situado en la casilla de cárcel y con estado *encarcelado*)
- El jugador debe poseer una tarjeta para salir de la cárcel.
- Debe ser el turno del jugador encarcelado.

Escenario básico:

- 1.El jugador devuelve la tarjeta al mazo correspondiente.
2. Se cambia el estado del jugador a *desencarcelado*.

Escenario alternativo:

No aplicable.

Poscondiciones:

- Jugador desencarcelado.
- Tarjeta colocada en el mazo correspondiente.

4.1.8 Pagar impuesto

Además de la funcionalidad de pagar impuestos, existe una especialización para aquellos impuestos especiales que se encuentran en el juego en los que se puede elegir entre pagar una cantidad fija o calcular un porcentaje sobre el capital.

Pagar impuesto
<p>Descripción:</p> <p>Al caer en una casilla de impuestos, se debe descontar al jugador la cantidad indicada..</p>
<p>Actores:</p> <ul style="list-style-type: none"> - Jugador
<p>Precondiciones:</p> <ul style="list-style-type: none"> - Debe existir una partida abierta. - El jugador debe estar situado en una casilla de impuestos. - Debe ser el turno del jugador involucrado.
<p>Escenario básico:</p> <ol style="list-style-type: none"> 1. Descontar la cantidad necesaria al jugador.
<p>Escenario alternativo:</p> <ol style="list-style-type: none"> 1. Si el jugador no ha podido pagar la deuda, porque no dispone de dinero necesario, se le debe dejar hasta finalizar el turno para que consiga dinero o se declare en bancarrota.
<p>Poscondiciones:</p> <ul style="list-style-type: none"> - Dinero del jugador actualizado

Pagar impuesto especial
<p>Descripción:</p> <p>Al caer en una casilla de impuestos especiales se debe descontar al jugador la cantidad indicada o calculada a partir del capital del propio jugador y el porcentaje indicado.</p>
<p>Actores:</p> <ul style="list-style-type: none"> - Jugador
<p>Precondiciones:</p> <ul style="list-style-type: none"> - Debe existir una partida abierta. - El jugador debe estar situado en una casilla de impuestos especiales. - Debe ser el turno del jugador involucrado.
<p>Escenario básico:</p> <ol style="list-style-type: none"> 1. El jugador elegirá la opción que desee: pagar el porcentaje o la cantidad fija. 2. Descontar la cantidad seleccionada al jugador.

Escenario alternativo:

2. Si el jugador no ha podido pagar la deuda, porque no dispone de dinero necesario, se le debe dejar hasta finalizar el turno para que consiga dinero o se declare en bancarrota.

Poscondiciones:

- Dinero del jugador actualizado

4.1.9 Cobrar sueldo

Cobrar sueldo
Descripción: El jugador debe cobrar un salario fijo al pasar por la casilla de salida.
Actores: - Jugador
Precondiciones: - Debe existir una partida abierta. - El jugador debe moverse desde una casilla previa a la salida hasta otra posterior. - Debe ser el turno del jugador involucrado.
Escenario básico: 1. Pagar al jugador el dinero indicado.
Escenario alternativo: No aplicable
Poscondiciones: - Dinero del jugador actualizado

4.1.10 Comprar propiedad

Al comprar una propiedad se realiza una transacción de activos entre la banca y un jugador, ya que las propiedades inicialmente pertenecen a la banca.

Al comprar una calle se debe tener en cuenta que si se completa un monopolio, se debe permitir construir sobre él.

Comprar propiedad
Descripción: Un jugador puede comprar una propiedad (calle, servicio o estación) al caer sobre la casilla que le representa y ésta no pertenece a ningún otro jugador.

<p>Actores:</p> <ul style="list-style-type: none"> - Jugador - Banca
<p>Precondiciones:</p> <ul style="list-style-type: none"> - Debe existir una partida abierta. - El jugador se encuentra en una propiedad libre, sin dueño. - El jugador dispone de dinero suficiente para comprar la propiedad. - Debe ser el turno del jugador involucrado.
<p>Escenario básico:</p> <ol style="list-style-type: none"> 1. El jugador decide comprar la propiedad. 2. Se descuenta el precio de la propiedad al jugador. 3. Se añade la propiedad a las posesiones del jugador comprador.
<p>Escenario alternativo:</p> <ol style="list-style-type: none"> 1. Si el jugador no quiere comprar la propiedad y está activa la opción de la subastas, se subastará la propiedad entre todos los jugadores. La comprará aquel que puje más alto. 4. Si se compra una calle, se debe comprobar si se completa un monopolio. Si se ha completado un monopolio, se otorgará el permiso de construcción a todas las calles miembro del mismo.
<p>Poscondiciones:</p> <ul style="list-style-type: none"> - Propiedad asignada a su nuevo dueño - Permiso de construcción actualizado en las calles. - Dinero del jugador actualizado

4.1.11 Hipotecar propiedad

Al igual que se ha explicado en el caso de las compras de propiedades, al hipotecar y deshipotecar propiedades hay que tener en cuenta el caso especial de las calles para otorgarles y denegarles el permiso de construcción en todo el monopolio.

Hipotecar propiedad
<p>Descripción:</p> <p>El jugador hipoteca una propiedad a cambio del dinero estipulado en cada propiedad. Al hipotecar la propiedad se renuncia a los derechos de cobrar por tenerla.</p>
<p>Actores:</p> <ul style="list-style-type: none"> - Jugador - Banca

Precondiciones:

- Debe existir una partida abierta.
- El jugador debe ser el propietario de la propiedad que desea hipotecar
- La propiedad no puede estar hipotecada.

Escenario básico:

1. El jugador cobra la cantidad estipulada por hipotecar su propiedad.
2. La propiedad pasará al estado hipotecado.

Escenario alternativo:

3. Si se hipoteca una calle, se deben actualizar los permisos de construcción.

Poscondiciones:

- La propiedad pasa a tener el estado hipotecado.
- Permisos de construcción actualizados
- Dinero del jugador actualizado

4.1.12 Deshipotecar propiedad

Al deshipotecar una propiedad hay que tener en cuenta las mismas consideraciones que se han explicado en el caso de las hipotecas.

Deshipotecar propiedad
Descripción: Un jugador devuelve el dinero que cobró por hipotecar la propiedad más un 20% de intereses y vuelve a tener todos los derechos sobre la propiedad.
Actores: <ul style="list-style-type: none"> - Jugador - Banca
Precondiciones: <ul style="list-style-type: none"> - Debe existir una partida abierta. - El jugador debe ser el propietario de la propiedad que desea deshipotecar - La propiedad debe estar hipotecada. - El jugador debe tener el dinero necesario para deshipotecar la propiedad.
Escenario básico: <ol style="list-style-type: none"> 1. El jugador paga la cantidad estipulada por deshipotecar su propiedad. 2. La propiedad deja de estar hipotecada.
Escenario alternativo: <ol style="list-style-type: none"> 3. Si se deshipoteca una calle, se deben actualizar los permisos de construcción.

Poscondiciones:

- La propiedad deja de tener el estado hipotecado.
- Permisos de construcción actualizados.
- Dinero del jugador actualizado

4.1.13 Gestionar edificaciones de las propiedades

Un jugador tiene la potestad de edificar sus calles, del mismo modo que puede vender sus casas y hoteles si lo considera oportuno. Es importante indicar que esto será posible sólo si la banca tiene suficientes edificios para construir. Al vender un hotel, también se debe comprobar que existan cuatro casas para sustituirle.

A continuación se describen los casos de uso *construir edificio* y *vender edificio*.

Construir edificio
<p>Descripción:</p> <p>Un jugador construirá edificios sobre las calles que tienen permiso para ser construidas. Este permiso se consigue completando un monopolio.</p>
<p>Actores:</p> <ul style="list-style-type: none"> - Jugador - Banca
<p>Precondiciones:</p> <ul style="list-style-type: none"> - Debe existir una partida abierta. - La propiedad que se desea construir debe ser una calle. - El jugador debe tener dinero suficiente. - El jugador debe ser el propietario de la calle que desea construir - La calle debe tener permiso de construcción. - La calle debe tener espacio para construir. - La banca debe tener edificios suficientes (tanto casas como hoteles)
<p>Escenario básico:</p> <ol style="list-style-type: none"> 1. Se comprueba que se cumplen las reglas de construcción. 2. El jugador paga el precio de construcción del edificio en una calle concreta. 3. Se construye una casa sobre la calle indicada.
<p>Escenario alternativo:</p> <ol style="list-style-type: none"> 1. Si no se cumplen las reglas, no se construye. 3. Si existen cuatro casas construidas, la nueva edificación será un hotel y las cuatro casas se devolverán a la banca.

Poscondiciones:

- Nueva edificación sobre la calle indicada.
- Edificios de la banca actualizados.
- Dinero del jugador actualizado

Vender edificio**Descripción:**

Un jugador recibe dinero por vender los edificios que ha construido previamente en sus calles. El dinero que se recibe de vender un edificio es la mitad de lo que costó comprarla.

Actores:

- Jugador
- Banca

Precondiciones:

- Debe existir una partida abierta.
- La calle de la que se desean vender los edificios, debe tenerlos construidos.
- El jugador debe ser el propietario de la calle que desea desedificar.

Escenario básico:

1. Se comprueba que se cumplen las reglas de construcción.
2. El jugador cobra el precio de venta del edificio en una calle concreta.
3. Se elimina una casa / hotel de la calle indicada.
4. Se devuelve el edificio a la banca.

Escenario alternativo:

0. Si se desea vender un hotel, se debe comprobar que la banca tenga cuatro casas disponibles
1. Si no se cumplen las reglas, no se realiza la operación de vender.

Poscondiciones:

- Edificación eliminada en la calle indicada.
- Edificios de la banca actualizados.
- Dinero del jugador actualizado

4.1.14 Alquilar propiedad

Se interpreta alquilar como el hecho de que un jugador caiga en una casilla que representa una propiedad de otro jugador por lo que deberá pagarle el precio estipulado. Para cumplir esta funcionalidad se necesitan dos jugadores que colaboren entre sí.

En el diagrama de casos de uso se ha representado esta colaboración con dos jugadores específicos (acreedor y deudor), en el que uno pagará y

el otro cobrará. Éste es un claro ejemplo de *suma nula*, que se explicó en el segundo capítulo de este proyecto (página 10).

El alquiler de las propiedades es similar para todos los tipos de propiedades, sólo varía la forma de calcular el precio de alquiler.

Alquilar propiedad
<p>Descripción:</p> <p>Un jugador pagará a otro cuando caiga sobre una propiedad que le pertenezca a este último. La cantidad a pagar irá en función del estado de la partida (número de estaciones que posea, casas construidas, ...).</p>
<p>Actores:</p> <ul style="list-style-type: none"> - Acreedor (Jugador) - Deudor (Jugador)
<p>Precondiciones:</p> <ul style="list-style-type: none"> - Debe existir una partida abierta. - El jugador deudor se encuentra en una propiedad que no está libre y no le pertenece. - El turno del juego es del jugador deudor. - La propiedad pertenece al jugador acreedor.
<p>Escenario básico:</p> <ol style="list-style-type: none"> 1. Se calcula el precio de alquiler de la propiedad. 2. El jugador deudor paga la cantidad obtenida al acreedor.
<p>Escenario alternativo:</p> <ol style="list-style-type: none"> 2. Si el jugador deudor no tiene dinero, deberá conseguir el dinero necesario antes de que finalice su turno. Si no lo consigue deberá declararse en bancarrota.
<p>Poscondiciones:</p> <ul style="list-style-type: none"> - Dinero de ambos jugadores actualizado.

4.1.15 Consultar ayuda

Consultar ayuda
<p>Descripción:</p> <p>El usuario seleccionará el tema de ayuda que desea consultar.</p>
<p>Actores:</p> <ul style="list-style-type: none"> - Jugador
<p>Precondiciones:</p> <p>No aplicable</p>

<p>Escenario básico:</p> <ol style="list-style-type: none"> 1. Se selecciona el tema de ayuda. 2. Se muestra la ayuda.
<p>Escenario alternativo:</p> <p>No aplicable</p>
<p>Poscondiciones:</p> <p>No aplicable.</p>

4.1.16 Conclusiones sobre los casos de uso

Con los requisitos aquí descritos se cubren todas las necesidades de un juego del estilo del Monopoly. Como se puede ver, la cantidad de acciones y opciones que se deben tener en cuenta es importante, lo cual se debe a la compleja normativa que se ha descrito en esta misma memoria (capítulo 2.3.1 página 21) y a la gran cantidad de objetos que intervienen durante una partida, como se explicará en la ontología del juego (siguiente apartado de éste capítulo).

4.2 Ontología

El juego se ha diseñado cumpliendo todos los requisitos definidos anteriormente, de forma que el jugador pueda disputar partidas usando un aplicación implementada siguiendo las directrices de este diseño.

Como se comprobará en las siguientes páginas, la envergadura del diseño es de un tamaño considerable, por lo que resulta muy complicado explicarlo todo junto. Por ello, y con el objetivo de mejorar la precisión en las cuestiones más importantes, se divide la explicación del diseño en partes más pequeñas:

- Diseño del tablero (jerarquía de casillas)
- El caso especial de las propiedades
- El jugador en las casillas de tarjetas
- Los propietarios de las propiedades
- El papel de la banca

- Las negociaciones entre jugadores
- El cliente de IA.

Se advierte que en cada caso se caracterizará cada elemento, únicamente, por su vista estática – atributos.

Antes de pasar a una explicación detallada, se mostrará el diseño completo para situar cada elemento dentro del contexto general del juego.

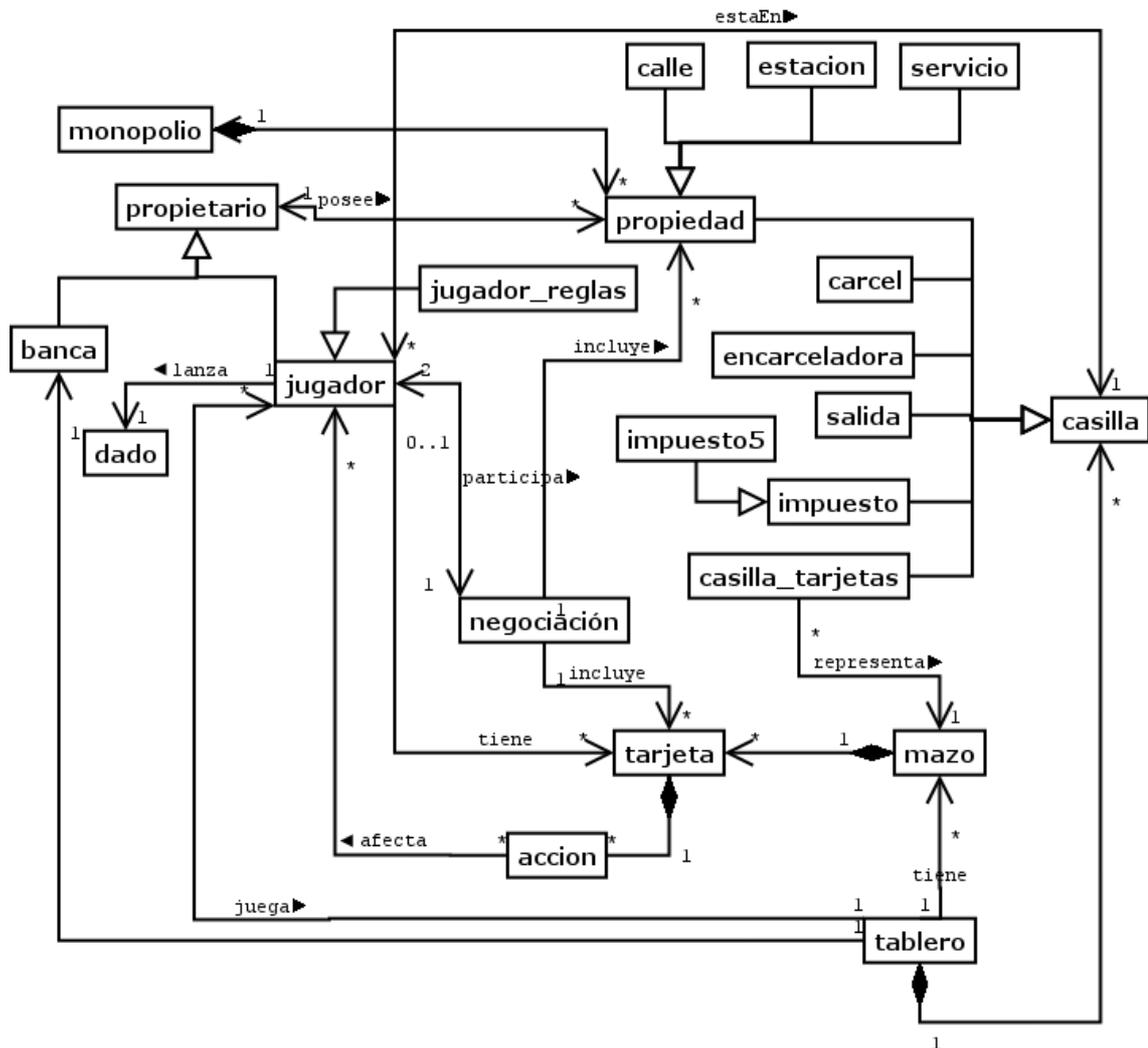


Diagrama 2: Diagrama de clases de Street Master's

La idea general del diseño que se muestra sobre estas líneas es representar el juego apoyándose en los principios del paradigma de la orientación a objetos: herencia, polimorfismo, encapsulación y abstracción.

Como se puede ver, la mayor parte del diseño se centra en representar las casillas del tablero, ya que se ha tomado la decisión de basar el funcionamiento del juego en base a dos objetos: el jugador y las casillas. De esta forma se consigue que mediante herencias y polimorfismos, se puedan completar todos requisitos que debe cumplir el juego de una forma tan eficiente como eficaz.

La clase *tablero* es el motor de la aplicación, ya que todas las operaciones que un usuario desee realizar sobre el juego deberán pasar por ella, que será la encargada de distribuir las acciones entre las clases necesarias. No obstante, no es su única misión, ya que además almacenará toda la información general sobre la partida, como se comentará más adelante.

El diseño mostrado incluye la posibilidad de realizar negociaciones entre varios jugadores, lo cual aporta una funcionalidad que no se cumple en algunos de los juegos de Monopoly analizados en el Estado de la cuestión¹⁵.

4.2.1 Diseño del tablero

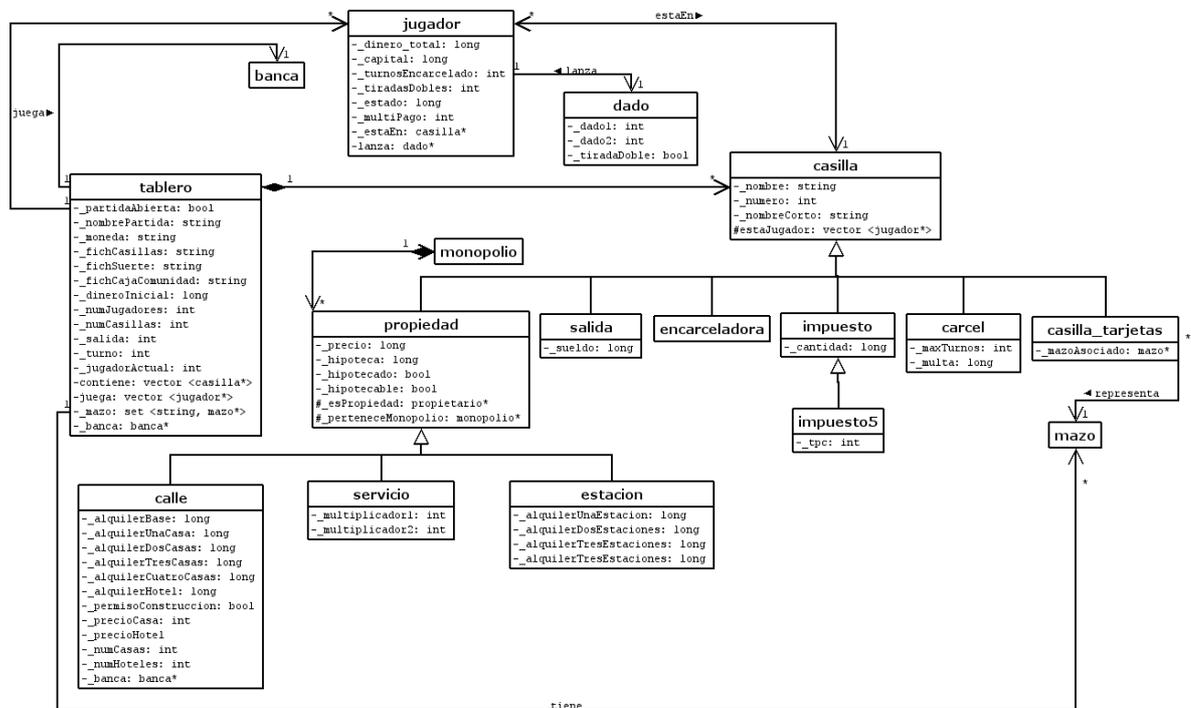


Diagrama 3: Diagrama de clases (diseño del tablero)

¹⁵ Ver sección 2.3.5 Juegos comerciales, página 29

La clase *tablero* se instancia una única vez ya que el juego sólo permite jugar una partida simultáneamente. Éste tablero almacenará todos los datos de interés de la partida: ficheros de origen de las tarjetas (*_fichSuerte* y *fichCajaComunidad*) y las casillas (*_fichCasillas*), el jugador que posee el turno de tirada (*_turno*) y el que tiene el control sobre el juego (*_jugadorActual*) y otros datos de interés de la configuración de la partida, como el dinero del que dispone cada jugador al comenzar.

Con los atributos descritos se puede llevar un control absoluto sobre el juego permitiendo que el turno avance sobre los distintos jugadores participantes y éstos puedan utilizar las funcionalidades descritas en el *análisis de requisitos*¹⁶.

El tablero de Street Master's está compuesto por un número indeterminado de casillas, ya que pueden ser cargados diferentes tipos de tablero de acuerdo unos patrones establecidos, tal y como se explicará en el *Manual de usuario*¹⁷, y una serie de jugadores que participarán en la partida.

Seguramente, la representación de las casillas es la parte más importante del diseño, ya que todo él gira en torno a los diversos tipos de casillas. Se han diseñado en el juego todas las diferentes casillas que se pueden encontrar en un tablero tradicional de Monopoly. Para ello, se ha creado una jerarquía de casillas en las que la clase base será la clase *casilla* y el resto de las casillas serán derivadas de ella, heredando aquellos atributos comunes (*nombre y número de casilla*) y valiéndose del principio de polimorfismo para desarrollar sus propias acciones. De esta manera se simplificará considerablemente la interacción entre jugadores y casillas. Las casillas identificadas en el tablero de Monopoly son las siguientes:

- Salida (clase *salida*): La casilla salida, como su propio nombre indica es la casilla en la que iniciarán el juego todos los jugadores. Además, se deberá pagar a los jugadores un salario al pasar por esta casilla. Dicho sueldo se almacena en el atributo *_sueldo*.

¹⁶Análisis de requisitos (4.1 de este documento)

¹⁷Manual de usuario (4.6 de la memoria)

- Propiedades (clase *propiedad*): Por la importancia que esta clase tiene en el desarrollo del juego, y teniendo en cuenta su complejidad, se dedicará un apartado especial para ella más adelante (sección 4.2.2).
- Impuestos (clase *impuesto*): Existen dos tipos de impuestos en el juego. El primero de ellos es el impuesto en el que se debe pagar una cantidad fija de dinero, representado en el diseño en la clase *impuesto*. Esta clase debe conocer la cantidad a pagar (atributo *cantidad*) por el jugador al caer sobre ella. El otro tipo de impuesto es el impuesto especial, clase *impuesto5*, que tradicionalmente ocupa la casilla número 5 en el tablero de Monopoly. Este impuesto además de almacenar la cantidad fija a pagar (atributo que se hereda de la clase *impuesto*), debe almacenar un tanto por ciento (*_tpc*) para que el jugador elija si desea pagar la cantidad fija o el porcentaje de su capital.
- Casilla que envía al jugador a la cárcel (clase *encarceladora*): Esta casilla se identifica en el tablero por un mensaje explícito, "Vaya a la cárcel". Esta clase no tiene ningún atributo especial. Al encarcelar a un jugador, se deberá llevar un control sobre el número de turnos encarcelados que lleva el jugador en dicha situación (atributo *_turnosEncarcelado* de la clase jugador).
- Cárcel (clase *carcel*): Esta clase tiene como objetivo mantener al jugador encarcelado hasta que se cumpla alguna de las condiciones para salir. Por ello debe disponer de la información necesaria para saber cuándo debe salir un jugador de sus dependencias. Estos datos son los atributos *_multa* y *_maxTurnos*, que representan las dos formas de salir de la cárcel más habituales: pagar por salir y salir de la cárcel por cumplir el máximo de turnos establecidos. La tercera forma descrita en los casos de uso (salir usando una tarjeta), se explicará más adelante cuando se expliquen las tarjetas de suerte y caja de comunidad.
- Casillas de tarjetas (clase *casilla_tarjetas*): Las casillas de tarjetas presentan una interesante discusión sobre su diseño, por lo que se

remite al lector al apartado 4.2.3 (página 62) para conocer el diseño de estas casillas.

Los jugadores se desplazarán por el tablero, según la cantidad que se obtenga al lanzar los dados (clase *dado*). Se debe tener en cuenta que si al lanzar los dados se obtiene una tirada doble, se debe incrementar el número de tiradas dobles del jugador (atributo *_tidadasDobles*), ya que si se obtiene una tirada doble, el turno no debe cambiar, y si se obtienen tres de forma consecutiva, el jugador deberá ser encarcelado.

Por último, es preciso enfatizar la importancia de la banca, representada en la clase del mismo nombre, que tendrá una gran importancia de aquí en adelante – en la compra de propiedades y en la construcción de edificios interviene de forma decisiva.

4.2.2 Propiedades

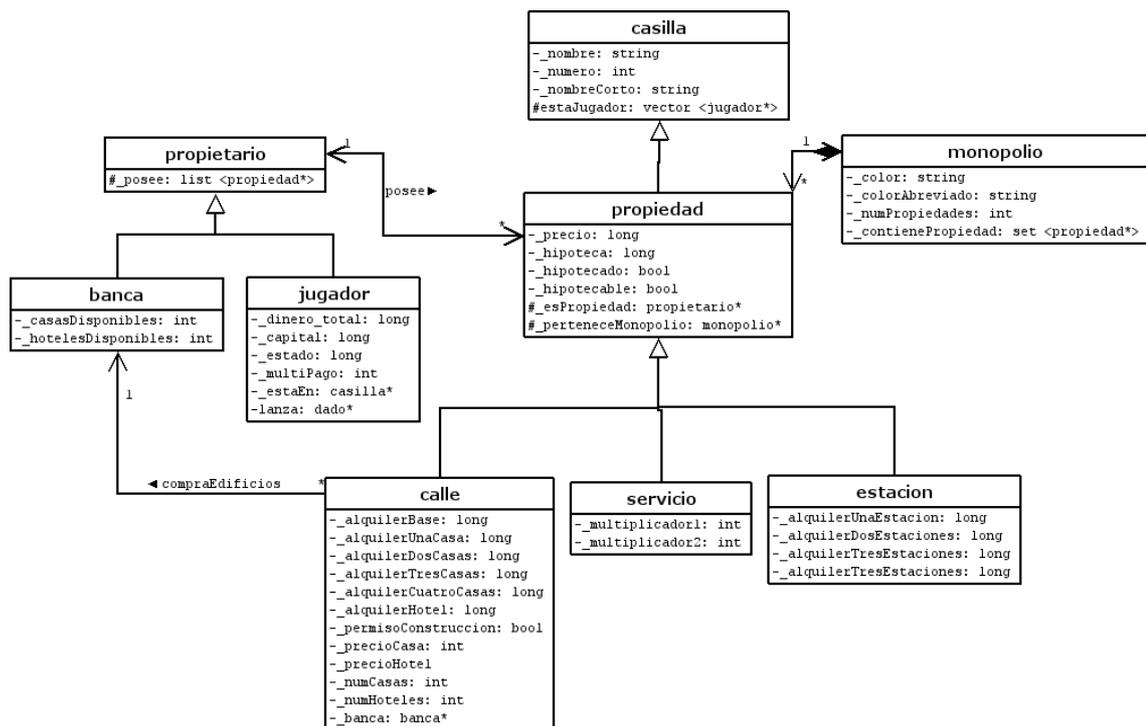


Diagrama 4: Diagrama de clases (diseño de las propiedades)

La clase derivada *propiedad* está entre las más complejas de todo el diseño, por lo que se ha decidido dedicarle una sección aparte.

Se entiende como propiedad toda aquella casilla que puede ser adquirida por un jugador con el objetivo de obtener beneficio de ella. Las propiedades deben pertenecer a un grupo de propiedades que en el juego de Street Master's se conocen como 'monopolio'. Por lo tanto, se ha creado una clase *monopolio* que agrupará a todas las propiedades del mismo grupo.

Esta clase es muy importante, ya que el comportamiento de las propiedades no es el mismo en función del número de propiedades del mismo grupo que se posean. Por ejemplo, una calle sólo es edificable si se tiene todo el grupo completo y el precio de alquiler de una estación depende del número de estaciones que se posean.

Las propiedades se han diseñado formando una nueva jerarquía donde la clase base es *propiedad*, y las derivadas son los distintos tipos de propiedades que existen a saber: *calle*, *servicio* y *estación*.

Todas las propiedades deben tener un propietario, que puede ser o bien la banca, o bien uno de los jugadores.

En función de quien sea el propietario, la propiedad tendrá un comportamiento u otro:

- Si el propietario es la banca, se interpretará la casilla como libre, y los jugadores podrán comprarla cuando caigan sobre ella por el precio estipulado (atributo *_precio* de la clase *propiedad*).
- Algo más compleja se convierte la situación cuando el propietario es un jugador. En este caso se abre un amplio abanico de posibilidades para gestionar las propiedades:
 - Las propiedades podrán ser hipotecadas a cambio de una cantidad de dinero (atributo *_hipoteca*), situándola en estado hipotecada.
 - La otra funcionalidad que aparece cuando se compra una propiedad es cobrar el alquiler a otros jugadores por caer en ella. La cantidad a cobrar se calculará de forma diferente en función del tipo de propiedad del que se trate. Si se trata de una calle, se obtendrá de comprobar el número de edificios que se han construido en dicha calle; las estaciones tendrán un

alquiler diferente en función de las propiedades de este tipo que posea el jugador acreedor; los servicios obtendrán el precio de multiplicar el número sacado en los dados por un multiplicador, que se obtendrá de comprobar el número de propiedades de este grupo que posee el jugador.

Además, la clase derivada *calle* ofrece la posibilidad de edificar y eliminar edificios sobre sus instancias. Al construir y eliminar un edificio, se deben cumplir todas las normas sobre edificación que se explicaron en el segundo capítulo, pero además, la banca debe tener edificios suficientes para colocarlos en la calle indicada (atributos *_casasDisponibles* y *_hotelesDisponibles* de la clase *banca*). Al construir un edificio se modificarán los atributos *_numCasas* y *_numHoteles* de la clase *calle*.

4.2.3 El jugador en las casillas de tarjetas

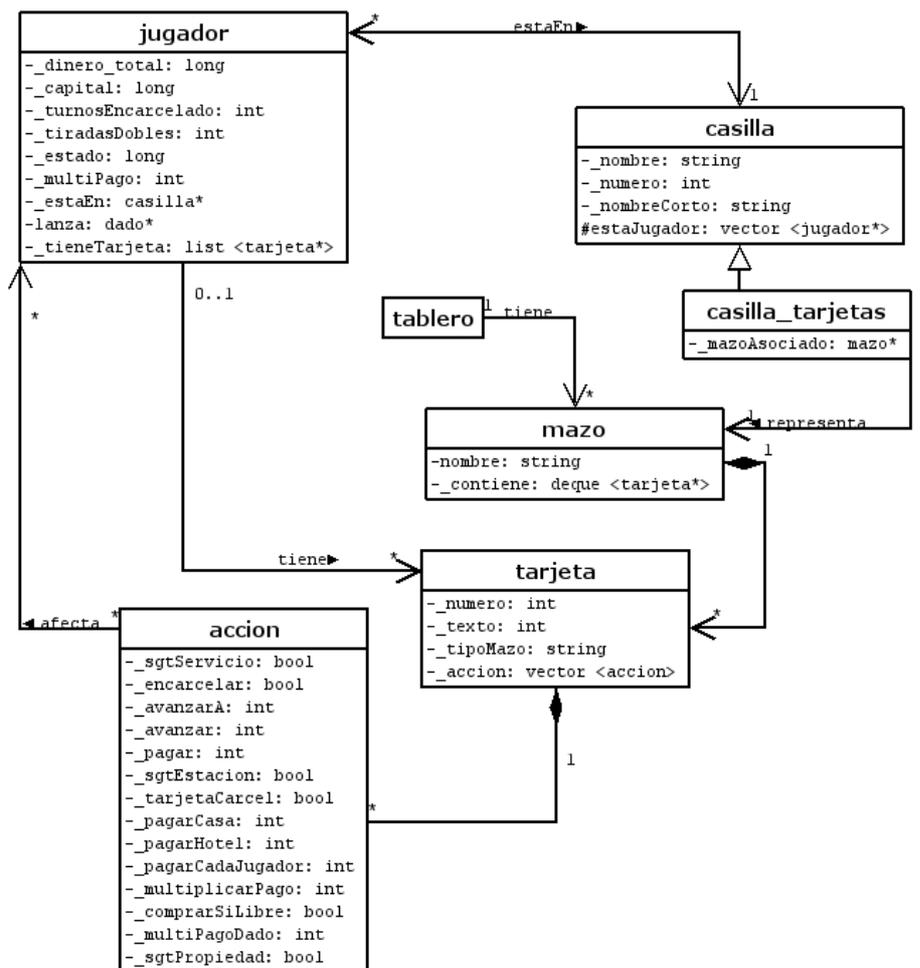


Diagrama 5: Diagrama de clases (diseño de las tarjetas)

Todas las casillas que implican coger una tarjeta de un mazo, tienen un mazo asociado del que recoger dichas tarjetas. Esta situación es algo diferente a la que se da en el resto de las casillas, ya que en este caso la acción que debe realizar el jugador vendrá dada por la ejecución de la acción que indique la tarjeta que se encuentre en la cima del mazo.

El modelado de las tarjetas es uno de los puntos más interesantes de todo el diseño. Obviamente, todas las tarjetas estarán asociadas a un único mazo y estarán identificadas por un número (atributo *numero*) y un texto explicativo de la misma (*_texto*).

Sin embargo, la pregunta que se plantea es cómo representar computacionalmente las acciones de cualquier tarjeta, ya sean una secuencia de acciones o un conjunto de alternativas de ellas.

Para resolver esta situación se ha creado una nueva clase: la clase *accion*. De esta forma una tarjeta podrá estar asociada a una o varias acciones, y así si un jugador coge una tarjeta con más de una acción, sólo deberá elegir una de ellas. Esta posibilidad no se contempla en varios de los juegos comentados en el Estado de la cuestión (ver capítulo 2.3.5, página 29).

El diseño de la clase *accion* se centra en aunar un diseño completo de la funcionalidad requerida con un diseño sencillo. La solución tomada como óptima en ambos sentidos consiste en crear una serie de atributos de dicha clase que recojan toda la información de las tarjetas que se pueden encontrar en cualquier versión del juego. Estos atributos se combinarán de la forma adecuada para dar como resultado acciones complejas.

Después de analizar las tarjetas de Monopoly, se han identificado las siguientes acciones representadas por los siguientes atributos:

- *_sgtServicio*: El jugador se deberá desplazar a la siguiente casilla de tipo servicio.
- *_sgtEstacion*: Se moverá el jugador a la siguiente estación del juego.
- *_sgtPropiedad*: El jugador se debe mover a la siguiente casilla de tipo calle.
- *_encarcelar*: Se encarcelará al jugador.

- *_avanzarA*: El jugador se deberá mover a la casilla indicada en este atributo.
- *_avanzar*: Se moverá al jugador la cantidad de casillas que indica el atributo.
- *_pagar*: El jugador debe pagar la cantidad indicada en el atributo. En caso de que se trate de un valor negativo, el jugador cobrará la cantidad.
- *_pagarCasa* y *_pagarHotel*: Se pagará la cantidad indicada por cada casa / hotel que haya construido.
- *_pagarCadaJugador*: Se indica el precio que se debe pagar a cada jugador de la partida. Al igual que se ha explicado en el atributo *_pagar*, un valor negativo de este atributo significa que el jugador cobrará del resto de participantes.
- *_multiplicarPago*: Si este atributo es un número mayor que uno, implicará que se modifique el valor del atributo *_multiPago* de la clase jugador. De esta forma se hace posible que el jugador pague en la siguiente propiedad el valor del alquiler multiplicado por esta cantidad.
- *_multiPagoDado*: La explicación es idéntica a la anterior, pero el valor por el que se multiplicará el alquiler se determinará por el lanzamiento de los dados.
- *_cobrarSalida*: Si se establece este atributo como falso, el jugador no cobrará si pasa por la casilla de salida en el caso de que se produzca un movimiento.
- *_tarjetaCarcel*: Este atributo se interpreta como una tarjeta que libera a un jugador de la cárcel. Una tarjeta que contenga una acción con este atributo positivo se podrá guardar para usarla más adelante.

En el caso de que la tarjeta sea de éste último tipo especificado, el jugador se quedará con la tarjeta para usarla como objeto de intercambio en una negociación o utilizarla cuando la necesite. Si un jugador se queda con una tarjeta, esta se eliminará del mazo y se añadirá a la lista de

tarjetas del jugador. Cuando éste la devuelva, se realizará el proceso inverso, colocándola al final de la cola de tarjetas.

Los mazos de tarjetas funcionarán como una cola (LIFO), exceptuando las tarjetas de cárcel que las podrá poseer un jugador.

4.2.4 Los propietarios

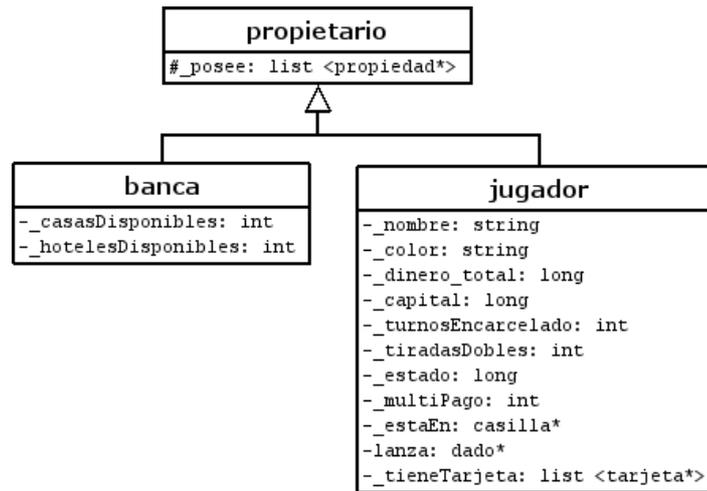


Diagrama 6: Diagrama de clases (diseño de los propietarios)

En el juego se han identificado dos tipos de propietarios para las propiedades de cualquier tipo. Éstos propietarios son la banca y los jugadores, siendo ambos clases derivadas de la clase base *propietario*.

La jerarquía que se presenta en esta parte del diseño representa fielmente la realidad, ya que creando los dos tipos de propietarios se puede representar correctamente el traspaso de propiedades que se llevan a cabo durante el juego.

El primer propietario de todas las propiedades es la banca, y si un jugador desea comprar una propiedad libre deberá comprársela a la banca. La banca no podrá recuperar esa propiedad a no ser que el jugador se declare en bancarrota y nadie puje por ella.

Las instancias de la clase *jugador* tienen todos los atributos necesarios para conocer su estado en cada momento: si tiene deudas o le deben dinero (*_estado*), los turnos en la cárcel, su nombre, su color de ficha y todas sus posesiones.

4.2.5 El papel de la banca

Puede resultar chocante que la clase *banca* no contenga ningún atributo que haga referencia al dinero. Esto tiene una explicación clara, que se basa en las reglas del juego original en el que se basa Street Master's. La banca tiene un papel importante en el juego, ya que debe ser ella la encargada de distribuir los bienes entre los jugadores. Las normas dicen en referencia al dinero de la banca:

“Si la banca se queda sin dinero, el banquero pone en circulación pagarés con la cantidad que sea necesario.”

Por lo tanto, se entiende que el dinero de la banca es infinito, por lo que no será necesario que se almacene cuanto dinero tiene.

4.2.6 Las negociaciones entre jugadores

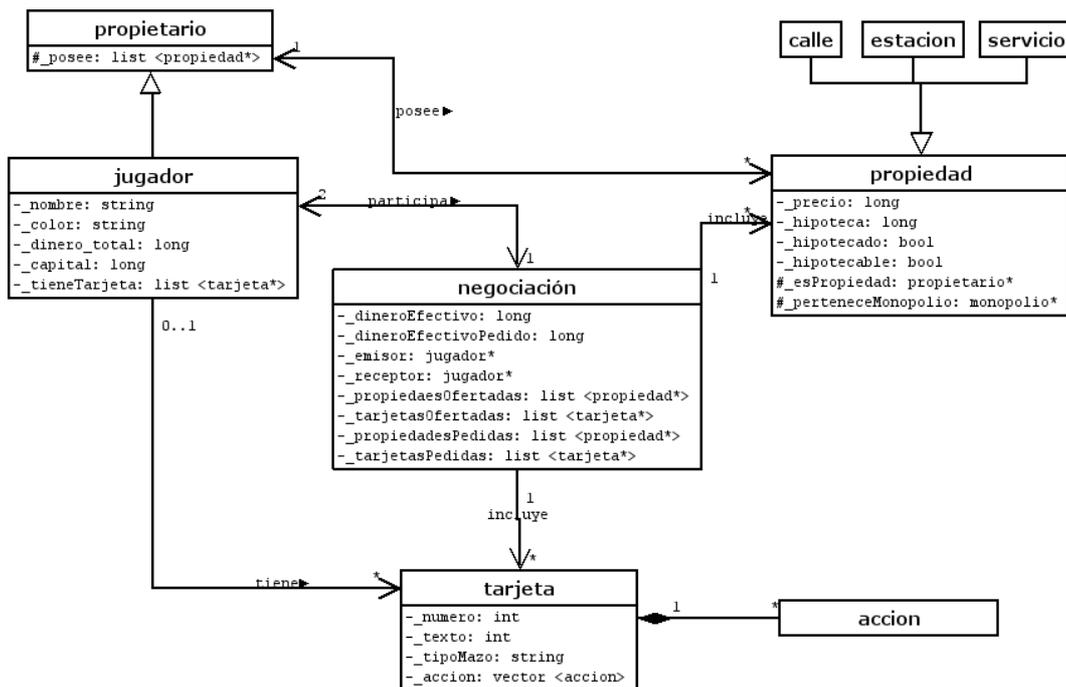


Diagrama 7: Diagrama de clases (diseño de las negociaciones)

El juego presenta la posibilidad de que se establezcan negociaciones entre jugadores. Este hecho se ha representado en el diseño mediante la clase *negociación*. En esta clase se incluyen todos los posibles intercambios que pueden darse entre dos jugadores.

Una norma obligatoria de las negociaciones es que se establezcan entre dos jugadores, por lo que se han definido los atributos *_emisor* y *_receptor*, que serán los protagonistas de la negociación. Una negociación debe indicar qué se ofrece (atributos *_dineroEfectivo*, *_propiedadesOfertadas* y *_tarjetasOfertadas*) y qué se pide a cambio (el resto de los atributos).

Las propiedades y tarjetas que se incluyan en la negociación, obviamente deben pertenecer a los jugadores que intervienen en ella.

No existe ningún otro objeto en el juego que pueda incluirse dentro de una oferta, por lo que no se ha planteado la posibilidad de incluir ningún otro atributo.

La vida de un objeto de la clase *negociación* durará desde que el emisor plantea la oferta hasta que el receptor la declina o acepta y se produce, definitivamente, el intercambio del tipo negociado.

4.2.7 El cliente de IA

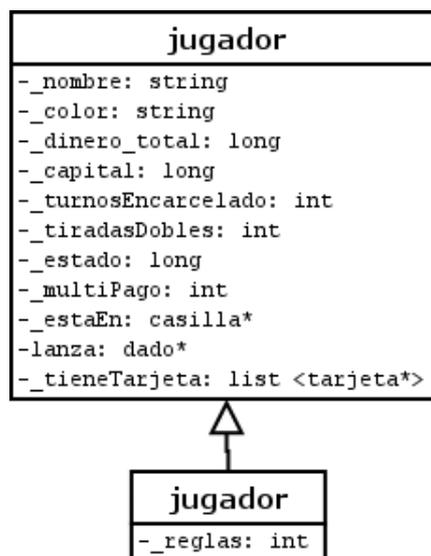


Diagrama 8: Diagrama de clases (cliente IA)

Se ha creado una clase específica para el cliente de IA que juega a Street Master's: clase *jugador_reglas*.

Se ha definido un atributo numérico (*_reglas*) en la clase *jugador_reglas* que permite diferenciar entre diferentes estrategias de juego. Actualmente, existen dos jugadores automáticos, uno que sigue una secuencia de reglas preestablecidas simulando una estrategia agresiva (véase sección 4.3.1,

página 68) y otro que sigue un comportamiento pseudo-aleatorio – sección 4.3.3, página 84. Se deja la puerta abierta a futuras ampliaciones en este sentido.

Esta clase hereda todos los datos desde la clase jugador y tomará las decisiones oportunas en cada situación basándose en las reglas que se explican en el siguiente apartado.

4.3 *Discusión de las reglas del cliente de IA*

Se han diseñado dos clientes de IA. Uno de ellos estará creado mediante un sistema basado en reglas que se ha diseñado buscando un agente capaz de competir con solvencia frente a jugadores humanos.

El otro cliente implementado consiste en un sencillo cliente de IA con un comportamiento pseudo-aleatorio, el cual podrá dar resultados dispares.

El objetivo que se busca al crear ambos clientes de IA es que el primero de ellos sea capaz de jugar lo mejor posible a Street Master's. Para ello, deberá ganar al jugador aleatorio el mayor número de partidas posibles.

A continuación se explicarán las decisiones tomadas en ambos casos.

4.3.1 *Cliente de IA basado en reglas*

Se ha diseñado un agente de IA para jugar a Street Master's mediante sencillas reglas. Este agente tomará las decisiones sobre todas las funcionalidades del juego excepto las negociaciones que se deshabilitarán todas las veces que jueguen jugadores automáticos.

Se ha creado un jugador capaz de razonar las decisiones que deba tomar a partir de sus propios datos y los de sus competidores.

Street Master's es un juego de n-agentes en el que se pueden dar colaboraciones y alianzas entre jugadores. La colaboración más común es la negociación entre jugadores que, como se ha comentado, queda desactivada cuando participan jugadores automáticos, pero también existe la posibilidad de hacer alianzas durante las subastas, lo cual se explicará con detalle a lo largo de este capítulo (sección 4.3.1.1 Pujar, página 70)

Los jugadores en este juego pueden tener actitudes muy agresivas gastando mucho dinero sin tener en cuenta las posibles consecuencias que pueda tener en el futuro. Otra actitud que suelen tomar los jugadores en este juego es bastante conservadora, gastando sólo lo imprescindible. La táctica considerada para el desarrollo del cliente de IA es eminentemente agresiva, pero guardando siempre un colchón de seguridad para evitar caer en riesgos innecesarios.

En este punto se explicarán las decisiones tomadas para cada uno de los operadores implementados para el agente inteligente:

- Pujar.
- Decidir si se compra una propiedad.
- Tomar decisiones para salir de la cárcel.
- Decidir entre varias acciones de una tarjeta.
- Elegir qué se pagará al caer en una casilla de impuestos.
- Hipotecar y deshipotecar propiedades.
- Construir y vender edificios de una calle.
- Pagar.
- Declararse en bancarrota

En muchas de las reglas de los operadores que se describirán a continuación aparecen pesos computados manualmente. Los valores de estos pesos se han obtenido empíricamente, tomando como valores definitivos los que han mejorado el comportamiento del agente. En función su valor, el jugador presenta un compartimento u otro.

La selección de las reglas que se ejecutan en cada momento se realiza comprobando cuál es la primera de ellas que cumple las condiciones actuales del sistema para cada operador, siguiendo el orden que se detalla en cada operador para realizar las comprobaciones.

4.3.1.1 Pujar

Se han diseñado una serie de reglas que determinan la cantidad a pujar dentro de una subasta conociendo la puja más alta que se ha hecho hasta ese momento.

Las reglas decidirán en cuánto se incrementará la puja máxima en función de diferentes intereses. Obviamente, se pujará más fuerte si la propiedad es de máximo interés por completar un monopolio o acercarse a ello.

Existe una pequeña colaboración entre rivales si uno de los jugadores puede conseguir completar un monopolio con la casilla que se está subastando (regla número 4). Lógicamente, al resto de los jugadores no les interesa que alcance ese objetivo. En ese caso, un jugador no pujará si el jugador que ha realizado la puja máxima no es el jugador que puede completar el grupo dándose la posibilidad de que se alíen varios jugadores contra otro. Esta regla es la parte más interesante del operador 'pujar'.

Las reglas que se han definido son las siguientes:

1.1	SI puja máxima > dinero disponible ENTONCES No pujar
1.2	SI puja máxima = 0 \wedge precio (propiedad) < dinero disponible ENTONCES pujar (0,5 \times precio (propiedad))
1.3	SI puja máxima \times 2 < dinero disponible \wedge Es ultima propiedad monopolio (jugador) ENTONCES pujar (1,3 \times puja máxima)
1.4	SI puja máxima \times 2 < dinero disponible \wedge Es última propiedad monopolio (propietario (puja máxima)) ENTONCES pujar (1,1 \times puja máxima)
1.5	SI único poseedor monopolio (jugador) \wedge puja máxima \times 2 < dinero disponible ENTONCES pujar (1,2 \times puja máxima)
1.6	SI puja máxima \times 10 < dinero disponible \wedge puja máxima < precio (propiedad) ENTONCES pujar (1,3 \times puja máxima)
1.7	SI puja máxima \times 3 < dinero disponible \wedge puja máxima < precio (propiedad) ENTONCES pujar (1,1 \times puja máxima)

4.3.1.2 Decidir la compra de propiedades

Según avanza el juego, no todas las propiedades tienen el mismo valor para cada jugador, máxime cuando el dinero comienza a escasear. Por ello, se ha diseñado un sistema de reglas para comprar propiedades que tenga en cuenta todos los factores que pueden influir en la decisión: dinero que se posee, propiedades que se tienen actualmente y el interés estimado que puedan tener el resto de jugadores por la propiedad en cuestión.

Esto se tiene en cuenta para todos los tipos de propiedades, pero se han definido diferentes reglas para cada uno de los tipos, ya que el interés que puede tener un jugador por una propiedad de tipo estación, no es el mismo que puede tener por una propiedad de tipo calle, ya que los beneficios previstos no son los mismos.

Las reglas buscarán por lo tanto maximizar el beneficio del jugador que debe tomar la decisión, intentando completar el máximo número de monopolios posibles, y tratando de perjudicar en la medida de lo posible los planes del resto de jugadores, evitando que formen los preciados monopolios.

A continuación se muestran las reglas creadas para cada uno de los tipos de propiedad.

4.3.1.2.1 Calles

Las calles son las propiedades más preciadas del juego, por lo que se comprarán todas aquellas que sean útiles en algún sentido, teniendo menos reparos en mantener parte del dinero sin gastar.

Cuando se expongan las reglas de los otros tipos de propiedades se verá que no ocurre lo mismo y se comprobará entonces como se deja algo de dinero disponible sin gastar (véase las secciones 4.3.1.2.2 y 4.3.1.2.3).

2.1.1	SI precio (calle) > dinero disponible ENTONCES ~comprar (calle)
2.1.2	SI Es ultima propiedad monopolio (jugador) ENTONCES comprar (calle)
2.1.3	SI monopolio (libre) ENTONCES comprar (calle)

2.1.4	SI único poseedor monopolio (jugador) ENTONCES comprar (calle)
2.1.5	SI ~(único poseedor monopolio (jugador)) \wedge posee parte monopolio (jugador) \wedge precio (calle) \times 3 < dinero disponible ENTONCES comprar (calle)
2.1.6	SI Es ultima propiedad monopolio (~jugador) ENTONCES comprar (calle)
2.1.7	SI precio (calle) \times 7 < dinero disponible ENTONCES comprar (calle)

4.3.1.2.2 Estaciones

Las estaciones se consideran útiles si se puede llegar a tener más de una estación, ya que el alquiler aumenta considerablemente. Esta máxima se sigue en las siguientes reglas:

2.2.1	SI precio (estación) > dinero disponible ENTONCES ~comprar (estación)
2.2.2	SI posee estación (jugador) > 1 \wedge precio (estación) \times 3 < dinero disponible ENTONCES comprar (estación)
2.2.3	SI estaciones libres \geq 2 ENTONCES comprar (estación)
2.2.4	SI precio (estación) \times 7 < dinero disponible ENTONCES comprar (estación)

4.3.1.2.3 Servicios

Los servicios, como se puede ver en las reglas se consideran menos importantes que otros tipos de propiedades, ya que su influencia en el juego es mucho menor, debido a su bajo alquiler. Por ello, se intentará salvar parte del dinero del jugador antes de comprar un servicio ya que no merece la pena arriesgarse a caer en bancarrota por unos beneficios menores, como se muestra en las siguientes reglas:

2.3.1	SI precio (servicio) > dinero disponible ENTONCES ~comprar (servicio)
-------	--

2.3.2	SI posee servicio (jugador) > 0 \wedge precio (servicio) \times 2 < dinero disponible ENTONCES comprar (servicio)
2.3.3	SI servicios ocupado = 0 \wedge precio (servicio) \times 2 < dinero disponible ENTONCES comprar (servicio)
2.3.4	SI precio (servicio) \times 7 < dinero disponible ENTONCES comprar (servicio)

4.3.1.3 Decidir salir de la cárcel

En Street Master's, no siempre es recomendable salir de la cárcel, ya que una vez que todas las propiedades del tablero han sido compradas, la mejor protección para no pagar, es permanecer en la cárcel. Mientras tanto, el jugador encarcelado seguirá cobrando al resto.

Por ello, se han creado las reglas pensando en esta teoría, tanto para salir pagando la multa, como para salir usando una tarjeta de suerte o caja de comunidad.

4.3.1.3.1 Salir de la cárcel pagando la multa.

Se hacen estimaciones de la conveniencia de salir de la cárcel en función del dinero que tenga el jugador. Cuanto más dinero tenga el jugador, menos importará que queden pocas casillas libres en el tablero porque es más posible que sea capaz de hacer frente a cualquier deuda y de esta manera podrá seguir comprando propiedades. Si no quedaran propiedades, se considera recomendable permanecer en la cárcel.

A continuación se muestran las reglas en las que se han definido los parámetros que indican cuando resulta conveniente salir de la cárcel y cuando no.

3.1.1	SI propiedades sin dueño = 100% \wedge multa < dinero disponible \wedge dinero disponible > 50000 ENTONCES pagar (multa)
3.1.2	SI propiedades sin dueño \geq 50% \wedge 3 \times multa < dinero disponible \wedge dinero disponible > 50000 ENTONCES pagar (multa)

3.1.3	SI propiedades sin dueño $\geq 20\% \wedge 5 \times multa < dinero disponible$ $\wedge dinero disponible > 50000$ ENTONCES pagar (multa)
-------	--

4.3.1.3.2 Salir de la cárcel usando una tarjeta.

En este caso, si no quedan al menos un porcentaje importante de casillas libres en el tablero y no se tiene una cantidad de dinero importante, se considera mejor permanecer encarcelado.

3.2.1	SI propiedades sin dueño $\geq 20\% \wedge dinero disponible > 50000$ ENTONCES devolver (tarjeta)
-------	--

4.3.1.4 Decidir entre varias acciones de una tarjeta

Las reglas creadas para elegir entre una de las posibles acciones que se ofrezcan en una tarjeta consiste simplemente en escoger la mejor opción.

La elección vendrá dada por las necesidades de cada momento, tomando como preferencias las acciones de cobrar y obtener licencias para salir de la cárcel.

4.1	SI existe acción (salir de la cárcel) $\wedge dinero disponible > 50000$ ENTONCES elegir (acción salir de la cárcel)
4.2	SI existe acción (cobrar) ENTONCES elegir (acción cobrar)
4.3	SI \sim existe acción (salir de la cárcel) $\wedge \sim$ existe acción (cobrar) ENTONCES elegir (acción por defecto)

4.3.1.5 Elegir cantidad a pagar en un impuesto especial

Esta regla es muy sencilla ya que tan sólo consiste en elegir el pago más pequeño entre el porcentaje del capital y la cantidad fija.

5.1	SI cantidad \leq porcentaje (capital (jugador)) ENTONCES pagar (cantidad)
5.2	SI cantidad $>$ porcentaje (capital (jugador)) ENTONCES pagar (porcentaje (capital (jugador)))

4.3.1.6 *Pagar*

El operador pagar es necesario en multitud de ocasiones. Alquilar una propiedad, al caer sobre una casilla de impuestos o recoger tarjetas de un mazo que ordenen hacerlo son un ejemplo de utilización de este operador.

Para poder pagar una cierta cantidad de dinero hay una condición indispensable, tener dinero suficiente. Si no se tiene dinero suficiente, existen dos formas de conseguirlo: vender edificios e hipotecar propiedades. Si con ninguna de estas fórmulas se alcanza el dinero suficiente para hacer frente a la deuda, será imposible pagar y se declarará entonces la bancarrota (ver sección 4.3.1.7, página 75).

A continuación se muestran las reglas implementadas, siendo *cantidad* la cantidad a pagar:

6.1	SI cantidad > dinero disponible \wedge existen calles construidas ENTONCES vender edificios
6.2	SI cantidad > dinero disponible \wedge existen propiedades hipotecables ENTONCES hipotecar
6.3	SI cantidad \leq dinero disponible ENTONCES pagar
6.4	SI cantidad > dinero disponible ENTONCES declarar bancarrota

4.3.1.7 *Declarar bancarrota*

Cuando un jugador no puede hacer frente a sus deudas y no puede reunir dinero de ninguna forma posible, debe declararse en bancarrota.

8.1	SI cantidad > dinero disponible \wedge \sim existen calles construidas \wedge \sim existen propiedades hipotecables ENTONCES declarar bancarrota
-----	--

Con las reglas explicadas en este apartado de la memoria se ha consigo dotar al juego con un agente de IA basado en reglas que juega de una forma automática, tomando decisiones con sentido ante las diferentes disyuntivas que se presentan durante el juego.

4.3.2 Algoritmo de búsqueda sin información para procesos de hipotecas y construcciones

La discusión que se realiza en esta sección sobre el algoritmo de búsqueda sin información sólo hace referencia al agente de IA basado en reglas expuesto en la sección anterior.

Los operadores más elaborados del agente de IA son la toma de decisiones para hipotecar, deshipotecar, comprar y vender edificios, ya que una mala gestión de estos procesos reducirían drásticamente sus opciones de victoria.

Para ello, se ha desarrollado un algoritmo de búsqueda sin información que toma como criterio para seleccionar la mejor opción la *teoría de la utilidad esperada*.

La teoría de la utilidad esperada explica como medir la utilidad que siempre satisface un criterio de *riesgo – neutral*, basándose en la aversión al riesgo que tienen las personas en el mundo real, ya que tienden a elegir resultados seguros, a pesar de existir opciones con un mayor beneficio pero que implican un riesgo mayor. Esta teoría se suele aplicar para resolver problemas encuadrados dentro de la teoría de juegos.

La utilidad esperada se calcula multiplicando el beneficio estimado de una situación por la probabilidad de que esta ocurra. En el juego del Monopoly, el beneficio (o pérdida) estimado se calcula mediante los alquileres de las propiedades y la probabilidad es la probabilidad de caer en una casilla.

$$U.E. = (\text{alquiler}_{\text{propiedad}_1} + \text{alquiler}_{\text{propiedad}_2} + \dots + \text{alquiler}_{\text{propiedad}_n}) \times \frac{1}{\text{numero}_{\text{casillas}}}$$

Cuando un jugador deba tomar una decisión para hipotecar, deshipotecar, comprar o vender edificios, deberá realizar una búsqueda exhaustiva entre todos los posibles subconjuntos generados a partir de sus propiedades. Para ello, se ha desarrollado un algoritmo de búsqueda del *primero en profundidad*.

Formalmente, este algoritmo es un algoritmo de *búsqueda sin información*, que expande y examina todos los nodos de un árbol sistemáticamente para buscar una solución sin usar ninguna heurística.

El siguiente pseudocódigo se aplicará para resolver todas las situaciones mencionadas:

```
vector alternativas (numero cantidad, vector L, numero índice)
{
  //Si llega al ultimo elemento se debe devolver falso
  SI ( índice == ultimo elemento )
    devolver falso;
  //Si se cumple la condición de parada propia de cada caso, se devuelve el
  //elemento actual
  SI ( condición de parada )
    devolver elemento actual;
  //En cualquier otro caso se hará una llamada recursiva ha todos los
  //siguientes elementos de la lista para cubrir todas las posibles
  //alternativas
  Recorrer la lista desde i = index mientras i != último elemento
  {
    candidato = candidato + alternativas (cantidad - calcularBeneficio
      (elemento actual), lista, iniciar en el siguiente elemento);
    //Si se cumple la condición de factibilidad propia de cada caso, se
    //deberá comprobar si es la mejor opción existente hasta el momento
    SI ( control de factibilidad )
      //Si es la mejor opción, se almacena en una variable
      SI ( condición de utilidad esperada )
        mejor = candidato;
    i = i + 1
  }
  //Se devolverá la mejor opción al final de todas las llamadas recursivas
  devolver mejor;
}
```

Algoritmo 1: Algoritmo de búsqueda sin información

Es importante que la lista de elementos que se introduce en el algoritmo esté ordenada de mayor a menor coste.

Este algoritmo será el mismo para todos los casos que se tratan a continuación, con la única diferencia de las condiciones de *parada*, *control de factibilidad* y de *cálculo de la utilidad esperada* (y que se muestran en cursiva en el pseudocódigo anterior).

- La *condición de parada* comprueba cuando se debe detener el algoritmo por haber satisfecho las condiciones impuestas.
- El *control de factibilidad* será distinto en cada caso y comprobará que la opción seleccionada cumpla con las normas del juego y con las necesidades impuestas por la situación. Por ejemplo, no se puede gastar más dinero del que se dispone.

- Por último, hay que comprobar si la opción generada es mejor que la que actualmente ostenta este estado. Esta comparación se realizará utilizando la *teoría de utilidad esperada*.

Como estas condiciones son particulares para cada uno de los casos, se explicarán más adelante para cada uno de ellos (a partir de la página 80).

Para ilustrar el funcionamiento del algoritmo se muestra un sencillo ejemplo de ejecución del mismo. Suponiendo el tablero de juego que se proporciona en el apéndice A de esta memoria (página 148), un jugador posee las propiedades 1, 6, 11, 24, 34 y 39 y desea obtener una cantidad de dinero 32000 unidades monetarias hipotecándolas. Para buscar la solución óptima se deberá buscar el subconjunto que presente una menor utilidad esperada, ya que se trata de alcanzar una cantidad de dinero concreta minimizando las pérdidas.

Propiedad	39	34	24	11	6	1
Alquiler	5000	2800	2000	1000	600	200
Hipoteca	20000	16000	11000	7000	5000	3000

A continuación, se muestra el árbol de búsqueda generado. Para mayor claridad se muestra cada rama por separado.

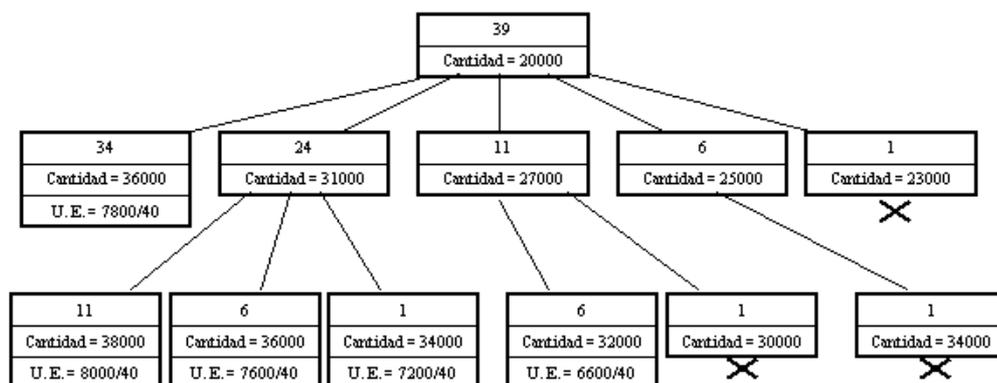


Imagen 11: Árbol de búsqueda con la raíz de la rama en la propiedad 39

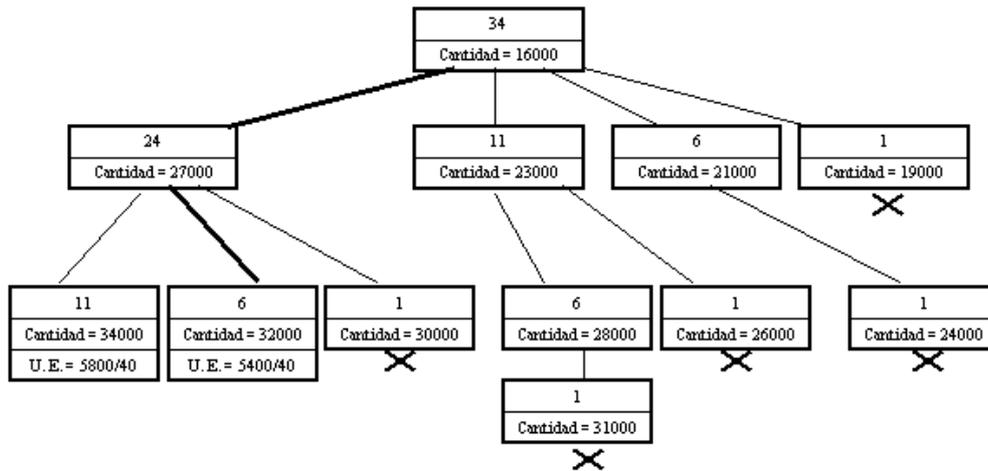


Imagen 12: *Árbol de búsqueda con la raíz de la rama en la propiedad 34.*

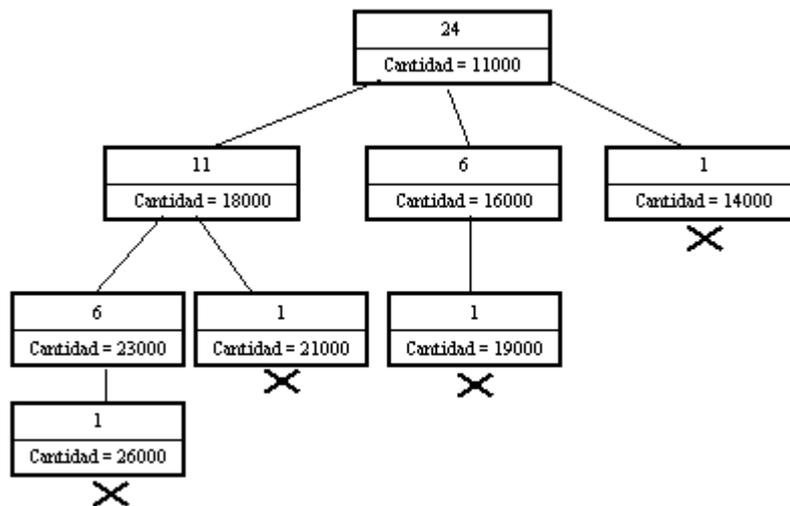


Imagen 13: *Árbol de búsqueda con la raíz de la rama en la propiedad 24.*

El resto de las posibilidades generadas a partir de las propiedades 11, 6 y 1 no pueden alcanzar la cantidad de dinero necesaria por lo que no aportan información de utilidad.

Como se puede observar en el árbol, el subconjunto de propiedades con el que se consigue una cantidad de dinero suficiente y además presenta una utilidad esperada menor (sacrificando menor beneficio) es el que forman las propiedades 34, 24 y 6. Por lo tanto, el agente hipotecaría estas propiedades.

Este es un ejemplo de hipotecas de propiedades, que se discutirá con detenimiento en la sección 4.3.1.7.1, pero el resto de los casos tiene un funcionamiento similar, realizando los cambios pertinentes en el algoritmo tal y como se explica en las siguientes secciones.

4.3.2.1 Hipotecar

Hipotecar es algo poco deseable en Street Master's, ya que si se posee una propiedad, lo apetecible es poder sacar beneficio de ella. Por lo tanto, un agente de IA sólo hipotecará sus propiedades cuando tenga una necesidad insalvable de dinero, e intentará obtener la cantidad de dinero requerida hipotecando el subconjunto de propiedades que impliquen una pérdida esperada menor.

Este objetivo se alcanza usando el algoritmo que se ha explicado anteriormente tomando como lista L el conjunto de propiedades del jugador ordenadas de mayor a menor coste. Como se puede ver en el algoritmo (véase algoritmo 1), existen tres condiciones importantes: *condición de parada del algoritmo*, *control de factibilidad* y *cálculo de la utilidad esperada*.

- La *condición de parada* en el caso de las hipotecas es que la cantidad de dinero que falta por cubrir durante la generación de un subconjunto, sea menor o igual que la que se puede conseguir hipotecando la propiedad actual. Si añadiéndola al subconjunto se alcanza la cantidad necesaria, se habrá encontrado un candidato a ser la mejor opción y no se continuarán insertando más propiedades al subconjunto.
- El *control de factibilidad* en este caso es muy sencillo. Simplemente hay que comprobar que el candidato es un subconjunto válido que genera una cantidad de dinero superior a la necesaria. Esta comprobación es necesaria ya que además de por la condición de parada, el algoritmo se detiene al llegar al final de la lista de propiedades, generando una alternativa no válida.
- Por último, se debe definir la condición del *cálculo de la utilidad esperada*. Esta condición sirve para comprobar si el subconjunto generado es la mejor opción posible. Para ello, se debe comprobar

si el cálculo de la utilidad esperada para dicho subconjunto es menor que la de la mejor opción encontrada hasta el momento. Con esto se consigue obtener el subconjunto que sacrifica el menor beneficio posible.

Un ejemplo de la utilización de este algoritmo es el mostrado en la sección anterior.

4.3.2.2 Deshipotecar

En este caso, se deshipotecarán propiedades siempre que se tenga dinero suficiente para hacerlo. Para evitar que un agente corra excesivos riesgos al deshipotecar propiedades, se ha determinado que la cantidad máxima que se puede invertir en este tipo de operaciones es el 70 % del dinero disponible.

El objetivo de este algoritmo es encontrar aquel subconjunto de propiedades hipotecadas que se puedan deshipotecar con el dinero disponible en un momento dado y suponga el máximo beneficio posible (máxima utilidad esperada).

En este caso, la lista estará formada por las propiedades hipotecadas. Las tres condiciones serán las siguientes.

- La *condición de parada* debe controlar que se alcance la cantidad justo anterior a llegar al máximo, y esto ocurrirá cuando no haya ninguna propiedad en la lista con un precio de deshipoteca menor que la cantidad que falta por cubrir.
- El *control de factibilidad* en este caso consiste en comprobar que no se haya superado la cantidad de dinero disponible y por lo tanto se pueda hacer efectiva la operación.
- Para maximizar el beneficio se debe comprobar que la *utilidad esperada* del nuevo subconjunto obtenido sea mayor que el ya identificado como mejor previamente. De esta manera se consigue obtener el subconjunto que se prevé que ofrezca un mayor beneficio.

El agente decidirá las deshipotecas al final de cada movimiento en el juego.

4.3.2.3 Vender

El caso de las ventas de edificios es muy similar al de las hipotecas explicado en la sección 4.3.2.1 (página 80), ya que ambos tienen como objetivo obtener una cantidad de dinero suficiente para saldar sus deudas. Si se observa cualquiera de las tarjetas de propiedad del juego de Monopoly, se puede comprobar que el beneficio obtenido por la venta de un edificio es mínimo en comparación con los que se pueden obtener del alquiler de propiedades edificadas.

En el caso de las ventas de edificios la lista del algoritmo de búsqueda representa las calles en las que se puede eliminar un edificio. Por ejemplo, si en una calle hay 4 casas construidas, ésta aparecerá 4 veces en la lista.

La complejidad de la gestión de edificios radica en respetar la compleja normativa del juego del Monopoly en el ámbito de las construcciones (véase las reglas del juego en el capítulo 2.3.1, página 21).

Las condiciones que se deben incluir en el algoritmo de búsqueda son las siguientes:

- La primera condición (*condición de parada*) hace que el algoritmo se detenga cuando con el precio de venta del siguiente edificio de la lista se alcance la cantidad de dinero que se necesita reunir. De esta forma se devuelve como alternativa de venta el subconjunto que incluye dicho edificio.
- El *control de factibilidad* es bastante complejo. A la condición de superar la cantidad indicada hay que sumarle otras que hagan cumplir las reglas del juego. Estas condiciones deben comprobar que la eliminación de casas dejen el monopolio en un estado correcto (la máxima diferencia de casas entre calles debe ser uno) y que haya edificios disponibles en la banca.
- El cálculo de la utilidad esperada es idéntico al que se ha explicado en el caso de las hipotecas, es decir, buscar la mínima suma de alquileres entre los subconjuntos posibles. En este caso hay que tener especial cuidado al obtener el alquiler en función de las casas que se deseen vender.

En el caso de que no sea posible obtener la cantidad necesaria con ninguna combinación, se venderán todas las casas construidas para liberar a todas las calles para que puedan ser posteriormente hipotecadas.

4.3.2.4 *Construir*

Las construcciones son una operación fundamental en el juego. Por ello, el agente deberá tener en cuenta que debe construir siempre que sea posible para aumentar el precio de alquilar cualquiera de sus propiedades. Sin embargo, se ha decidido que el jugador no invierta una cantidad de dinero que suponga un riesgo de quiebra y se ha establecido como inversión máxima el 70% de su capital. Además, el jugador sólo invertirá cuando tenga más de 50.000 unidades monetarias, que se considera una cantidad de dinero suficiente para no correr riesgos innecesarios.

El agente de IA decidirá si construir al final de cada turno, después de haber deshipotecado todas las propiedades que considera oportuno (véase la sección 4.3.2.2, página 81)

Del mismo modo que se ha explicado en el apartado anterior para el caso de las ventas de edificios, la lista representa el conjunto de edificios que se pueden construir sobre las propiedades de un jugador, añadiendo un propiedad a la lista tantas veces como edificios se puedan construir sobre ella.

Las condiciones que se deben incluir en el algoritmo tienen como objetivo construir las casas y hoteles que aporten un beneficio máximo siguiendo la *teoría de la utilidad esperada* y cuyo precio total de construcción no supere la cantidad destinada para ello. A continuación se discuten dichas condiciones:

- La *condición de parada* debe controlar que el algoritmo finalice cuando no exista ningún edificio en la lista, a partir de la situación de la propiedad actual, que tenga un precio inferior a la cantidad de dinero disponible.
- El *control de factibilidad* es muy similar al explicado en el caso de las ventas (sección 4.3.1.7.3). Se debe comprobar se respetan las reglas del Monopoly y además, que no se supere la cantidad de dinero disponible en la construcción del subconjunto candidato.

- El *cálculo de la utilidad esperada*, una vez más, consiste en sumar los alquileres de las calles una vez realizadas las operaciones de compra de edificios. Si el resultado obtenido es mayor que la mejor opción encontrada hasta el momento, se selecciona el nuevo candidato como mejor opción.

4.3.3 Cliente de IA basado en un comportamiento aleatorio

El cliente regido por un comportamiento aleatorio es mucho más sencillo de explicar, ya que la mayor parte de sus decisiones no responden a un razonamiento lógico. El potencial del cliente de IA descrito anteriormente, debe ser muy superior a éste, como se verá en el capítulo de resultados (ver capítulo 5, página 117).

Los operadores de elección de cantidad de dinero a pagar en una casilla de impuesto especial, declarar bancarrota y selección de la acción de la tarjeta de suerte o caja de comunidad son iguales que los descritos anteriormente, ya que vienen dados por las instrucciones del juego (caso de la bancarrota) o hacerlo de otra forma supondría una gran desventaja para este jugador, como se ve claramente en el operador relacionado con los impuestos especiales.

El resto de operadores se basarán en condiciones probabilísticas en las que el resultado de la decisión se obtiene en función de los parámetros que intervienen en ella. De esta forma, la probabilidad de tomar una decisión será diferente en cada ocasión, dependiendo del resultado de esta función.

A continuación se explican estos operadores de una forma más detallada.

4.3.3.1 Pujar

La decisión de pujar o no, se tomará de una forma aleatoria, pero teniendo en cuenta el dinero del que dispone el jugador y la cantidad máxima pujada.

Para ello, se calculará un número entre 0 y 99 usando el algoritmo de Mersenne Twister¹⁸, y se comparará con la cantidad obtenida de la siguiente fórmula:

$$P = (\text{Dinero disponible} - \text{puja máxima}) \times 100 / \text{dinero disponible}$$

Si la cantidad obtenida de esta forma es menor que la calculada con esta fórmula, se pujará por encima de la puja máxima. En caso contrario, el jugador saldrá de la puja.

De esta forma se evita que el jugador puge por encima de sus posibilidades y cuanto más alta es la puja es menos probable que lo haga porque el número obtenido es menor. Por ejemplo si el resultado obtenido en esta función es 70, existirá un 70% de probabilidades de que decida pujar por encima.

Se ha decidido que el jugador puge una única unidad monetaria por encima de la cantidad establecida. Esto se ha pensado así, para no dejar en desventaja al jugador aleatorio frente al resto.

El funcionamiento de la mayoría de los operadores seguirá la misma estructura por lo que las explicaciones sucesivas se referirán exclusivamente al diseño de la función de probabilidad.

4.3.3.2 Decidir la compra de propiedades

La probabilidad de comprar una propiedad se calculará en función del dinero disponible y de su precio. Esta probabilidad se calculará siguiendo la siguiente fórmula.

$$P = (\text{Dinero disponible} - \text{precio}) \times 100 / \text{dinero disponible}$$

En este caso, también se obtendrá un número aleatorio que se comparará con este resultado de la misma manera que en el caso anterior. Si el valor aleatorio es menor que el calculado, se comprará la propiedad.

4.3.3.3 Decidir salir de la cárcel

Una vez más, la probabilidad de salir de la cárcel vendrá dada por dos parámetros. En este caso no serán cantidades monetarias, si no que serán

¹⁸ Ver el capítulo 4.4 para más información. Siempre que se hable de aleatoriedad en este cliente de IA se referirá al algoritmo de Mersenne Twister.

el número de propiedades totales y las del jugador las que intervendrán en el cálculo.

$$P = (\text{número de propiedades del jugador}) \times 100 / \text{número total de propiedades}$$

Si se trata de salir de la cárcel por medio de la devolución de una tarjeta, será necesario comprobar que el jugador disponga de dicha tarjeta. Si se trata de salir de la cárcel mediante el pago de una multa, se tendrá que comprobar si dispone de dinero suficiente para hacer el pago.

Nuevamente, la decisión se tomará en función de un número aleatorio que se comparará con el resultado obtenido en la fórmula anterior.

4.3.3.4 Hipotecar propiedades

Las hipotecas serán necesarias como último recurso si no se tiene dinero suficiente para hacer frente al pago. En este caso es necesario asegurar que se va a hipotecar el número de propiedades necesarias para conseguir el dinero total, incluso si fuera necesario se deberían hipotecar todas las propiedades.

Para asegurar que esto ocurra hipotecando el mínimo número de propiedades, se deberán hipotecar las propiedades una a una empezando por la más cara.

4.3.3.5 Deshipotecar propiedades

Para tratar las deshipotecas, se vuelve a tratar el caso de las probabilidades. La probabilidad de deshipotecar una propiedad, se calculará en función del coste de deshipotecarla y del dinero disponible.

$$P = (\text{dinero disponible} - \text{precio de deshipoteca}) \times 100 / \text{dinero disponible}$$

Como ocurre en el resto de los casos, la decisiones se tomará en función de un número generado de forma aleatoria.

4.3.3.6 Construir edificios en las calles

La construcción de edificios se realizará sobre un monopolio concreto por turno. El monopolio se seleccionará de la misma forma que se ha resuelto el resto de los problemas, mediante un cálculo de probabilidades y un número aleatorio que determine si se lleva a cabo o no la construcción. Para ello se

recorrerán todos los monopolios hasta encontrar aquel que se deba construir en función del cálculo realizado.

Esta probabilidad se calculará mediante la fórmula:

$$P = (\text{dinero disponible} - \text{precio de un edificio}) \times 100 / \text{dinero disponible}$$

Siendo *precio* el del edificio del monopolio.

Una vez se haya seleccionado un monopolio para construir sobre él, se construirá sobre cada una de las calles mientras se siga dando la condición de que el número aleatorio sea menor que la probabilidad calculada con la fórmula anterior.

4.3.3.7 Venta de edificios

La venta de edificios se llevará a cabo de un forma similar a la que se explicó en el caso de las hipotecas.

Se deberán vender edificios hasta alcanzar una cantidad determinada. Si fuera necesario vender todas las casas y hoteles, así se deberá hacer.

En este caso, al igual que en la construcción de edificios, se deben tener en cuenta las reglas del juego en este sentido.

4.4 Generación aleatoria de números

Para resolver los casos en los que interviene el azar es necesario utilizar un generador de números pseudo-aleatorio. Para el proyecto Street Master's se debe aplicar el azar en los siguientes casos:

- Lanzar los dados
- Barajar las tarjetas de suerte y caja de comunidad
- Toma de decisiones del agente basado en decisiones aleatorias

Se ha decidido usar el generador de Merssene Twister [Matsumoto, M. y Nishimura, T., 1998] ya que ofrece unas series de números aleatorios mucho mejores que las que se pueden obtener mediante el generador `random()` (o alguna de sus versiones como `rand()` o `srand()`) del lenguaje de programación C.

4.5 Librería GNU Readline

Para mejorar y facilitar el uso de la consola del juego se ha utilizado la librería GNU Readline¹⁹.

Esta librería ofrece funciones que permiten editar la línea de comando que se está escribiendo. Las principales funcionalidades que aporta son la utilización de las flechas del teclado para desplazarse por el historial de comandos ejecutados y el uso del tabulador para auto completar los comandos que se están escribiendo en cada momento.

Un ejemplo de cómo trabajan estas funciones es la forma en que Bash realiza las operaciones antes descritas.

4.6 Manual de usuario

Una vez explicada la estructura y funcionalidad de Street Master's se pasa a explicar su instalación y uso.

4.6.1 Instalación de Street Master's

La instalación de Street Master's en un ordenador personal es muy sencilla, pero dicho ordenador debe cumplir ciertos requisitos básicos.

4.6.1.1 Requisitos mínimos

- Sistema operativo Linux Debian. La aplicación ha sido implementada y compilada bajo una plataforma Linux Debian, usando las clases y plantillas que ofrece la *librería STL de C++* y la librería *GNU Readline*.
- Para compilar el código fuente es necesario instalar el compilador `g++`.
- Es necesario instalar la librería GNU Readline para compilar y ejecutar el código de Street Master's, ya que ha sido utilizada para desarrollar la consola del juego.

4.6.1.1.1 Instalación

La instalación de Street Master's se realiza en tres sencillos pasos: *instalación de la librería Readline, configuración de la instalación y compilación de la aplicación.*

¹⁹ Se puede encontrar la librería Readline en la página web del proyecto GNU Readline, en ftp.gnu.org

4.6.1.1.2 Librería GNU Readline

Si ya se tiene instalada esta librería, se remite al lector al siguiente punto. Si no la tiene instalada, se presentan dos sencillas formas de hacerlo:

Instalar el paquete Debian

Instalar el paquete Debian (*.deb*) que incluye la librería *readline* es la opción más rápida y sencilla para aquellos usuarios que tengan acceso al sistema operativo como superusuario (*root*). Para ello hay que seguir los siguientes pasos:

- 1) Entrar como superusuario en el sistema

```
$ su
```

- 2) Instalar el paquete usando el comando *apt-get*.

```
$ apt-get install libreadline5-dev
```

Una vez concluido el proceso, la librería quedará instalada correctamente en el ordenador y ya estará lista para ser usada.

Compilar las fuentes descargadas de la web oficial

La otra opción existente es compilar e instalar el código fuente descargado desde el sitio oficial²⁰ de la librería. Para ello, se deben seguir los siguientes pasos:

- 1) Descargar el archivo *.tar.gz* (versión actual *readline-5.1.tar.gz*)
- 2) Descomprimir el archivo descargado.

```
$ tar -zvxf readline-5.1.tar.gz
```

- 3) Configurar la instalación.

```
$ ./configure
```

Si se desea se puede indicar el directorio de la instalación mediante el parámetro *--prefix=PATH*.

- 4) Compilar la librería

```
$ make
```

- 5) Por último, sólo resta instalar la librería en el ordenador

²⁰ ftp.gnu.org

```
$ make install
```

Para más opciones avanzadas de la instalación, se recomienda leer los archivos README e INSTALL que se incluyen en la distribución de la misma.

4.6.1.1.3 *Configurar la instalación*

Si se ha instalado la librería readline siguiendo la segunda opción en un directorio no estándar, es necesario cambiar en el archivo Makefile el directorio de dicha librería. Para ello se deberán eliminar los comentarios las siguientes líneas:

```
#INC-DIR = -I. -I../include -I/usr/X11R6/include -I/usr/local/include  
#LIB-DIR = -L. -L/usr/X11/lib -L -L/usr/local/lib -L/usr/X11R6/lib
```

En esas líneas se deberá indicar el emplazamiento de la instalación.

4.6.1.1.4 *Compilación e instalación de Street Master's*

La instalación de Street Master's es extremadamente sencilla. Simplemente hay que compilar el código fuente mediante el comando `make`.

4.6.2 Ejecución

Para ejecutar la aplicación se deberá lanzar el programa ejecutando `./street`. Una vez ejecutado el programa se iniciará la consola del juego, en la cual se pueden comenzar a utilizar los comandos propios del juego.

4.6.3 Jugar a Street Master's

Street Master's es un juego que permite que participen hasta 6 jugadores en una partida, de los cuales varios de ellos se pueden configurar como agentes de IA.

Una vez iniciada la aplicación se pueden comenzar a utilizar los comandos propios del juego, que ofrecen una funcionalidad completa para jugar a una versión para consola de Linux del popular juego de Monopoly.

Antes de comenzar la explicación del propio juego, se hacen notar los siguientes puntos:

- Es posible usar comandos simples de Linux en cualquier momento desde la consola del juego, sin necesidad de abandonar la partida.

- Al igual que en una consola de Linux, se puede navegar por los últimos comandos mediante las flechas del teclado, facilitando así la ejecución de comandos.
- Igualmente, se podrán autocompletar los comandos a ejecutar mediante el uso de la tecla de tabulación.

Una vez aclarado el funcionamiento básico de la consola de juego se pasa a explicar el juego en sí.

4.6.3.1 *Cómo jugar a Street Master's*

El juego se ha creado de manera que se puedan ejecutar todas las opciones que se contemplan en el reglamento mediante sencillos comandos. El funcionamiento del juego se detalla en las siguientes secciones.

4.6.3.1.1 *Inicio del juego*

Una vez ejecutado el programa, lo primero que se debe hacer es abrir una partida para jugar. Para ello, se puede crear una partida nueva o cargar una partida guardada con anterioridad.

- Partida nueva

Para crear una partida nueva se debe teclear el comando 'nuevo' y automáticamente el programa iniciará su creación.

En primer lugar, el programa preguntará si se desea cargar la configuración por defecto del juego. Si no se está seguro, se recomienda decir que sí, ya que en caso contrario se preguntará por datos que posiblemente no se conozcan.

A continuación, se deben introducir los datos de los jugadores (nombre y color de la ficha). En este punto se podrá decidir si se quiere incluir jugadores manejados por el ordenador.

```

*****
                Street Master's v. 1.0
*****

>Para obtener ayuda teclee 'ayuda'
street master's> nuevo
¿Desea cargar la configuración por defecto del juego?(recomendado) (s/n)s

Introduzca los siguientes datos para generar su partida
  Número de jugadores (2 - 6): 4
-Introduzca los siguientes datos para crear los jugadores:
JUGADOR 1
¿Quieres que sea un jugador controlado por el ordenador? (s/N) n
  -Nombre: Javier
  -Color ( amarillo rojo azul verde naranja blanco ): amarillo

JUGADOR 2
¿Quieres que sea un jugador controlado por el ordenador? (s/N) s
  -Nombre : Jugador Automático #2
  -Color ( asignado por defecto ): rojo

JUGADOR 3
¿Quieres que sea un jugador controlado por el ordenador? (s/N) █

```

Una vez introducidos los datos de los jugadores, se cargan en el juego las casillas y las tarjetas y se pedirá a cada jugador que lance los dados para establecer el orden de tirada.

```

Jugador #1 pulse 'INTRO' para lanzar su dado:
  -Total lanzado = 4 ( 1 + 3 )
Jugador #2 pulse 'INTRO' para lanzar su dado:
  -Total lanzado = 8 ( 5 + 3 )
Jugador #3 pulse 'INTRO' para lanzar su dado:
  -Total lanzado = 10 ( 4 + 6 )
Jugador #4 pulse 'INTRO' para lanzar su dado:
  -Total lanzado = 5 ( 2 + 3 )
Jugador #5 pulse 'INTRO' para lanzar su dado:
  -Total lanzado = 9 ( 3 + 6 )

>>>> Jugador #3, lanzas en primer lugar.

```

Configuración personalizada

Si se desea cargar una configuración del juego diferente de la establecida por defecto, se deben introducir manualmente las opciones necesarias: fichero para cargar el tablero, ficheros de tarjetas de suerte y caja de comunidad, moneda de juego, dinero inicial de los jugadores, número total de casas y hoteles y si se habilitarán las subastas en los casos en los que los jugadores decidan declinar la compra de una propiedad.

Esta opción sólo se recomienda para usuarios avanzados del juego.

```

>Para obtener ayuda teclee 'ayuda'
street master's> nuevo
¿Desea cargar la configuración por defecto del juego?(recomendado) (s/n)n
Introduzca:
  -Fichero para cargar el tablero: madrid.tab
  -Fichero para cargar el mazo de suerte: suerte.tjt
  -Fichero para cargar el mazo de caja de comunidad: caja_comunidad.tjt
  -Tipo de moneda para jugar: pesetas
  -Dinero inicial: 120000
  -Casas disponibles inicialmente: 34
  -Hoteles disponibles inicialmente: 10
¿Quieres que se subasten las propiedades que no se compran? (s/N)

```

- Cargar una partida existente

La otra opción para iniciar una partida es cargar una ya existente. Para ello, se debe ejecutar el comando 'cargar'. Este comando se puede ejecutar introduciendo como parámetro el nombre de la partida que se quiere cargar o ejecutando el comando y luego indicando el nombre de la partida.

Si no se conoce el nombre de la partida se puede consultar mediante el comando 'lista -g', que mostrará la lista de partidas guardadas. Los archivos guardados se mostrarán con una extensión .sav, pero al ejecutar el comando 'cargar', el nombre de la partida no debe incluir dicha extensión.

```

street master's (Jugador #2)> lista -g
PARTIDAS GUARDADAS:
-----
bancarrota.sav  ej3.sav      ejemplo.sav   hipotecar.sav  partida1.sav  partida6.sav
casas2.sav     ejemplo2.sav ej.sav        ja.sav         partida3.sav  partida9.sav
casas.sav      ejemplo3.sav hip2.sav      jugadorA.sav   partida4.sav  partidaEjemplo.sav
construir.sav  ejemplo4.sav hipotecadas.sav  partida10.sav  partida5.sav  partidaJavi.sav
street master's (Jugador #2)>

```

4.6.3.1.2 Desarrollo de una partida

Una vez cargada la partida se puede empezar a jugar. El funcionamiento del juego es muy sencillo, ya que el ordenador controlará toda la acción del juego para que el jugador sólo se tenga que preocupar de idear sus estrategias e intentar llevarlas a cabo.

La instrucción 'tirar' es el comando básico del juego. A partir de su invocación, el jugador lanzará sus dados haciendo que se mueva su ficha y generando una serie de acciones que variarán en función de la casilla de destino del jugador.

```
Jugador #3:
DADO 1 = 1   DADO 2 = 5   TOTAL = 6

-Casilla de origen
-----
|          azul cielo          |
|-----|
|          GLORIETA CUATRO CAMINOS          |
|-----|
|          Precio 10000          |          6|
|-----|

-Casilla de destino
-----
|          COMPAÑÍA DE ELECTRICIDAD          |
|-----|
|          Precio 15000          |          12|
|-----|
```

Las únicas condiciones para que un jugador pueda lanzar sus dados es que sea el poseedor del turno de juego y no exista ningún jugador con deudas pendientes.

Conocer que jugador tiene el turno de tirada es tan sencillo como observar el prompt del juego. Si se observa, se puede ver como aparece un nombre entre paréntesis. Ese jugador será el poseedor del turno.

```
street master's (Jugador #4)>
```

Existe la posibilidad de que aparezca un nombre entre corchetes además del ya mencionado jugador entre paréntesis. Ese jugador será el que controla en ese momento la consola de juego, es decir, el jugador sobre el que recaen las acciones que se lleven a cabo. Obviamente una de esas acciones no podrá ser lanzar los dados puesto que el turno no le pertenece.

Si en algún momento de la partida, un jugador al que no le corresponda lanzar los dados desea realizar alguna otra acción, deberá indicarlo mediante el comando 'cambiar', que le colocará como el sujeto de las acciones que se realicen hasta que indique que ha concluido mediante el comando 'fin'.

```
street master's (Jugador #4)> cambiar -color amarillo
street master's (Jugador #4) [~Jugador #1]> █
```

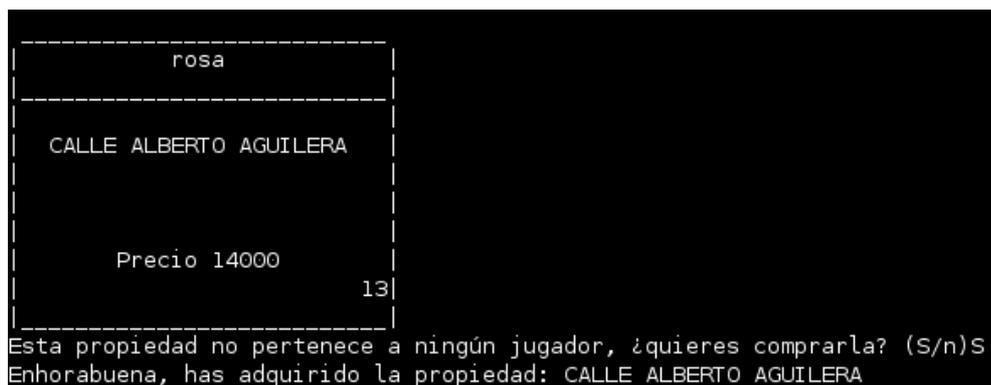
Retomando la funcionalidad que permite a un jugador desplazarse por el tablero, se deben explicar las acciones en las que un jugador se puede ver involucrado en función de su casilla de destino.

- *Casilla de tipo propiedad*

Se entiende por una casilla de tipo propiedad todas las casillas de calles, estaciones y servicios públicos que aparecen en el tablero de juego.

Las acciones en este tipo de casillas son de dos tipos en función de si la casilla tiene dueño o no en el momento de caer sobre ella.

Si la propiedad no tiene dueño, se ofrecerá la posibilidad de comprarla, a lo que habrá que responder sí (S) o no (N). Si se decide comprar la casilla se añadirá a la lista de propiedades del jugador comprador. En caso contrario, la casilla quedará libre.



Si se ha establecido la opción de subastar propiedades en la configuración y se decide no comprar una propiedad, ésta será subastada entre todos los jugadores.

Si la propiedad tiene un dueño cuando el jugador cae sobre ella deberá pagar la cantidad que establezca la tarjeta de propiedad por caer sobre ella. Si se dispone de dinero suficiente, el juego descontará automáticamente la cantidad de dinero del jugador que debe pagar, y se la añadirá al capital del cobrador.

```

-Casilla de destino
-----
          azul
-----
      PASEO DEL PRADO
      (1 casa)
      Precio 40000
          39
-----
Jugador #2 debes pagar a Jugador #1 por caer en su propiedad.
+Total a pagar: 20000

```

Si el jugador no tiene dinero no pagará pero la partida no podrá seguir hasta que pague o se declare en bancarrota. Más adelante se explica como conseguir dinero, pagar y declararse en bancarrota.

- *Casilla de tipo impuesto*

En primer lugar se debe diferenciar entre dos tipos de impuestos: los impuestos normales y los especiales.

Si el jugador cae sobre un impuesto normal, el juego descontará al jugador automáticamente la cantidad de dinero que se estipula en dicha casilla. Si no se dispone de dinero suficiente, el jugador podrá reunir dinero hasta final de turno o declararse en bancarrota.

```

-Casilla de destino
-----
          IMPUESTO DE LUJO
-----
          Pagar 10000
          38
-----
Jugador #3 debes pagar 10000 a la banca a modo de impuesto, por haber caido en la casilla IMPUESTO DE LUJO

```

Si el impuesto es especial, el jugador deberá elegir si prefiere pagar una cantidad de dinero fija o una cantidad calculada a partir de un tanto por ciento de su capital total. Una vez que el jugador elija una de las opciones, el proceso a seguir es el mismo que el comentado anteriormente.

jugador de la cárcel. Las tarjetas de este último tipo, se podrán utilizar para salir de la cárcel de la forma que se explica más adelante.

- *La cárcel*

La casilla de cárcel como tal, no implica ninguna acción sobre el jugador que caiga en ella tras lanzar los dados. Sin embargo, cuando un jugador caiga sobre la casilla “vaya a la cárcel”, se enviará al jugador a la cárcel, lo que implica estar inmovilizado durante un cierto número de turnos.



El jugador no podrá hacer nada para evitarlo hasta que le vuelva a llegar el turno de lanzar los dados, cuando se le preguntará si desea salir de la cárcel usando una tarjeta (si la tiene) o pagando una multa. La respuesta en ambos casos sólo puede ser ‘sí’ o ‘no’. En caso afirmativo saldrá inmediatamente de la cárcel y lanzará los dados de la forma habitual para proseguir con el juego.

```
Jugador #2:
Puedes usar la tarjeta para salir de la cárcel de la que dispones.
¿Quieres hacerlo? (s,N)Si lo deseas puedes pagar una multa de 5000 para salir de la cárcel.
¿Deseas pagar la multa? (s/N)
```

Al cumplir el periodo máximo de turnos en la cárcel (el número dependerá de la configuración del tablero), el jugador saldrá directamente de la cárcel.

- *Casilla de tipo salida*

Cuando el jugador pase por la casilla de **salida**, se pagará automáticamente el sueldo establecido al jugador.

```

-Casilla de destino
-----
          SALIDA
          Cobre 20000
          cada vez que
          pase por aquí.
          =====> 0
-----
Jugador #3 cobras 20000 por pasar por la casilla de salida.

```

4.6.3.1.3 Bancarrota

Cuando un jugador debe pagar una cantidad de dinero y no dispone de ella se dice que esta en bancarrota o quiebra. En esta situación, deberá tratar de conseguir dinero antes de finalizar el turno actual, ya que si no lo consigue será eliminado de la partida actual.

Street Master's ofrece tres alternativas para aumentar el capital en efectivo de los jugadores: *venta de casas construidas*, *hipoteca de propiedades* y *negociaciones con otros jugadores*. Todos ellos serán explicados detenidamente más adelante.

Cuando se consiga el dinero suficiente para saldar sus deudas debe emplear el comando 'pagar' para que el acreedor cobre su deuda.

Si no ha sido posible conseguir dinero suficiente para cubrir la deuda, el jugador debe retirarse de la partida usando el comando 'bancarrota'.

Una vez eliminado el jugador se eliminarán todas las casas de sus calles y sus propiedades pasarán a pertenecer a su acreedor.

Si el acreedor no es un único jugador (es la banca o varios jugadores), las propiedades se deberán subastar entre el resto de contrincantes.

Si el acreedor es un único jugador, las propiedades pasarán directamente a pertenecerle.

4.6.3.1.4 Gestión de hipotecas

Las hipotecas son una funcionalidad muy socorrida para conseguir dinero. Si un jugador lo necesita podrá **hipotecar** una o varias de sus propiedades que cumplan con las normas del juego.

Para conocer qué propiedades de un jugador cumplen con los requisitos necesarios para ser hipotecados se debe utilizar el comando 'hipotecar'. Al hacerlo, se mostrará una lista con las propiedades hipotecables acompañadas de los datos básicos para elegir la propiedad que más convenga.

```
street master's (Jugador #3)> hipotecar
(11) GLORIETA DE BILBAO           Hipoteca = 7000
(15) ESTACIÓN DE LAS DELICIAS     Hipoteca = 10000
(18) CALLE VELÁZQUEZ             Hipoteca = 9000
(24) CALLE CEA BERMUDEZ         Hipoteca = 12000
(25) ESTACIÓN DEL MEDIODÍA       Hipoteca = 10000
(27) CALLE BAILÉN                Hipoteca = 13000
```

A continuación, si lo desea podrá hipotecar cualquiera de las propiedades mostradas con el comando anterior volviendo a utilizar la instrucción 'hipotecar' acompañada del número de la propiedad que se desea hipotecar. Por ejemplo, si se desea hipotecar la "Calle Velázquez", que ocupa la casilla número 18, se deberá ejecutar "hipotecar 18" en el tablero proporcionado por Street Master's.

```
street master's (Jugador #3)> hipotecar 18
CALLE VELÁZQUEZ ha sido hipotacada
```

En cualquier momento, un jugador podrá **deshipotecar** sus propiedades, siempre y cuando tenga dinero suficiente para hacerlo. El comando para hacerlo es 'deshipotecar'.

Igual que ocurre con las hipotecas, se podrán consultar todas las propiedades hipotecadas (y por lo tanto deshipotecables) mediante el comando 'deshipotecar'.

```
street master's (Jugador #3)> deshipotecar
(18) CALLE VELÁZQUEZ           Levantar hipoteca = 9900
(25) ESTACIÓN DEL MEDIODÍA     Levantar hipoteca = 11000
```

Para deshipotecar una propiedad concreta se deberá acompañar al comando 'deshipotecar' con el número de propiedad. Por ejemplo para deshipotecar la Calle Velázquez, el comando deshipotecar sería "deshipotecar 18" en el tablero cargado por defecto.

```
street master's (Jugador #3)> deshipotecar 18
CALLE VELÁZQUEZ ha sido deshipotecada
```

Si un jugador al que no le corresponde lanzar los dados desea realizar alguna de estas acciones puede hacerlo usando el comando 'cambiar' para obtener el control del juego.

4.6.3.1.5 Gestión de edificios

Los edificios (casas y hoteles) son una característica propia de las calles, por lo que esta explicación se centra exclusivamente en este tipo de propiedades.

Una calle es edificable cuando un jugador posee todas las calles de su mismo color y además no haya ninguna calle de dicho grupo que tenga menos casas que ella. Del mismo modo, no se pueden vender edificios de una calle del grupo, si otra calle del mismo grupo tiene más edificios construidos. Conociendo esta norma básica del juego, se puede comenzar a explicar cómo construir y vender casas y hoteles en Street Master's.

Para conocer sobre qué calles puede **construir** un jugador, éste debe usar el comando 'construir', que mostrará una lista con todas las calles que pueden ser construidas acompañada de información relevante de cada una de ellas.

```
street master's (Jugador #5) [~Jugador #3]> construir
(16) AVENIDA FELIPE II           ACTUALMENTE: 2 casas
(18) CALLE VELÁZQUEZ           ACTUALMENTE: 2 casas
(19) CALLE SERRANO             ACTUALMENTE: 2 casas
```

Una vez conocidas las calles edificables, el jugador podrá decidir construir cualquiera de ellas. Para hacerlo, sólo tendrá que ejecutar el comando 'construir' acompañado del número de casilla que representa la calle que se desee construir. No se debe indicar si se quiere construir una casa o un hotel porque el sistema lo detectará automáticamente.

```
street master's (Jugador #5) [~Jugador #3]> construir 16
Enhorabuena, su nueva casa ha costado 10000
AVENIDA FELIPE II ha sido construida.
```

Para construir varias casas, se ejecutará el comando tantas veces como sea necesario.

Las ventas de las casas se realizan de la misma forma que las construcciones. Para ello se usará el comando 'vender', sin parámetros para conocer las calles que tienen casas construidas y se pueden vender, y acompañado del número de casilla para confirmar una venta de una casa.

```
street master's (Jugador #5) [~Jugador #3]> vender
(16) AVENIDA FELIPE II          ACTUALMENTE: 3 casas
(18) CALLE VELÁZQUEZ           ACTUALMENTE: 3 casas
(19) CALLE SERRANO             ACTUALMENTE: 3 casas
```

En el ejemplo que se muestra en la imagen, se podría vender una casa de la Calle Velázquez, introduciendo el comando 'vender 18'.

Un dato a tener en cuenta, tanto para las construcciones como las ventas de edificios, es conocer de cuántas casas y hoteles dispone la banca porque una vez que se terminen no se podrán comprar más. Para ello se debe usar el comando 'ver -banca', que mostrará cuantos edificios tiene la banca y de que tipo son.

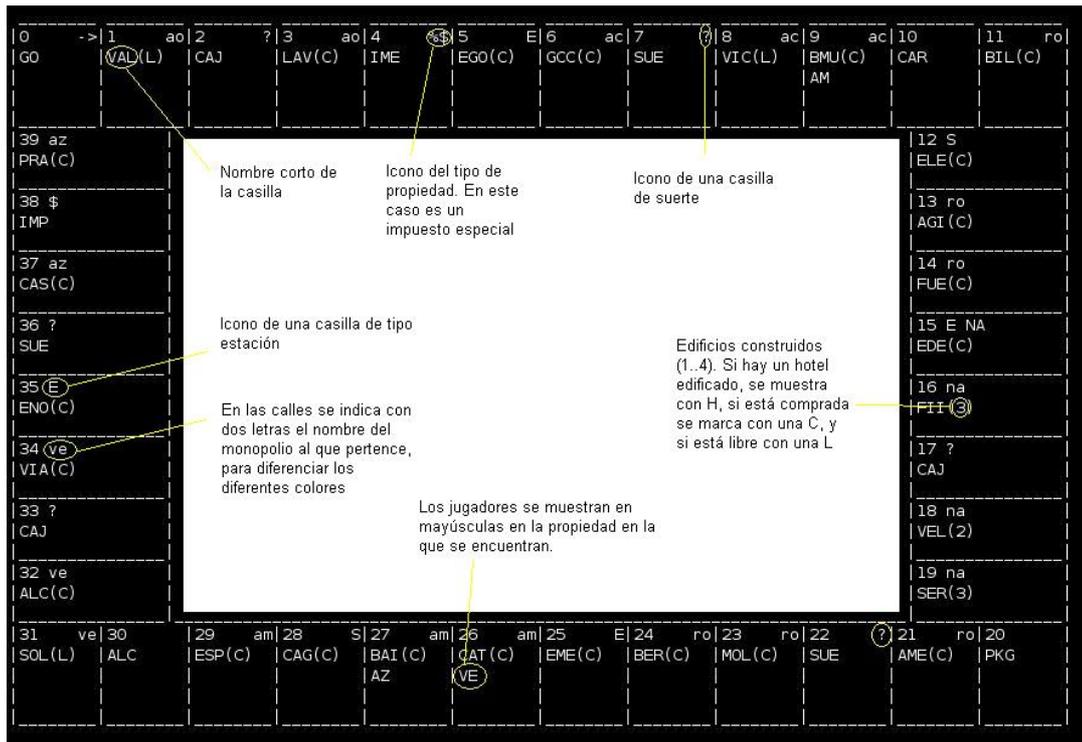
4.6.3.1.6 Ver el estado de la partida

No disponer de un interfaz gráfico no supone no poder ver el tablero ni poder hacer un seguimiento pormenorizado de la partida. Para poder ver el estado de cualquier elemento que participe en una partida de Street Master's, se utilizará el comando 'ver'.

Este comando podrá ser ejecutado en cualquier momento y el resultado será el mismo independientemente del jugador que lo ejecute. Mediante esta instrucción se puede obtener información sobre el tablero de juego, las casillas, los jugadores e incluso la banca. A continuación se explica como utilizar todas las opciones que ofrece este comando.

- Consultar el tablero

Para consultar el tablero de juego se debe ejecutar el comando 'ver' (sin ningún parámetro opcional) y se mostrará el tablero de juego con toda la información necesaria para situar a cada jugador en la partida: estado de las casillas y posiciones de los jugadores.



El espacio es muy reducido por lo que se ha seguido una representación muy concisa. Existe un comando para poder comprender sin problemas la notación utilizada. Dicho comando ofrece la posibilidad de consultar el tablero en modo texto, mediante una tabla en la que se muestra la información de todas las casillas. Si se prefiere ejecutar esta opción, se debe ejecutar el comando 'ver -leyenda'.

Num.	Nombre corto	Tipo*	Nombre	Estado	Num.	Nombre corto	Tipo*	Nombre	Estado
0	GO	->	SALIDA	-	20	PKG	T	PARQUE GRATUITO	L
1	VAL	ao	RONDA DE VALENCIA	4	21	AME	ro	AVENIDA DE AMÉRICA	L
2	CAJ	?	CAJA DE COMUNIDAD	-	22	SUE	?	SUERTE	-
3	LAV	ao	PLAZA LAVAPIÉS	3	23	MOL	ro	CALLE MARÍA DE MOLINA	C
4	IME	\$\$	IMPUESTO SOBRE CAPITAL	-	24	BER	ro	CALLE CEA BERMUDEZ	C
5	EGO	E	ESTACIÓN DE GOYA	L	25	EME	E	ESTACIÓN DEL MEDIODÍA	C
6	GCC	ac	GLORIETA CUATRO CAMINOS	C	26	CAT	am	AVENIDA DE LOS REYES CATÓLICOS	C
7	SUE	?	SUERTE	-	27	BAI	am	CALLE BAILÉN	L
8	VIC	ac	AVENIDA REINA VICTORIA	C	28	CAG	S	COMPAÑÍA DE AGUAS	C
9	BMU	ac	CALLE BRAVO MURILLO	L	29	ESP	am	PLAZA DE ESPAÑA	L
10	CAR	T	CÁRCEL	L	30	ALC	T	VAYA A LA CÁRCEL	L
11	BIL	ro	GLORIETA DE BILBAO	L	31	SOL	ve	PUERTA DEL SOL	C
12	ELE	S	COMPAÑÍA DE ELECTRICIDAD	L	32	ALC	ve	CALLE ALCALÁ	L
13	AGI	ro	CALLE ALBERTO AGUILERA	C	33	CAJ	?	CAJA DE COMUNIDAD	-
14	FUE	ro	CALLE FUENCARRAL	C	34	VIA	ve	GRAN VÍA	L
15	EDE	E	ESTACIÓN DE LAS DELICIAS	L	35	ENO	E	ESTACIÓN DEL NORTE	C
16	FII	na	AVENIDA FELIPE II	C	36	SUE	?	SUERTE	-
17	CAJ	?	CAJA DE COMUNIDAD	-	37	CAS	az	PASEO DE LA CASTELLANA	C
18	VEL	na	CALLE VELÁZQUEZ	C	38	IMP	\$	IMPUESTO DE LUJO	-
19	SER	na	CALLE SERRANO	L	39	PRA	az	PASEO DEL PRADO	C

(*) El tipo indica de que tipo de casilla se trata:
 \$\$ Impuesto con la posibilidad de pagar un porcentaje sobre el capital
 \$ Impuesto
 Az Indica el color del monopolio al que pertenece la calle
 E Estación
 S Servicio
 -> Casilla de salida. Indica la dirección del juego

Además en el tablero se puede ver el estado de cada propiedad
 (L) Propiedad libre, sin dueño
 (1)..(4) Indica el número de casas construidas en una calle
 (H) Indica que se ha construido un hotel
 (C) Indica que la propiedad posee un dueño

- *Consultar las casillas*

Cada casilla del tablero tiene información muy específica que merece ser mostrada independientemente. Para consultar una casilla concreta se debe ejecutar el comando 'ver -casilla' acompañado del número de casilla que se desea consultar. Por ejemplo, para ver la casilla número 1, se deberá ejecutar "ver -casilla 1".

```

RONDA DE VALENCIA

Alquiler base = 200
Alquiler 1 casa = 1000
Alquiler 2 casas = 3000
Alquiler 3 casas = 9000
Alquiler 4 casas = 16000
Alquiler hotel = 25000
  Precio = 6000
  Hipoteca = 3000
  Precio por casa = 5000
  Precio por hotel = 5000

PROPIETARIO: Jugador #1 (amarillo)
EDIFICACIONES: 4 casa(s)

1
azul oscuro
    
```

Esta opción es especialmente útil para conocer las posibilidades que ofrece una propiedad.

- *Consultar los jugadores*

Street Master's también permite consultar el estado de cada uno de los jugadores obteniendo información sobre sus propiedades y su capital en todo momento.

Si se quiere obtener un resumen de la información de cada jugador, en el que se muestra sus nombres, colores de las fichas, dinero del que disponen y su estado económico actual (negativo si tienen deudas y positivo si son acreedores de un moroso), se debe ejecutar el comando 'ver -jugador'.

TURNO	NOMBRE	COLOR	DINERO	CAPITAL	SALDO
0	Jugador #1	amarillo	206600	321600	0
1	Jugador #2	rojo	166600	287600	0
2	Jugador #3	azul	196800	269800	0
3	Jugador #4	verde	146800	282800	0

Si la información que se precisa es más amplia, se puede consultar cada uno de los jugadores mediante el mismo comando acompañado del color de

su ficha. Para ver la información del jugador que tiene asignada la ficha amarilla, se debe ejecutar 'ver -jugador amarillo'.

```

-NOMBRE:      Jugador #1
-FICHA (color): amarillo
-CAPITAL:     295000
+DINERO EFECTIVO: 125000
+CAPITAL INVERTIDO: 170000
PROPIEDADES (9) :
(1) RONDA DE VALENCIA      GRUPO: azul oscuro  CONSTRUIDO: 4 casas
(3) PLAZA LAVAPIÉS        GRUPO: azul oscuro  CONSTRUIDO: 3 casas
(8) AVENIDA REINA VICTORIA GRUPO: azul cielo   CONSTRUIDO: (No tiene permiso para construir)
(13) CALLE ALBERTO AGUILERA GRUPO: rosa         CONSTRUIDO: (No tiene permiso para construir)
(18) CALLE VELÁZQUEZ      GRUPO: naranja     CONSTRUIDO: (No tiene permiso para construir)
(23) CALLE MARIA DE MOLINA GRUPO: rojo         CONSTRUIDO: (No tiene permiso para construir)
(24) CALLE CEA BERMUDEZ   GRUPO: rojo         CONSTRUIDO: (No tiene permiso para construir)
(28) COMPAÑÍA DE AGUAS    [Servicio]
(35) ESTACIÓN DEL NORTE   [Estacion]

```

En este caso se muestran, además de los datos económicos y personales, todas las posesiones del mismo (propiedades y tarjetas que le liberan de la cárcel).

- Consultar la banca

Como ya se ha comentado cuando se explicó la compra y venta de casas, se puede consultar el número de edificios disponibles mediante el comando 'ver -banca'.

```

Número de casas disponibles 25
Número de hoteles disponibles 12

```

No se muestra la cantidad de dinero disponible porque como se advirtió en la sección 4.2.5, página 66, se considera que es infinito.

4.6.3.1.7 Subastas

El proceso de subastas sigue una secuencia en la que cada jugador deberá pujar en orden por la propiedad que se ofrece. Un jugador no está obligado a pujar, pudiendo pasar o incluso salir de la puja.

Dentro de las subastas se deben conocer tres normas básicas:

- 1) Si un jugador quiere pasar en un turno, debe indicarlo mediante la palabra 'paso'.
- 2) Si un jugador quiere abandonar la puja, lo indicará con la palabra 'salir'. Si hubiera más objetos a subastar, el jugador podrá pujar a partir del siguiente objeto en liza.
- 3) La propiedad la ganará el jugador que haga la puja más alta cuando finalice la subasta. La subasta terminará cuando el turno

vuelva al jugador que ostenta la puja más elevada hasta el momento. Esto significará que el resto de jugadores han pasado, han ofrecido una cantidad de dinero inferior o han abandonado la puja.

```

OBJETO PARA PUJAR: CALLE SERRANO
-----
CALLE SERRANO

Alquiler base = 1600
Alquiler 1 casa = 8000
Alquiler 2 casas = 22000
Alquiler 3 casas = 60000
Alquiler 4 casas = 80000
Alquiler hotel = 100000
Precio = 20000
Hipoteca = 10000
Precio por casa = 10000
Precio por hotel = 10000

naranja 19

Puja Jugador #1 ( máximo 245700 ) : 230
Puja Jugador #3 ( máximo 166400 ) : 420
Puja Jugador #4 ( máximo 200900 ) : 5000
Puja Jugador #5 ( máximo 102700 ) : 12000
Puja Jugador #1 ( máximo 245700 ) : paso
Puja Jugador #3 ( máximo 166400 ) : salir
Jugador #3, has abandonado la subasta.
Puja Jugador #4 ( máximo 200900 ) : █

```

Las propiedades ganadas por cada jugador se unirán a las suyas en el momento en que finalice la puja. Si no hay ningún ganador la propiedad será de la banca.

4.6.3.1.8 Negociaciones

Esta funcionalidad sólo estará activa cuando no participen jugadores automáticos en la partida.

Para iniciar una negociación, el jugador que desee realizar una oferta deberá ejecutar el comando 'negociar'. El programa guiará al usuario durante el resto del proceso.

En primer lugar, se debe elegir el jugador al que se desea enviar la oferta seleccionándolo entre los que aparecen en la lista que se muestra en pantalla.

A continuación, se piden los datos de las propiedades que ofrece el jugador al emisor elegida:

- Dinero efectivo: Cantidad de dinero que se ofrecerá.

- **Propiedades:** Se preguntará al jugador si quiere añadir alguna propiedad, en cuyo caso deberá seleccionar de la lista que se muestre las propiedades que desee incluir en la oferta.
- **Tarjetas:** Si el jugador posee alguna tarjeta para salir de la cárcel las podrá incluir en la operación, respondiendo si (S) cuando se le pregunte si desea hacerlo.

```

street master's (Jugador #2)> negociar
Ejige el jugador al que deseas hacer la oferta
(0) Jugador #1 [amarillo]
(2) Jugador #3 [azul]
(3) Jugador #4 [verde]
(C) Cancelar
Elige: 2
Se inicia la negociacion entre Jugador #2 y Jugador #3
Jugador #2 indica lo que ofreces
Jugador #2 indica la cantidad de dinero que quieres ofrecer. Si no quieres ofrecer dinero efectivo, introduce 0.
Dinero en efectivo: 125000
¿Deseas incluir alguna propiedad en la operacion? (S/n)s
LISTADO DE PROPIEDADES:
(6) GLORIETA CUATRO CAMINOS          GRUPO: azul cielo    CONSTRUIDO: (No tiene permiso para construir)
(14) CALLE FUENCARRAL                GRUPO: rosa         CONSTRUIDO: (No tiene permiso para construir)
(25) ESTACIÓN DEL MEDIODÍA           [Estacion]
(28) COMPAÑÍA DE AGUAS                [Servicio]
(35) ESTACIÓN DEL NORTE               [Estacion]
(39) PASEO DEL PRADO                  GRUPO: azul         CONSTRUIDO: (No tiene permiso para construir)
(C) Cancelar
Indica el número de propiedad que quieres incluir.
Elige: 6
Propiedad incluida
Propiedades incluidas:
GLORIETA CUATRO CAMINOS
¿Deseas incluir alguna propiedad más en la operacion? (S/n)n
¿Deseas incluir alguna tarjeta para salir de la cárcel en la operacion? (S/n)s

```

Una vez establecidos las propiedades que se ofrecerán, se tienen que configurar las propiedades que se pedirán a cambio. El proceso a seguir es igual que el anterior con los datos del jugador que recibirá la oferta.

Una vez que se ha terminado de definir la oferta, ésta se envía al jugador receptor de la misma, al cual se le pregunta si acepta la oferta. En caso afirmativo, se realiza el intercambio de propiedades.

4.6.3.1.9 *Guardar partida*

Para guardar una partida se debe invocar al comando 'guardar' seguido del nombre de la partida. Si la partida ya existe con este nombre, se sobrescribirá la información guardada.

4.6.3.1.10 *Fin del juego*

El juego terminará cuando sólo quede un jugador en la partida, siendo éste el ganador.

Si se desea abandonar la partida y dejarla a medias se puede hacer con el comando 'salir' que sale del juego completamente o con el comando 'cerrar' que sólo cierra la partida actual.

4.6.3.1.11 Ayuda

Street Master's cuenta con una ayuda disponible en todo momento durante el juego. Se podrán realizar consultas rápidas sobre cada uno de los comandos del juego o una ayuda más detallada mediante pequeño tutorial, que está desarrollado tanto en HTML como en texto plano.

4.6.3.1.11.1 Comando ayuda

Ejecutando el comando *ayuda* se obtiene una información rápida y precisa sobre la utilidad de cada uno de los comandos del juego, en la que se indica su funcionalidad y su modo de uso.

Para conocer conceptos generales sobre la ayuda, y cómo utilizar este comando se debe introducir el comando ayuda. Se mostrarán todos los comandos disponibles en Street Master's.

Para acceder a la ayuda concreta de uno de los comandos de Street Master's, se debe indicar como parámetro el nombre del comando que se quiere consultar. Por ejemplo, para obtener información sobre el comando *nuevo*, se debe introducir '*ayuda nuevo*'.

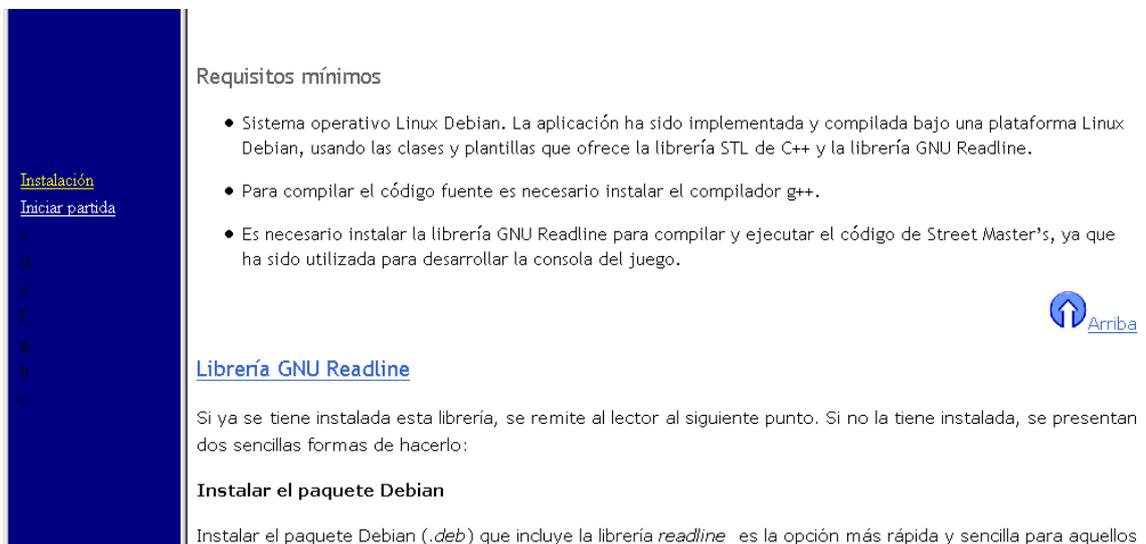
```
NUEVO
-----
NOMBRE
    nuevo - Crea una nueva partida.
SINOPSIS
    nuevo
DESCRIPCIÓN
    El comando 'nuevo' creará una nueva partida de Street Master's. Al ejecutar este comando se dará la opción de
    cargar una configuración por defecto, o por contra, crear una partida siguiendo una configuración personal.
    El autor recomienda encarecidamente usar la configuración por defecto ya que de otra forma no se podrá
    garantizar que los ficheros de casillas y tarjetas cumplan con el estándar establecido. En cualquier caso, si
    se desea cargar una partida personalizada, deberá conocer los ficheros de casillas, suerte y caja de comunidad,
    ya que serán requeridos durante la carga.
    Por último se pedirán los datos de los jugadores. Los jugadores podrán ser controlados por el ordenador, para
    lo cual se deberá indicar cuando el programa lo pregunte. En el caso de que el jugador sea manejado por la
    máquina, ésta controlará todos los comandos del juego, exceptuando el comando 'tirar', que se ejecutará
    manualmente para poder realizar un seguimiento correcto del juego.
MAS
    cargar - Carga una partida ya existente.
street master's> █
```

4.6.3.1.11.2 Tutorial

Para obtener una información más detallada sobre el juego y cómo jugar se debe consultar el tutorial creado con dicho fin. Para poder verlo en formato HTML es necesario configurar la variable de entorno MONOPOLY_BROWSER con el nombre del navegador preferido.

Para ejecutar esta ayuda, se debe introducir el comando *minicomo*.

Si se ha configurado la variable de entorno, se abrirá una ventana con la información en HTML. En caso contrario, se mostrará un fichero de texto plano paginado, por el cual se podrá navegar con los comandos *anterior* y *siguiente*.



Requisitos mínimos

- Sistema operativo Linux Debian. La aplicación ha sido implementada y compilada bajo una plataforma Linux Debian, usando las clases y plantillas que ofrece la librería STL de C++ y la librería GNU Readline.
- Para compilar el código fuente es necesario instalar el compilador g++.
- Es necesario instalar la librería GNU Readline para compilar y ejecutar el código de Street Master's, ya que ha sido utilizada para desarrollar la consola del juego.

[Librería GNU Readline](#)

Si ya se tiene instalada esta librería, se remite al lector al siguiente punto. Si no la tiene instalada, se presentan dos sencillas formas de hacerlo:

Instalar el paquete Debian

Instalar el paquete Debian (.deb) que incluye la librería *readline* es la opción más rápida y sencilla para aquellos

4.6.3.2 Jugar contra la máquina

Si se ha seleccionado uno o varios jugadores controlados por el ordenador se deben tener en cuenta ciertas consideraciones.

Todos las acciones del jugador automático serán controladas por el ordenador excepto el comando 'tirar', que debe ser ejecutado por un jugador humano. De esta forma se podrá realizar un seguimiento de las acciones del ordenador.

Como si se tratara de un jugador más, se puede realizar cualquier tipo de acción antes y después de que lance el jugador automático, refiriéndose esto a las hipotecas, construcciones, ventas y demás acciones que no requieran poseer el turno del juego.

Además, es importante dejar claro que no se puede negociar cuando un jugador controlado por el ordenador interviene en la partida.

El resto del juego es idéntico al juego entre personas.

4.6.3.3 Personalizar el juego

Una de las características del juego, es la facilidad para personalizar el tablero y las tarjetas siguiendo una sencilla plantilla.

El **fichero de casillas** o lo que es lo mismo, el tablero de juego, debe distinguir entre los diferentes tipos de casillas que se encuentran en el sistema siguiendo el siguiente modelo.

Tipo de casilla (id)	Representación
Calle	calle número nombre nombre corto color precio hipoteca alquiler base alquiler 1 casa 2 casas 3 casas 4_casas hotel precio casa precio hotel
Servicio	servicio número nombre nombre corto precio hipoteca multiplicador para 1 servicio multiplicador para 2 servicios
Estación	estacion número nombre nombre corto precio hipoteca alquiler con 1 estación 2 estaciones 3 estaciones 4 estaciones
Parking	parking número nombre nombre corto
Impuesto	impuesto número nombre nombre corto precio
Impuesto especial	impuesto_especial número nombre nombre corto tanto por ciento precio fijo
Tarjeta	tarjeta número nombre nombre corto mazo asociado
Salida	salida número nombre nombre corto sueldo
Cárcel	carcel número nombre nombre corto máximo de turnos multa
A la cárcel	a_la_carcel número nombre nombre corto

Como se puede ver, simplemente consiste en indicar los datos de cada casilla siguiendo el formato dado. Un ejemplo de tablero es el que se proporciona en el juego en el fichero *madrid.tab*.

Existen dos **ficheros de tarjetas**, ya que existen dos mazos en el juego. No obstante se podría cargar el mismo fichero para ambos si así se quisiera.

Ambos ficheros tienen la misma configuración, que por necesidades computacionales siguen una estructura diferente a la que se ha mostrado para el caso de las casillas.

Cada una de las tarjetas debe respetar la siguiente estructura:

```
[Texto descriptivo]
numero= 0
{ACCION
    (acciones básicas)+ }+
;
```

Siendo cada una de las acciones básicas uno de los siguientes parámetros:

Acción básica	Descripción	Valores posibles
sgtServicio	Se mueve al jugador a la siguiente casilla de servicios.	{0,1}
encarcelar	El jugador es encarcelado.	{0,1}
avanzarA	El jugador se mueve a la casilla que se indique.	0..número de casillas
avanzar	El jugador se moverá el número de casillas que se indiquen.	Número positivo: Avanzar Número negativo: Retroceder
pagar	El jugador pagará la cantidad indicada a la banca.	Número positivo: Pagar Número negativo: Cobrar
sgtEstacion	El jugador avanza a la siguiente estación.	{0,1}
tarjetaCarcel	Tarjeta para salir libre de la cárcel.	{0,1}
pagarPorCasa	Pagar la cantidad indicada por cada casa construida.	Número positivo
pagarPorHotel	Pagar la cantidad indicada por cada hotel construido.	Número positivo
pagarATodos	Pagar la cantidad indicada al resto de los jugadores.	Número positivo: Pagar Número negativo: Cobrar
sgtPropiedad	El jugador avanza a la siguiente calle.	{0,1}
multiPago	Se combina con acciones de movimiento del jugador (avanzar o avanzarA). El alquiler de la casilla destino se multiplicará por esta cantidad.	Número positivo
multiPagoDado	Funcionalidad igual que la anterior, pero la cantidad por la que multiplica se	{0,1}

	obtendrá de lanzar los dados.	
cobrarSalida	Asociado a las acciones de movimiento, se indica si se cobra si se pasa por la casilla de salida.	{0,1}

Se podrá crear cualquier tarjeta que se pueda representar como una combinación de los parámetros anteriores. Para ver un ejemplo, se pueden ver los ficheros *suerte.tjt* y *caja.tjt* que se incluyen en el juego.

Si se quiere cargar cualquiera de los tableros o tarjetas personalizadas, se deberá indicar al iniciar el juego que no se cargue la configuración por defecto y seguir los pasos que se indican. (Para más información se remite al lector a leer como se abre una partida nueva)

4.6.3.4 Guía rápida

A continuación se muestran todos los comandos del juego acompañados de una pequeña descripción.

COMANDO	OPCIONES	DESCRIPCIÓN
ayuda	-	Mostrar la ayuda genérica.
	<comando>	Mostrar la ayuda específica del comando indicado.
bancarrotar	-	Eliminar al jugador de la partida.
cambiar	-color <color>	Cambiar el control del juego al jugador cuyo color de ficha se corresponda con el indicado.
	-nombre <nombre>	Cambiar el control del juego al jugador cuyo nombre se corresponda con el indicado.
cargar	-	Permite introducir un nombre de partida para cargarla
	<nombre>	Cargar la partida indicada como argumento.
cerrar	-	Cerrar la partida actual
construir	-	Mostrar una lista de las calles edificables.
	<número de calle>	Construir un edificio en la calle indicada.
deshipotecar	-	Mostrar todas las propiedades hipotecadas.
	<número de propiedad>	Deshipotecar la propiedad indicada.
fin	-	Devolver el control del juego al jugador que posee el turno

guardar	-	Permite introducir un nombre de partida para guardarla
	<nombre>	Guardar la partida con el nombre introducido como argumento.
hipotecar	-	Mostrar todas las propiedades hipotecables.
	<número de propiedad>	Hipotecar la propiedad indicada.
lista	-	Mostrar todos los archivos del juego.
	-g	Mostrar las partidas guardadas.
negociar	-	Iniciar el proceso de negociación.
nuevo	-	Crear una partida nueva.
pagar	-	Pagar las deudas contraídas.
salir	-	Salir del juego.
tirar	-	Lanzar los dados y mover al jugador.
vender	-	Mostrar una lista de las calles edificadas y con edificios que se pueden vender.
	<número de calle>	Eliminar un edificio de la calle indicada.
ver	-	Mostrar el tablero.
	-banca	Mostrar la información sobre edificios disponibles.
	-casilla <numero>	Mostrar la casilla identificada por el número de casilla indicado.
	-jugador	Mostrar la información resumida de todos los jugadores.
	-jugador <color>	Mostrar la información del jugador que se corresponde con el color indicado.

4.7 Conclusiones

En este capítulo de la memoria del proyecto Street Master's se ha mostrado como se ha realizado la aplicación cubriendo todos y cada uno de los objetivos marcados en el capítulo anterior.

La funcionalidad cubierta por la aplicación creada es muy amplia ofreciendo posibilidades muy características que no son ofrecidas por otros juegos comerciales que buscan objetivos familiares. Se puede comprobar haciendo un repaso a los juegos comentados en el Estado de la cuestión (página 10) como algunos de ellos no ofrecen un cliente de IA y otros no incluyen entre sus opciones las subastas o las negociaciones debido a su complejidad – por ejemplo *Moneylandia 2.00*.

La representación de la información del juego se considera muy buena, ya que cubre todas las necesidades del mismo mediante una estructura muy completa basada en el paradigma de la orientación a objetos. Además, se considera sencillo de entender ya que las jerarquías y los polimorfismos facilitan las asociaciones entre las clases.

El juego desarrollado se ha creado mediante un interfaz de texto muy elaborado, concediendo una imagen más amigable al juego. Sin embargo, lo más representativo de la implementación del juego es la gran cantidad de comandos y opciones que ofrece el mismo, que facilitan la interacción del jugador con el juego.

Capítulo 5

Resultados

5 Resultados

En las pruebas realizadas al sistema propuesto en este trabajo se persiguen tres objetivos principales: evaluar la ontología de Street Master's, así como evaluar la eficacia y eficiencia del cliente de IA basado en reglas.

La aplicación ha sido probada bajo un sistema operativo Linux Debian instalado sobre una máquina con un microprocesador de 1.60 Ghz y 256 MB de RAM.

5.1 Ontología

La ontología creada cubre todos los requisitos que se han identificado durante la fase de análisis, por lo que se considera una buena representación del juego. Además, ha quedado demostrado que un cliente de IA puede usar dicha ontología para jugar a esta versión del Monopoly contra otros clientes o jugadores humanos.

5.2 Evaluación de la eficacia del cliente de IA basado en reglas

Para evaluar el cliente de IA basado en reglas se han preparado diferentes partidas para que dicho jugador compita contra el agente desarrollado con un comportamiento aleatorio y contra jugadores humanos. Los resultados obtenidos son muy satisfactorios, tal y como se muestra en las siguientes secciones.

Todas las referencias a reglas que se hacen a lo largo de la explicación de resultados se refieren a las que se exponen en el capítulo dedicado a la discusión de las mismas (página 68).

5.2.1 Competición entre varios agentes de IA

En primer lugar, se discuten los resultados obtenidos al enfrentar a los dos agentes de IA programados en el juego.

Las partidas organizadas se han disputado entre 4 jugadores automáticos (dos de cada tipo) para dar un mayor dinamismo al juego y facilitar la reproducibilidad de los experimentos.

Es importante aclarar que se han filtrado los resultados que aquí se van a mostrar, ya que en ocasiones las partidas pueden llegar a ser muy largas

dado que puede suceder que ningún jugador consiga adquirir el número de propiedades necesarias para poder edificar y por ello, la cantidad que un jugador gasta por término medio a lo largo de una vuelta al tablero de Street Master's es menor que la que cobrará al concluir cada una de ellas. Esto da como resultado un capital continuamente en crecimiento para todos los jugadores. Por ello, estas partidas han sido descartadas de los resultados.

5.2.1.1 Resultados de una partida completa

Se ha considerado un primer caso que consiste en jugar una partida completa de Street Master's con la configuración por defecto del juego²¹ (250.000 unidades de saldo inicial y tablero y tarjetas proporcionados en el fichero por defecto del juego).

Las partidas completas no permiten observar con claridad la competitividad de los clientes de IA, ya que resulta muy complicado que los jugadores completen monopolios y puedan gestionar edificaciones. Esto se debe a la dificultad que supone que un jugador caiga en todas las casillas de un mismo grupo de color antes que lo haga otro en, al menos, una de ellas y la compre. Además, al no completarse dichos monopolios, los precios de alquiler quedan muy reducidos y las decisiones a tomar para obtener más capital para afrontar deudas son triviales.

Sin embargo, se ha podido observar que los agentes basados en reglas de comportamiento no aleatorio tratan de conseguir monopolios completos siempre que caen sobre una propiedad. Este comportamiento agresivo se debe a la regla 2.1.2 por la que siempre que tiene dinero y le queda por comprar una única casilla del monopolio la compra y las reglas 2.1.3 y 2.1.4 que hacen que el agente siempre compre cuando exista una posibilidad, aunque sea lejana, de completar el monopolio.

El comportamiento del cliente basado en reglas no sólo trata de completar sus propios monopolios, si no que además, trata de evitar que otros jugadores completen los suyos. Por ello, si a la dificultad probabilística de caer en todas las casillas de un grupo de color se une que los jugadores basados en reglas tratan de impedir que otros jugadores consigan

²¹ Véase el tablero proporcionado por defecto en el anexo A de esta memoria (página 148).

completarlo, tenemos como resultado un gran número de partidas en las que no se puede construir. La regla 2.1.6 se ha pensado precisamente para provocar esta situación, en la que los jugadores basados en reglas tratan de ser los únicos que completen monopolios.

Se ha comprobado empíricamente que este tipo de partidas son ganadas por el jugador que consigue completar un monopolio completo. Normalmente, sólo consigue completarse un único monopolio y es complicado que se de un duelo entre varios jugadores en igualdad de condiciones.

El resultado en este tipo de partidas es favorable al jugador basado en reglas con comportamiento no aleatorio, que ha ganado el 60 % de las partidas disputadas sobre un total de 10.

Sin embargo, es mucho más interesante estudiar los resultados sobre partidas preparadas que implican una mayor complejidad en las decisiones a tomar y se puede demostrar mejor la solidez real del agente de IA basado en reglas. Además, resolver partidas en una situación avanzada permite la reproducibilidad de las mismas.

5.2.1.2 Partida desde el inicio con saldo reducido

Para simular una situación en la que no será conveniente comprar todas las propiedades en las que se cae, se ha creado una partida con el tablero completamente libre, pero con un saldo inicial de 40.000 unidades monetarias, lo que supone un capital escaso para invertir.

En este caso, al comienzo de la partida, lo esperado es que las propiedades de cada jugador crezcan muy poco a poco, para alcanzarse el mayor auge del crecimiento más adelante. Si un jugador comprara indiscriminadamente todas las propiedades en las que cae, tendría muchas opciones de perder todo su capital y caer en bancarrota.

En este escenario se demuestra el buen comportamiento del jugador automático basado en reglas para seleccionar las propiedades interesantes en situaciones de capital escaso.

En las siguientes figuras se muestra el caso de una partida ganada por el jugador amarillo.

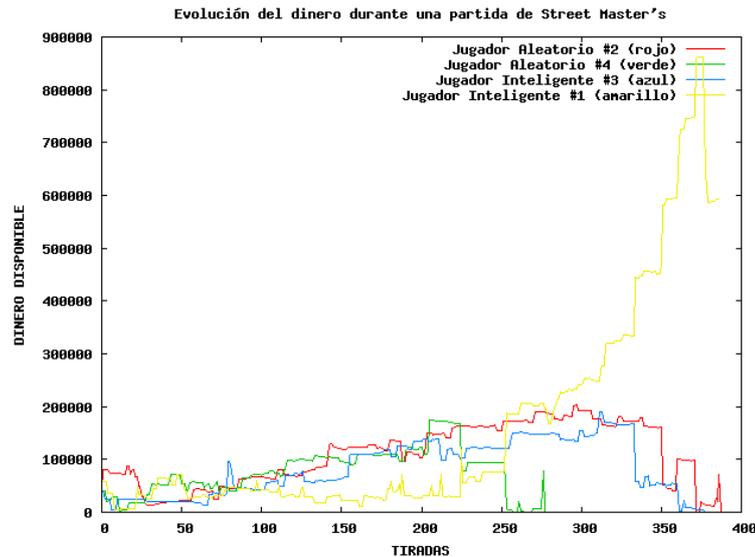


Figura 1: *Evolución del dinero durante una partida con capital reducido*

Se puede ver como el dinero no es el factor decisivo en la partida, ya que hasta aproximadamente el turno 250 el jugador ganador tenía una cantidad de dinero inferior al que tenían el resto de los jugadores. Se observa como el jugador basado en reglas es capaz de mantenerse en la partida hasta ese momento con un dinero muy escaso, lo que prueba su buena gestión del capital, tanto en las compras de propiedades (reglas del capítulo 4.3.1.2, página 71) como en las edificaciones (algoritmo de búsqueda del capítulo 4.3.2, página 76).

En general, se ha observado que el dinero no es el factor decisivo para ganar una partida de Monopoly y sí lo es la habilidad con la que se invierte.

En situaciones complicadas el agente basado en reglas es capaz de seleccionar muy acertadamente las propiedades que debe hipotecar para alcanzar el dinero necesario para cubrir sus deudas y seguir cobrando la máxima cantidad de dinero posible. En muchas ocasiones se ha comprobado claramente que el jugador, a pesar de haber tenido que hipotecar sus propiedades, es capaz de recuperar su capital gracias a las que quedan disponibles, lo cual permite que en varias vueltas al tablero recupere todas las propiedades hipotecadas. Este buen funcionamiento se debe a la eficacia del algoritmo de búsqueda desarrollado para los casos de las hipotecas y deshipotecas (véase el capítulo 4.3.2, página 76).

En este escenario no se ha podido comprobar la eficacia del algoritmo de búsqueda en las situaciones de compra y venta de edificios porque el jugador que ha podido construir no ha tenido que solventar ninguna dificultad.

Las grandes oscilaciones de dinero se deben en su mayor parte a deudas con la banca por tarjetas de suerte y caja de comunidad.

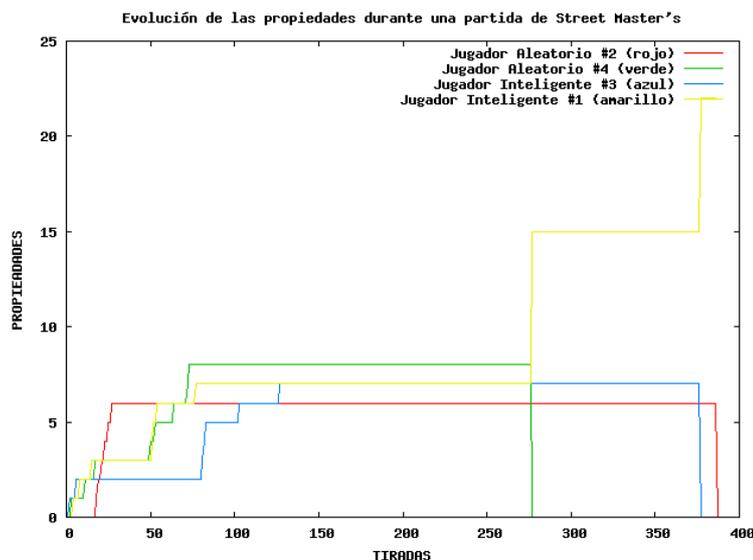


Figura 2: Evolución del número de propiedades en una partida con saldo reducido.

Como se esperaba, en esta partida concreta las propiedades están muy repartidas entre todos los jugadores, y ello no permite emitir un juicio de valor sobre la importancia de la posesión de propiedades en la partida con estos datos. Sin embargo, en otras partidas disputadas, se ve claramente como la posesión de propiedades no garantiza la victoria en la misma si no se consigue edificar y otro jugador sí lo hace. Esta observación permite afirmar que lo más importante para ganar una partida es conseguir construir y gestionar con habilidad las edificaciones y las hipotecas. Sin embargo, para poder completar monopolios es necesario adquirir propiedades. Adquirir muchas hace que la probabilidad de completarlos sea mayor.

En la figura 2 se puede observar como la compra de propiedades se hace de una forma paulatina, especialmente para el jugador azul. Esto se debe a que el jugador no ha tenido la posibilidad de comprar propiedades durante un largo periodo de tiempo, y cuando la ha tenido, no lo ha considerado

oportuno, porque los beneficios que le podrá otorgar eran menores ya que no podía llegar a completar los monopolios y su dinero era escaso, por lo que no resultaba interesante invertirlo en empresas poco rentables. Este comportamiento se debe a las reglas detalladas en el capítulo 4.3.1.2 (página 71).

En estas reglas, se ha asignado un peso en la última regla de cada tipo (2.1.7, 2.2.4 y 2.3.4) que puede variar, pero se ha demostrado empíricamente que este valor es bueno para no correr el riesgo de caer en bancarrota cuando la propiedad en realidad no es una buena oportunidad. Con valores mayores apenas compra propiedades y con menores, en muchas ocasiones, cae en situaciones de bancarrota por malgastar el capital.

En la figura 2 se puede observar que cuando un jugador cae en bancarrota, el jugador acreedor consigue todas sus propiedades. Esto se debe a que ha caído en bancarrota al tener que pagarle a él. Esta situación deja en clara ventaja a dicho jugador. Por ello, es importante llevar una buena gestión de las propiedades y sus hipotecas, y siempre mantener propiedades activas. Los jugadores basados en reglas mantienen las propiedades que suponen un mejor alquiler deshipotecadas gracias a la aplicación de la *teoría de la utilidad esperada* en el algoritmo de búsqueda (véase el capítulo 4.3.2, página 76).

En la siguiente figura se muestra como el único jugador que ha conseguido formar un monopolio y construir, es el ganador de la partida, demostrando la importancia que las edificaciones tienen en la partida.

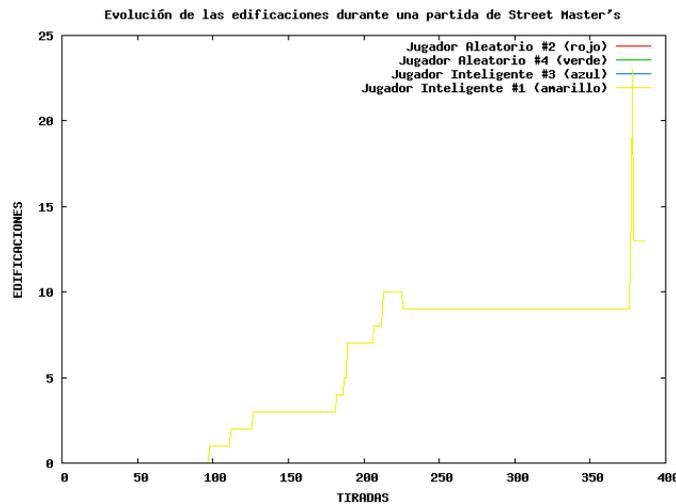


Figura 3: Evolución de las edificaciones en una partida con saldo limitado

Se han jugado 6 partidas sobre este escenario y el balance a favor del jugador basado en reglas es de 5 a 1 (83 % de victorias del jugador basado en reglas). En este caso, al igual que en el anterior, se da la situación de que el jugador que primero alcanza a formar un monopolio se muestra como claro candidato a ganar.

5.2.1.3 Partida iniciada en una situación avanzada

Un caso de estudio muy interesante ocurre cuando el juego ya ha avanzado y cada jugador dispone de un monopolio y se encuentra cerca de completar otro. En una partida diseñada así, la gestión de edificios e hipotecas decidirá el resultado final.

Un ejemplo de este caso es el que se detalla en la siguiente tabla²²:

	Jugador #1 (amarillo) Basado en reglas		Jugador #2 (rojo) Comportamiento aleatorio		Jugador #3 (azul) Basado en Reglas		Jugador #4 (verde) Comportamiento aleatorio	
	Número de casilla	Edificios	Número de casilla	Edificios	Número de casilla	Edificios	Número de casilla	Edificios
Propiedades	6	2	1	3	11	1 casa	16	-
	8	2	3	3	13	-	18	-
	9	2	31	-	14	-	19	-
	39	-	34	-	26	-	21	-
				29	-	24	-	
Dinero	60000		60000		60000		60000	
Capital total	162000		164000		168000		162000	

²² El número de casilla se corresponde con el número de la casilla del tablero proporcionado por defecto en el juego Street Master's. Para ver el tablero, se remite al lector al anexo A.

Se ha procurado que los capitales y las propiedades del juego estén repartidas de una forma equitativa entre todos los jugadores.

En la partida que se toma como ejemplo se ve una interesante competición entre los jugadores rojo y amarillo, representantes de cada uno de los agentes implementados. Se ha observado en varios escenarios generados que suelen ser más fuertes aquellos jugadores que completan monopolios más caros, que en este caso son precisamente el rojo y el amarillo.

En la gráfica que muestra la evolución del dinero de los jugadores a lo largo de la partida, se puede ver como, una vez más, el jugador rojo, que durante prácticamente toda la partida ha tenido una mayor cantidad de dinero, no es el ganador. Mención especial merece el jugador basado en reglas que ha gestionado excelentemente su situación económica durante las primeras 60 tiradas, cuando su situación se acercaba a la quiebra.

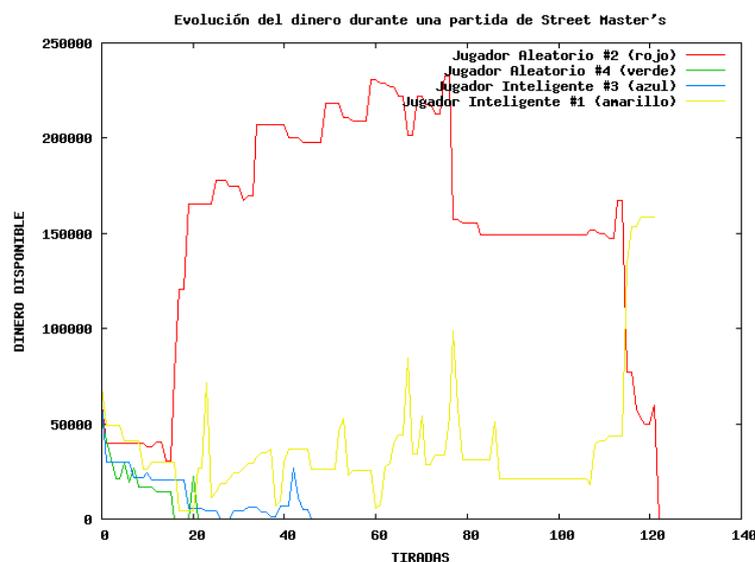


Figura 4: Evolución del dinero en una partida cargada con datos predefinidos

El jugador amarillo no sólo ha completado sus monopolios, sino que ha evitado que otros lo hagan. Durante la partida se pudo ver como el jugador rojo tuvo la posibilidad de comprar propiedades que podían evitar que el jugador amarillo completara un monopolio y no lo hizo. Esto se debe a que los agentes con un comportamiento pseudo-aleatorio no tiene en cuenta la posición del resto de los jugadores.

El comportamiento del jugador basado en reglas fue opuesto, ya que las compró evitando así tener que pagar más adelante altos alquileres. Esta situación es muy deseable, ya que se ha conseguido que la toma de decisiones no sea sólo en función de sus propios intereses, si no que también tiene en cuenta los de los demás jugadores. Este comportamiento se ha implementado en las reglas que deciden si se adquiere o no una propiedad. En el caso de las calles se trata de la regla 2.1.6, que hace que un jugador compre siempre que otro pueda completar un monopolio. Así evita que otros puedan construir.

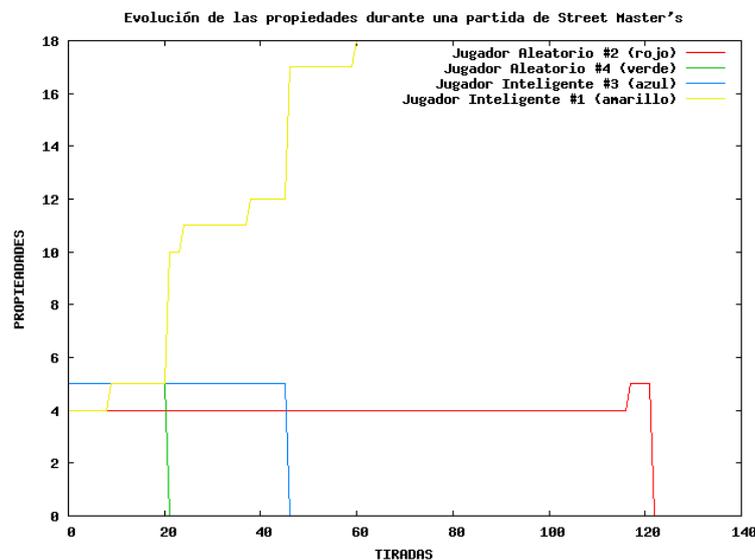


Figura 5: Evolución del número de propiedades en una partida cargada con datos predefinidos

En ambas gráficas se puede ver como el jugador amarillo experimenta un crecimiento importante cuando otro jugador es eliminado. Esto se debe a que el es responsable de su eliminación y hereda todas sus propiedades.

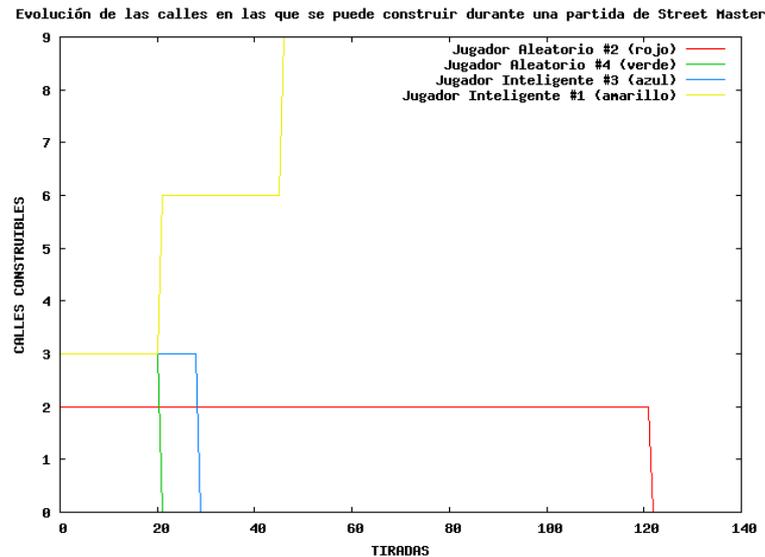


Figura 6: Evolución del número de calles edificables en una partida cargada con datos predefinidos

En la siguiente figura (figura 7) se muestra la evolución de las edificaciones de cada jugador. En este escenario sí se puede hacer una valoración del comportamiento del jugador en la gestión de edificios. Se muestra como el jugador amarillo (a la postre vencedor) consigue mantener una línea creciente de edificaciones, sólo interrumpida en torno al turno 20 por dificultades económicas (véase figura 4). Como se puede ver en ambas figuras, la recuperación del jugador es excelente. Esto es debido a que el algoritmo de búsqueda aplicado en la venta y compra de edificios es muy eficaz en todas las situaciones (véase el capítulo 4.3.2, página 76, para obtener más información del mismo). El jugador azul ha tenido que vender todas sus edificaciones, pero lo ha hecho de forma escalonada al contrario que ocurre con los jugadores rojo y verde. Esto se debe a que el algoritmo de búsqueda selecciona mucho mejor las casas a vender que el de los jugadores aleatorios (véase los jugadores pseudo-aleatorios en el capítulo 4.3.3, página 84).

En la misma figura se puede ver como el jugador verde, basado en un comportamiento pseudo-aleatorio construye muy rápidamente pero luego debe vender todas sus propiedades. Esto demuestra su ineficacia, ya que invierte sin medir sus riesgos.

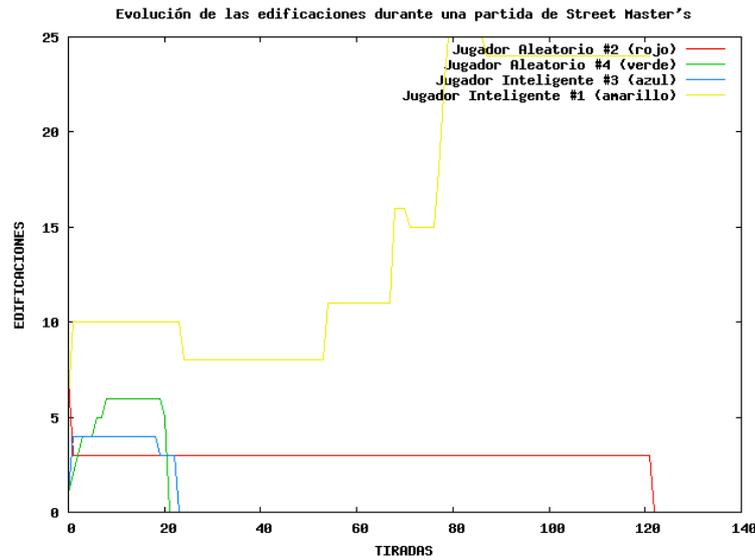


Figura 7: Evolución del número de edificios en una partida cargada con datos predefinidos

En este escenario, el jugador basado en reglas gana claramente al jugador pseudo-aleatorio por un resultado de 7 – 2 (77 % de las partidas).

5.2.1.4 Competición contra jugadores humanos

Para jugar contra jugadores humanos se han creado partidas de tres jugadores en las que intervienen un representante de cada agente de IA creado.

En las pruebas realizadas se ha podido ver como en muchos casos el jugador basado en reglas toma unas decisiones realmente buenas, pero en ocasiones demasiado arriesgadas.

En condiciones escasas de dinero, el jugador basado en reglas sólo trata de alcanzar sus propios monopolios y evitar que otros jugador los alcancen, desestimando la compra de propiedades secundarias como estaciones y monopolios parcialmente ocupados por otros jugadores. Este comportamiento se debe a las reglas 2.2.3 y 2.3.3 que hacen que el jugador no invierta en propiedades de estos tipos si ya hay jugadores que las poseen y no tienen demasiado dinero. Esto permite ahorrar dinero y sólo invertir en aquellas propiedades que se consideran mejores.

En algunas ocasiones tratar de interferir en los objetivos de los otros jugadores supone un riesgo demasiado alto que el agente de IA decide asumir. A veces, este riesgo es justificado y bien asumido, pero en algunos

casos hace que se quede en una situación delicada. Este hecho ocurre con las casillas de tipo calle en las que la regla 2.1.6 hace que se compren siempre las propiedades que pueden ser de utilidad a otros para terminar de formar el monopolio. No se tiene en cuenta el precio si se tiene dinero suficiente y puede suponer situarse en una situación delicada, pero se considera un opción acertada, ya que el desembolso que se debe realizar para comprarla es mucho menor que el dinero que puede costar un futuro alquiler. Por ejemplo²³, comprar la Calle Velázquez cuesta 18.000 unidades monetarias, mientras que un alquiler cuando se construyan 3 casas cuesta ya más del triple (55.000 unidades monetarias).

Para ejemplificar el comportamiento del jugador basado en reglas se muestra el resultado obtenido en una partida con un saldo inicial de 40.000 unidades monetarias. En este caso el ganador es el jugador humano que mueve la ficha verde.

El jugador humano supera en este caso al resto construyendo de una forma rápida y eficaz, aún arriesgando parte de su capital. Se advierte, sin embargo, que en otras ocasiones esta estrategia ha supuesto perder la partida.

El jugador basado en reglas construye demasiado despacio en esta partida. Esto se explica por lo tarde que ha conseguido completar un monopolio. Esto no depende de las decisiones que haya tomado, ya que siempre que ha podido comprar una propiedad que le facilitaba completar un monopolio, lo ha hecho. Esta situación se ha visto motivada porque no ha caído nunca en propiedades que le permitieran completar sus monopolios hasta el turno 100 (véase figura 9).

En las siguientes gráficas se muestra la evolución del dinero, en la que se ve que el agente de IA es capaz de mantenerse en juego a pesar de tener siempre un capital escaso, y la evolución de las propiedades edificables, que marcarán el desarrollo de la partida. El jugador pseudo-aleatorio tiene fuertes caídas en su capital que no presenta el jugador basado en reglas.

²³ Los datos se dan en función del tablero proporcionado como ejemplo en el anexo A, página 148

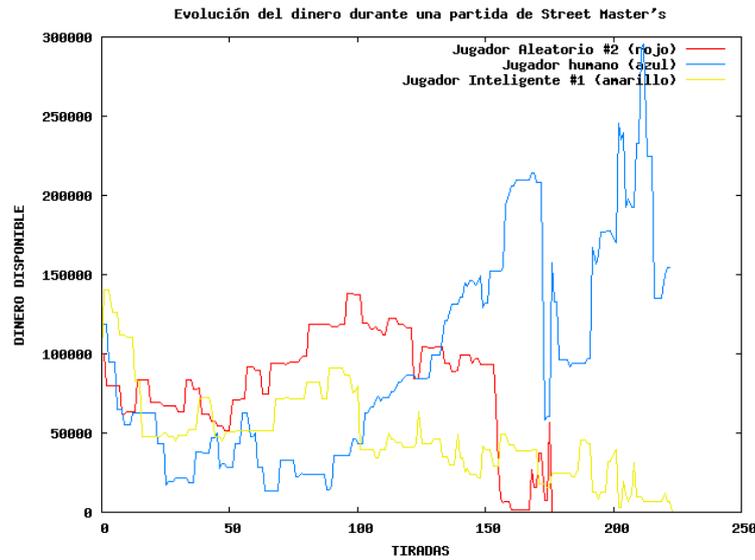


Figura 8: Evolución del dinero en una partida con contrincantes humanos

Aunque el jugador basado en reglas no ha conseguido ganar esta partida, ha demostrado una gran capacidad de resolución para solventar situaciones de escasez de dinero. Por ejemplo, en torno al turno 200 se ve como sufre un descenso en el dinero disponible que se debe a un alto alquiler que debe pagar. Esta situación la supo salvar vendiendo sus edificios e hipotecando parte de sus propiedades. En esta ocasión no ha podido aguantar en esta situación porque la diferencia era muy grande entre ambos jugadores. Sin embargo, en otras partidas sí ha conseguido remontar partidas complicadas.

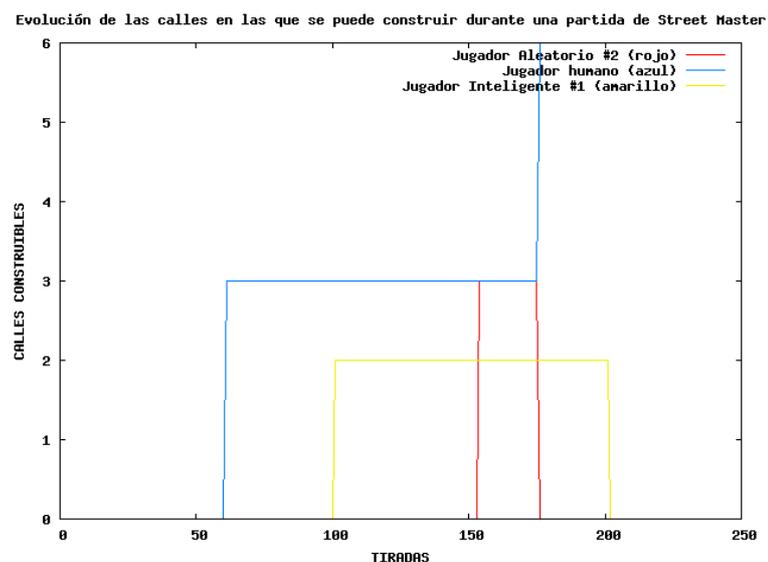


Figura 9: Evolución de las calles edificables en una partida con contrincantes humanos

A pesar de que el jugador basado en reglas no ha sido capaz de superar a jugadores humanos en muchas ocasiones, sí ha mostrado un comportamiento competitivo en todas las partidas. El agente de IA sólo ha podido ganar 2 partidas de 7 disputadas (28% de victorias).

5.2.1.5 Subastas

Las subastas son una parte muy específica de Street Master's y merecen una atención especial.

Se han realizado varias pruebas creando situaciones especiales de subastas entre varios agentes de IA con resultados muy favorables al jugador basado en reglas. Para ello, se han creado escenarios en los que todos los jugadores disponen de la misma cantidad de dinero y cada jugador posee unas propiedades diferentes para demostrar que cada agente basado en reglas sólo puja por aquello que presente un interés para él conociendo sus límites para no caer en bancarrota.

5.2.1.5.1 Subasta de una propiedad interesante para otro jugador

En el primer escenario creado para comprobar la eficacia del sistema de reglas se ha asignado a cada jugador 150.000 unidades monetarias y a un jugador aleatorio las propiedades 31 y 32 (véase el anexo A en el que se detalla el tablero al que se hace referencia, página 148). Interviene también en la subasta un agente basado en reglas que no posee ninguna propiedad. Al subastarse la propiedad número 34, que completaría un monopolio al unirlas con las que ya posee el jugador aleatorio, se han identificado los siguientes comportamientos:

- En varias ocasiones (20 % sobre 10 pruebas) el agente basado en reglas realiza la primera puja (16.000 unidades monetarias) y el jugador con un comportamiento pseudo-aleatorio se retira de la misma, perdiendo así la oportunidad de obtener una propiedad de su interés. El jugador basado en reglas consigue evitar que se forme un monopolio (regla 1.2) pujando en primer lugar y obligando al jugador pseudo-aleatorio a decidir si pujar por encima de él.
- En la mayor parte de los casos (60 %) la subasta la gana el jugador pseudo-aleatorio, pero el jugador basado en reglas consigue que el

precio de la propiedad se incrementa notablemente. Este comportamiento se debe a la regla 1.4, en la que se estima que al otro jugador le interesa la propiedad en cuestión y puja para que el precio crezca. Este es un comportamiento muy deseable para las subastas, ya que interfiere en los objetivos del resto de los jugadores haciendo que el interesado gaste más dinero del que estimaría en un principio. En el ejemplo de la propiedad 34, el agente pseudo-aleatorio llega a pagar 80903 unidades monetarias por una propiedad que cuesta 32000.

- El agente basado en reglas arriesga una importante cantidad de dinero para lograr que se incremente el precio, basándose en la idea de que el otro jugador tiene un mayor interés por la propiedad actual, por lo que se puede dar el caso de que éste se retire antes que el primero y por lo tanto el jugador basado en reglas pague una cantidad de dinero muy elevada para sus intereses. En el peor caso obtenido en las pruebas realizadas, el jugador basado en reglas ha pagado 73547 unidades monetarias por la propiedad. Esto no supone ningún problema, ya que se controla que el jugador no gaste una cantidad de dinero que le acerque a la quiebra (véase el peso asignado en la regla 1.4, que controla el gasto de cada jugador). Esta situación sólo se ha dado en el 10 % de los casos por lo que se considera un riesgo asumible.

En el resto de los casos no detallados (10 %), el jugador basado en reglas gana la propiedad por un precio menor que el 150% del precio real de la propiedad.

En general el comportamiento del jugador basado en reglas en esta situación es muy bueno, ya que interfiere eficazmente en los objetivos del resto de jugadores.

Cuando participan varios jugadores en la subasta se observa que el comportamiento es similar al detallado en este punto. La única diferencia que se ha observado se da cuando un jugador basado en reglas debe pujar y la puja máxima no pertenece al jugador que puede completar el monopolio. En dicho caso, el agente pasa el turno en esa ronda ya que el objetivo de que no se asigne la propiedad al agente que posee el resto del

monopolio se cumple sin necesidad de que intervenga, y por tanto, arriesgue parte de su capital (ver regla 1.4).

5.2.1.5.2 Subasta de una propiedad interesante para el propio jugador

También es destacable el comportamiento de un jugador basado en reglas cuando se subasta una propiedad que puede servirle para completar un monopolio.

Para ello, se ha diseñado un escenario en el que cada jugador cuenta con 150.000 unidades monetarias y un jugador basado en reglas que posee las propiedades 31 y 32 (véase el anexo A, página 148). El jugador que compite contra él por la propiedad es un jugador pseudo-aleatorio.

En las pruebas realizadas, el jugador basado en reglas consigue hacerse con la propiedad en la mayoría de los casos (80% sobre un total de 10 pruebas), sin caer en ningún riesgo de bancarrota ya que la regla 1.3 evita que esto pueda ocurrir. Las situaciones más comunes identificadas en una subasta son:

- El jugador obtiene la propiedad por un valor inferior al del precio real. En estos casos el jugador basado en reglas puja en primer lugar (ver regla 1.2) y el otro agente se retira muy pronto de la puja. Esto ha ocurrido en el 20% de las pruebas realizadas.
- El jugador alcanza el objetivo de lograr la propiedad con un precio final menor al doble de lo que marca la tarjeta de propiedad completando así un monopolio (40% de los casos).
- La situación menos deseable es obtener la propiedad por una cantidad muy elevada. Esto ocurre en el 20% de los casos. Sin embargo, esta situación no es preocupante porque está totalmente controlado que el jugador no arriesgue en las pujas más dinero del que estime conveniente (véase los pesos de las reglas 1.1 y 1.3).
- El jugador pseudo-aleatorio sólo ha conseguido ganar la propiedad en el 20% de las pruebas realizadas y siempre ha sido a un alto precio. Esto se debe a que el agente basado en reglas ha considerado la puja realizada excesiva y ha interpretado que podría ser un riesgo invertir

tal cantidad (la cantidad de dinero se limita en el peso dado en la regla 1.3).

Se han realizado también pruebas con poco dinero (50.000 unidades monetarias) y el jugador basado en reglas no ha asumido riesgos innecesarios, lo cual es una situación muy deseable. El jugador pseudo-aleatorio ha ganado la mayoría de estas pujas (70% sobre 10 intentos) pero renunciando a una cantidad de dinero superior a 30.000 unidades monetarias, que se considera mucho comparándolo con el disponible.

5.2.1.6 Otras consideraciones

Como dato cuantitativo debe mencionarse que el resultado final de las partidas disputadas entre los dos tipos jugadores automáticos en diferentes escenarios es de 35 – 15 a favor de los jugadores controlados por reglas (70% de victorias).

La mayor parte de las derrotas del jugador basado en reglas han ocurrido cuando se han presentado escenarios en los que las propiedades están totalmente asignadas formando monopolios y los jugadores deben construir. En estos casos se ha observado que el jugador aleatorio construye mucho más deprisa y provoca la bancarrota de los jugadores controlados por el sistema de basado en reglas. Sin embargo, en el resto de los casos, el jugador basado en reglas se ha mostrado muy superior.

El hecho de que las construcciones sean tan lentas en el jugador basado en reglas se debe a que se ha decidido construir sólo cuando se tiene más de 50.000 unidades monetarias. Esto es importante para asegurar que no caigan en situaciones de quiebra precipitadas. En los casos en los que se comience la partida con monopolios completos, el jugador basado en reglas jugará en desventaja, pero esta situación no es común, ya que el jugador inteligente en una partida normal, irá construyendo progresivamente sus edificios.

En general el comportamiento del cliente basado en reglas ha sido muy efectivo en partidas completas y situaciones específicas del juego.

5.3 Evaluación de la eficiencia del cliente de IA basado en reglas

Para cantidades pequeñas de dinero y de elementos en la lista a evaluar en los algoritmos de las hipotecas y compra / venta de edificios el resultado es muy bueno. Se entiende por cantidades pequeñas cantidades de menos de 50,000 unidades monetarias y menos de 30 elementos en la lista, pudiendo variar el tiempo de espera según oscilen estos valores²⁴.

El crecimiento en el tiempo de ejecución aumenta siguiendo una función exponencial en función de la cantidad de dinero y, sobre todo, del número de elementos de la lista. Se ha llegado a tener que esperar tres horas para una lista de 52 elementos y una cantidad de dinero de 128000 unidades monetarias, en el caso de la construcción de edificios.

Dados los resultados obtenidos, se puede afirmar que se ha conseguido crear un agente de IA eficaz pero ineficiente cuando el tamaño del problema crece.

²⁴ En estas pruebas se ha usado el tablero clásico de Street Master's

Capítulo 6

Conclusiones

6 Conclusiones

El trabajo ha cumplido todos los objetivos marcados al comienzo de este proyecto, logrando una completa versión informatizada de Monopoly que incluye un agente de IA capaz de jugar contra otros jugadores con unos resultados notables en cuanto a eficacia se refiere.

Además, Street Master's se ha elaborado pensando en la adaptabilidad del juego en función de las preferencias del usuario, por lo que se abre un amplio abanico de posibles configuraciones que van desde la personalización del tablero y las tarjetas hasta la configuración de varias de las opciones del juego. Esto aporta una gran riqueza al juego, ya que se puede jugar una gran cantidad de partidas diferentes en función de los gustos de cada jugador.

En este proyecto se han distinguido dos partes perfectamente diferenciadas: el diseño de la ontología y la creación de un agente de IA basado en reglas.

6.1 Ontología

El diseño realizado para Street Master's cumple todos los requisitos funcionales identificados en la fase de análisis del proyecto, lo cual permite jugar partidas de esta versión del juego del Monopoly siguiendo las reglas oficiales del mismo, mostradas en el segundo capítulo de esta memoria. Se puede afirmar que se ha realizado una representación fiel del juego original incluyendo numerosas opciones del mismo y respetando las normas preestablecidas.

Muchas de las versiones del juego analizadas no soportan algunas opciones que se han incluido en este proyecto. Por ejemplo, prácticamente ninguno de los juegos existentes permiten que una tarjeta de alguno de los mazos incluyan varias opciones para elegir entre ellas. Sin embargo, en este proyecto se ha incluido esta opción, dando así una mayor versatilidad al diseño de partidas en Street Master's.

El diseño se ha realizado pensando en posibles ampliaciones y modificaciones que puedan surgir en el futuro, por lo que la jerarquía de casillas realizada dota al sistema de una modularidad que facilitará las

labores de mantenimiento con un esfuerzo menor por parte de los desarrolladores.

6.2 Cliente de IA

El agente de IA desarrollado mediante reglas ha resultado muy eficaz contra el otro cliente diseñado en el juego (con un comportamiento aleatorio), ganando en una gran cantidad de ocasiones (70% de las disputas) y mostrando comportamientos muy deseables, sobre todo en la gestión de hipotecas y edificios, en los que soluciones conseguidas son muy acertadas y han marcado la diferencia con los jugadores con comportamientos aleatorios.

Sin embargo, su eficiencia se ha antojado deficiente cuando crece el tamaño de la lista de propiedades del jugador ya que el algoritmo diseñado se basa en buscar todas las posibilidades, por tratarse de un algoritmo de búsqueda sin información. El resultado obtenido sigue siendo muy bueno, pero el tiempo empleado es muy grande, al menos en un ordenador equipado con un microprocesador que funciona a 1.6 Ghz y 256 MB de memoria RAM.

El resto de las reglas han mostrado un funcionamiento adecuado, aplicando cierta lógica a las decisiones tomadas manteniendo una actitud principalmente agresiva en el juego, pero sin perder nunca de vista el objetivo de mantener un remanente de dinero suficiente para afrontar las posibles deudas y arriesgando sólo lo necesario, evitando de esta manera gastos innecesarios. Las reglas se pueden adaptar pensando en otras estrategias de juego.

Al tratarse de un juego de azar, se puede afirmar que es imposible lograr un agente óptimo capaz de ganar todas las partidas que dispute, ya que no se puede controlar el resultado de lanzar el dado, lo cual puede implicar que el jugador no pueda comprar propiedades ni completar monopolios, que como se ha demostrado en el capítulo de resultados, es el hecho que marca la diferencia en el juego. Sin embargo, la mayor parte de las partidas serán ganadas por jugadores capaces de razonar correctamente en la toma de decisiones.

Un buen jugador debe mostrar su superioridad a la hora de tomar decisiones complicadas, como son la gestión de edificios e hipotecas, ya que es la parte del juego que no depende del azar. En este caso, el cliente de IA desarrollado mediante reglas ha mostrado un comportamiento muy bueno haciendo uso de la "teoría de la utilidad esperada" y el algoritmo de búsqueda sin información explicado con detalle en el capítulo 4 (página 76).

Se ha demostrado en los resultados expuestos en el capítulo anterior, que el ganador de las partidas es el jugador que mejor ha sido capaz de gestionar las edificaciones y los monopolios completos.

El cliente de IA no cuenta con un sistema capaz de negociar con otros jugadores. Es un problema complejo, que debe tener en cuenta los datos de todos los jugadores para tomar decisiones acertadas.

6.3 Implementación

Por último, se quiere destacar la aplicación de librerías que tienen una influencia importante en el juego.

Por todos es sabido que la aleatoriedad absoluta actualmente es imposible en informática, por lo que se deben usar algoritmos que generen resultados pseudo-aleatorios. Este punto es importante ya que se debe utilizar la aleatoriedad para lanzar los dados, para barajar las tarjetas y para la toma de decisiones del agente de IA implementado siguiendo este comportamiento. Se ha utilizado el algoritmo de *Merssene Twister* que se ha explicado en el capítulo 4 para obtener las series numéricas con un amplio grado de aleatoriedad.

También se ha usado la librería *Readline* que aporta una mayor riqueza a la consola de juego.

Capítulo 7

Líneas futuras

7 Líneas futuras

En este capítulo se presentan las posibles ampliaciones y líneas de investigación a seguir para hacer de Street Master's un proyecto más completo y eficiente.

7.1 Clientes de IA

En esta primera versión de Street Master's, se han incluido dos tipos de agentes de IA: un jugador que se rige por un comportamiento aleatorio y el otro controlado por un sistema de producción basado en reglas.

Se interpretan dos posibles avances en este sentido dentro del proyecto:

1. *Mejorar la eficiencia y eficacia del cliente basado en reglas*

Se ha demostrado a lo largo del proyecto que el jugador basado en reglas es capaz de ganar con regularidad a un jugador con un comportamiento aleatorio y competir contra jugadores humanos con cierta habilidad, pero su eficiencia cuando el jugador controla una gran cantidad de dinero y propiedades decrece drásticamente. Esto se debe al algoritmo de búsqueda de la mejor opción para deshipotecar y construir, que se basa en un algoritmo de búsqueda sin información, lo cual implica una búsqueda exhaustiva y empeora la eficiencia (más información en el capítulo 4.3).

Por ello, se considera que se puede mejorar esta eficiencia aplicando otros algoritmos más avanzados para la búsqueda de la mejor solución.

2. *Crear nuevos clientes de IA.*

Existen técnicas más avanzadas que las empleadas en los clientes implementados en esta versión para crear agentes de IA.

En el segundo capítulo de esta memoria se han descrito los algoritmos de n-agentes, que pueden ser empleados en este tipo de problemas. Se ha prestado especial atención al algoritmo \max^n , que se considera una buena base para comenzar a desarrollar nuevas soluciones al problema.

Otra técnica que se puede utilizar para crear nuevos agentes es el aprendizaje automático. Ésta técnica consiste en recopilar datos a partir de ejemplos y procesarlos en busca de la mejor solución ante cada situación.

Los nuevos agentes se pueden probar compitiendo contra los ya existentes para probar su eficacia.

Los clientes implementados en la versión actual no cuentan con un sistema programado para resolver los procesos de negociación. Tanto las negociaciones como las subastas de propiedades se consideran retos interesantes para el desarrollo de algoritmos de búsqueda de IA.

7.2 Conexión de Street Master's con el servidor Monopd

El servidor Monopd es un servidor que permite jugar partidas a juegos similares al popular Monopoly. La descripción del servidor se incluye en el Estado de la cuestión de esta memoria (véase página 28).

La implementación de esta ampliación permitirá conectar varios clientes de Street Master's para competir entre sí, lo cual facilitará en gran medida la difusión del juego y permitirá probar los nuevos agentes contra jugadores desarrollados por investigadores diferentes.

Incluso, en el caso de que el proyecto tuviera éxito, esta ampliación podría facilitar la creación de campeonatos de jugadores de Street Master's en busca de la mejor solución.

7.3 Interfaz gráfico

La primera versión de Street Master's ha sido desarrollada bajo un entorno de texto para Linux. En futuras versiones se podría incluir un interfaz gráfico que facilite el manejo del juego a usuarios poco acostumbrados a trabajar bajo consola.

Capítulo 8

Bibliografía

8 Bibliografía

8.1 Referencias bibliográficas

Adarve Afán de Rivera, Roberto, 2006. *Proyecto AI-Live*. Madrid. Universidad Carlos III de Madrid.

Binmore, Ken, 2004. *Teoría de juegos*. Madrid. Mc Graw Hill.

Brooks, R. A., Agosto 1991. *Intelligence Without Reason*, Proceedings of 12th Int. Joint Conf. on Artificial Intelligence, Sydney, Australia.

Ceballos, Fco. Javier, 2003. *Programación orientada a objetos con C++*. 3ª edición. Madrid. Ra-Ma.

Korf, Richard E., 1991. *Multi-player Alpha-beta pruning*. Artificial Intelligence. Vol. 48. 99 – 111.

López Alejos, Guillermo, 2004. *Aplicación de la inteligencia artificial a juegos de estrategia desarrollados en entornos de software libre*. Madrid. Universidad Carlos III de Madrid.

Luckhardt, Carol A. y Irani, Keki B., 1986. *An algorithmic solution of N-person games*. Proceedings AAAI-86. 158 – 162

Matsumoto, M. and Nishimura, T. 1998. *Mersenne Twister: A 623-dimensionally equidistributed uniform pseudo-random number generator*, ACM Transactions on Modeling and Computer Simulation. Vol. 8 .3 - 30.

Muller, M. 1995. *Computer go as a sum of local games: An application of combinational game theory*. Tesis doctoral. Universidad de Alberta (Edmonton), Canadá.

Musser, David R., Derge, Gillmer J. and Saini, Atul. 2001. *STL Tutorial and Reference Guide. C++ Programming with the Standard Template Library*. 2nd edition. Estados Unidos. Addison Wesley.

Newborn, Monty, 1997. *Kasparov vs Deep Blue. Computer chess come of age*. New Yoek. Springer

Newell, A., and H. A. Simon, Marzo 1976. *Computer science as empirical inquiry: Symbols and search*. Commun. Assoc. Comput. Machinery.

Nilsson, Nils J., 2000. *Inteligencia Artificial. Una nueva síntesis*. Madrid. McGraw-Hill.

Russell J., Stuart y Norvig, Peter, 2004. *Inteligencia Artificial. Un enfoque moderno*. 2ª edición. Madrid. Pearson Prentice Hall.

Schaeffer, Jonathan and van den Herik, Jaap, 2002. *Chips Challenging champions games, computers and artificial intelligence*. Amsterdam, Holanda. Elsevier Science Publishers B.V.

Sturtevant, Nathan R. y Korf, Richard E., 2000. *On pruning – techniques for multi-player games*. AAAI-2000. 201 – 207.

Sturtevant, Nathan R., 2002. *A comparison of algorithms for multiplayer games*. Computer and games. Lecture notes in computer science. Vol. 2883. 108 – 122.

Tesauro, Gerald, Marzo 1995. *Temporal Difference learning and TD-Gammon*. Communications of the ACM. Vol. 38 No. 3.

8.2 Referencias a direcciones de Internet

Cliente Atlantik para Monopd	http://www.sourcentral.org/man/SUSE100/es/6+atlantik
Descarga de demos de los juegos comentados en el estado de la cuestión	www.softonic.com
Historia de Monopoly	http://es.wikipedia.org/wiki/Monopoly
Historia y datos de interés del juego	aula.el-mundo.es/aula/noticia.php/2005/02/11/aula1108062943.html
Inteligencia Artificial	http://es.wikipedia.org/wiki/Inteligencia_Artificial
Librería GNU Readline para C++	ftp.gnu.org
Monopoly (Hasbro)	http://es.download.games.yahoo.net/fiche.php?intIdGame=2771&strField=scr
Monopoly Here and Now	www.infobaeprofesional.com/interior/index.php?p=nota&idx=16785
Monopoly Here and Now (sitio)	www.monopolylive.co.uk

oficial)	
Póker	http://games.cs.ualberta.ca/webgames/poker/
Reglamento e historia	http://jubilo.ca/juegos/monopoly/
Reglamento oficial del juego	http://tt.tf/gamehist/rules/mg-rules.html
Risk	ask.softonic.com/ie/20999/TEGNet
Servidor Monopd	http://sourceforge.net/projects/monopd/
Sitio oficial de Monopoly en Estados Unidos	www.hasbro.com/monopoly
Sitio oficial de Monopoly en Reino Unido	www.monopoly.co.uk
STL Standard Library	http://www.msoe.edu/eecs/ce/courseinfo/stl/
Teoría de juegos	http://es.wikipedia.org/wiki/Teor%C3%ADa_de_juegos

Anexos

Anexos

A. Tablero por defecto de Street Master's

Se presenta aquí el tablero creado para jugar a Street Master's, indicando número de casilla, nombre, grupo al que pertenece o monopolio (si es aplicable) y precio.

Número	Nombre	Monopolio	Precio
0	Salida	-	20000 (Cobrar)
1	Ronda de Valencia	Azul oscuro	6000
2	Caja de comunidad	-	-
3	Plaza Lavapiés	Azul oscuro	6000
4	Impuesto sobre capital	-	20000 o 10% (Pagar)
5	Estación de Goya	Estación	20000
6	Glorieta de Cuatro Caminos	Azul cielo	10000
7	Suerte	-	-
8	Avenida Reina Victoria	Azul cielo	10000
9	Calle Bravo Murillo	Azul cielo	12000
10	Cárcel	-	-
11	Glorieta de Bilbao	Rosa	14000
12	Compañía de electricidad	Servicios	15000
13	Calle Alberto Aguilera	Rosa	14000
14	Calle Fuencarral	Rosa	16000
15	Estación de las Delicias	Estación	20000
16	Avenida de Felipe II	Naranja	18000
17	Caja de comunidad	-	-
18	Calle Velázquez	Naranja	18000
19	Calle Serrano	Naranja	20000
20	Parque gratuito	-	-
21	Avenida de América	Rojo	22000
22	Suerte	-	-
23	Calle de María de Molina	Rojo	22000
24	Calle Cea Bermúdez	Rojo	24000
25	Estación del Mediodía	Estación	20000
26	Avenida de los Reyes Católicos	Amarillo	26000
27	Calle Bailén	Amarillo	26000
28	Compañía de aguas	Servicio	15000
29	Plaza de España	Amarillo	28000
30	Vaya a la cárcel	-	-
31	Puerta del Sol	Verde	30000
32	Calle Alcalá	Verde	30000
33	Caja de comunidad	-	-
34	Gran Vía	Verde	32000
35	Estación del norte	Estación	20000
36	Suerte	-	-
37	Paseo de la Castellana	Azul	35000
38	Impuesto de Lujo	-	10000 (Pagar)
39	Paseo del Prado	Azul	40000

Cada una de las propiedades tiene unas propiedades comunes, en función del tipo al que pertenezca (calle, servicio o estación). A continuación se muestran los alquileres de cada una de ellas identificadas por su número de casilla.

Calles

Número	Alquiler						Precio edificio		Hipoteca
	Base	1 casa	2 casas	3 casas	4 casas	Hotel	Casa	Hotel	
1	200	1000	3000	9000	16000	25000	5000	5000	3000
3	400	2000	6000	18000	32000	45000	5000	5000	3000
6	600	3000	9000	27000	40000	55000	5000	5000	5000
8	600	3000	9000	27000	40000	55000	5000	5000	5000
9	800	4000	10000	30000	45000	60000	5000	5000	6000
11	1000	5000	15000	45000	62500	75000	10000	10000	7000
13	1000	5000	15000	45000	62500	75000	10000	10000	7000
14	1200	6000	18000	50000	70000	90000	10000	10000	8000
16	1400	7000	20000	55000	75000	95000	10000	10000	9000
18	1400	7000	20000	55000	75000	95000	10000	10000	9000
19	1600	8000	22000	60000	80000	100000	10000	10000	10000
21	1800	9000	25000	70000	87500	105000	15000	15000	11000
23	1800	9000	25000	70000	87500	105000	15000	15000	11000
24	2000	10000	30000	75000	92500	110000	15000	15000	12000
26	2200	11000	33000	80000	97500	115000	15000	15000	13000
27	2200	11000	33000	80000	97500	115000	15000	15000	13000
29	2200	12000	36000	85000	102500	120000	15000	15000	14000
31	2600	13000	39000	90000	110000	127500	20000	20000	15000
32	2600	13000	39000	90000	110000	127500	20000	20000	15000
34	2800	15000	45000	100000	120000	140000	20000	20000	16000
37	3500	17500	50000	110000	130000	150000	20000	20000	17500
39	5000	20000	60000	140000	170000	200000	20000	20000	20000

Estaciones

Número	Alquiler si se tiene...				Hipoteca
	1 estación	2 estaciones	3 estaciones	4 estaciones	
5	2500	5000	10000	20000	10000
15	2500	5000	10000	20000	10000
20	2500	5000	10000	20000	10000
25	2500	5000	10000	20000	10000

Servicios

En el caso de los servicios se debe multiplicar la cantidad obtenida al lanzar el dado por la que indica la siguiente tabla.

Número	Multiplicador si tiene...		Hipoteca
	1 compañía de servicios	2 compañías de servicios	
12	400	1000	7500
28	400	1000	7500