Universitat Oberta de Catalunya

uoc.edu

La Tienda de Pepe

José David Chávez Vescance

Máster Universitario en Diseño y Programación de Videojuego

Área de Posgrado Informática, Multimedia y Telecomunicación

Helio Tejedor Navarro

Nombre Profesor/a responsable de la asignatura

07/06/2020

FICHA DEL TRABAJO FINAL

Título del trabajo:	*La Tienda de Pepe*
Nombre del autor:	*José David Chávez Vescance*
Nombre del consultor/a:	*Helio Tejedor Navarro*
Fecha de entrega:	06/2020
Titulación:	*Máster Universitario en Diseño y Programación de Videojuegos*
Área del Trabajo Final:	*Área de Posgrado Informática, Multimedia y Telecomunicación*
Idioma del trabajo:	*Español*
Palabras clave	*Juegos educativos*

Resumen

Este trabajo consiste en el diseño y desarrollo de un videojuego de operaciones matemáticas básicas (suma, resta y multiplicación) hasta 100, dirigido a estudiantes que inician su vida escolar. Se simulan las interacciones que ocurren en una tienda, procurando que las operaciones matemáticas aparezcan en un contexto que resulte familiar y aplicable en el mundo real. El videojuego está pensado para ser un recurso educativo que docentes del contexto escolar pueden usar como herramienta para que los estudiantes practiquen y refuercen de manera autónoma los aprendizajes sobre las operaciones básicas que construyen en el aula de clase.

Abstract:

La Tienda de Pepe its a project about the design and development of a video game that lead to practicing basic mathematical operations (addition, subtraction and multiplication) up to 100, aimed to students starting their school life. The interactions that occur in a store are simulated, trying to make the mathematical operations appear in a context that is familiar and applicable in the real world. The video game is designed to be an educational resource that teachers in the school context can use as a tool for students to practice and reinforce autonomously the learning about the basic operations that they start at classroom.

Índice

Listado de Tablas

Listado de Figuras

1. Introducción

1.1 Contexto y justificación del Trabajo

Los juegos están en la mira de la educación, y no sin justa causa, pues consiguen lo que muchos docentes no: personas divirtiéndose mientras superan obstáculos. A menudo, en contextos laborales y educativos no se encuentra tanto interés por realizar una tarea.

La intensa motivación que generan los juegos en los jugadores ha despertado la curiosidad del mundo académico y empresarial. El poder atractivo de los juegos ha sido explicado de múltiples maneras. Se plantea, por ejemplo, que los juegos propician la vivencia de experiencia óptimas o de flujo, donde las personas alcanzan estados tan intensos de concentración que distorsionan la percepción del tiempo y de sí mismas, transformando las horas en minutos y olvidando las sensaciones de hambre, sueño y cansancio.

Las actividades que propician ese tipo de experiencias se caracterizan porque tienen metas claras, brindan retroalimentación significativa e inmediata, otorgan al usuario control o posibilidades de toma de decisiones y además balancean de manera adecuada la dificultad del reto propuesto en función de la habilidad de la persona (Csíkszentmihályi, 1990). No es raro identificar varias de estas características en los juegos.

A lo anterior, Yu-Kai Chou (2015), emprendedor en temas de gamificación, añade que algunos juegos son tan atractivos porque tienen mecánicas finamente alineadas con impulsos centrales o nucleares que están detrás de cualquier cosa que motiva a las personas: 1) el deseo de ser parte de algo más grande; 2) el interés por desplegar nuestra creatividad y recibir feedback; 3) la atracción por interactuar con otras personas y construir relaciones; 4) la curiosidad que despierta el azar y lo inesperado; 5) el impulso por querer evitar el castigo, la pérdida y el dolor; 6) el deseo de lo escaso; 7) el anhelo de poseer y controlar y 8) la motivación por querer mejorar, superar retos, crecer y sentirnos inteligentes.

Los juegos no solo han llamado la atención del sector educativo por lo atractivos que resultan. En general, jugar cualquier juego (educativo o de entrenamiento) exige el ejercicio de habilidades cognitivas y/o psicomotrices, como planeación, clasificación, formulación y comprobación de hipótesis, creatividad, toma de decisiones, razonamiento visoespacial, coordinación visomotora, cálculo, razonamiento lógico entre muchas otras más.

Justamente, por el creciente reconocimiento del potencial efecto positivo de los juegos en las vidas de las personas es que actualmente se habla de *games for chage*, *games for impact*, *gamification*, *edutainment*, *game based learning, game*, *game based pedagogy*, *serious games*.

La Tienda de Pepe es un videojuego que responde a dicha tendencia. La iniciativa de construir un juego de matemáticas en el contexto de una tienda se esbozó, por primera vez y de forma rudimentaria, en una consultoría estatal, que consistía en la producción de un videojuego sobre diversas áreas disciplinares (inglés, lenguaje, matemáticas), para niños con necesidades educativas especiales de instituciones educativas oficiales. Por el alcance y las restricciones

de dicho proyecto, en términos de tiempo, dinero y objetivos, la idea finalmente no fue refinada ni implementada, solamente quedó expuesta en una sesión cualquiera de ideación. En lugar de una simulación de las transacciones que podían ocurrir ordinariamente en una tienda, donde se utilizaran las operaciones matemáticas básicas, en dicha consultoría se adaptaron ejercicios de libros de texto en un formato digital e interactivo que brindaban retroalimentación contingente.

En el marco del Máster universitario de Diseño y programación de Videojuegos, surgió la posibilidad de retomar y depurar aquella idea antes pospuesta. Este proyecto consiste en el diseño y desarrollo de un videojuego educativo sobre operaciones matemáticas básicas (suma y resta de números naturales hasta dos cifras/digitos) para servir como recurso de enseñanza dirigido a niños y niñas de primer o segundo grado (sistema educativo colombiano).

El diseño pedagógico de la Tienda de Pepe responde a un principio clave de las teorías constructivistas del aprendizaje: la contextualización. La Tienda de Pepe simula transacciones que pueden ocurrir en un establecimiento comercial, con el fin de que el niño pueda partir de sus conocimientos previos para encontrarle un sentido a realizar cálculos utilizando las operaciones matemáticas básicas.

En la literatura educativa es frecuente la oposición de los conceptos aprendizaje repetitivo/memorístico vs aprendizaje comprensivo y profundo. En este trabajo no se aboga por la eliminación del aprendizaje que requiere la memorización de datos, hechos y/o procedimientos, lo que se critica es que la enseñanza consista exclusivamente en la memorización de información que no tiene sentido para el estudiante. Un juego educativo como la Tienda de Pepe, puede ser un recurso de enseñanza que ayude al docente a mostrar de manera interactiva la relevancia y el sentido del aprendizaje de las operaciones matemáticas básicas, al mismo tiempo que, como juego, brinda oportunidades de práctica para que el niño fortalezca y ejercite sus habilidades para realizar cálculos mentales simples, realizando sumas y restas.

1.2 Objetivos del Trabajo

Objetivo General

Diseño y Desarrollo de un videojuego educativo sobre operaciones matemáticas básicas (suma, resta y multiplicación) que simule contextos reales de aplicación de contenidos, como recurso para la enseñanza dirigida a estudiantes de primer grado de Educación Básica Primaria (sistema educativo colombiano).

Objetivos Específicos

- Diseñar y programar mecánicas simples y comprensibles que simulen las transacciones de una tienda.
- Diseñar e implementar niveles progresivos que demanden conteo, sumas y restas mentales hasta dos cifras.
- Proponer una estética visual y sonora que pueda resultar atractiva a la población objetivo.
- Afinar el juego en términos de funcionalidad, completitud y balanceo.

1.3 Método

El desarrollo del proyecto comenzó por el estudio de lineamientos gubernamentales sobre la enseñanza de las matemáticas en los primeros años escolares. Concretamente, se consultaron los Estándares básicos de competencias en matemáticas (Ministerio de Educación Nacional 2006), los Derechos Básicos de Aprendizaje: Matemáticas (Ministerio de Educación Nacional, 2016), así como libros de texto oficiales dirigidos a estudiantes de primer grado (Ministerio de Educación Nacional, 2010), con el ánimo de conocer el tipo de ejercicios que se proponen a los niños y las niñas y la manera como progresivamente se complejizan la enseñanza de los contenidos. Al mismo tiempo, también se hizo una exploración general de videojuegos de matemáticas para niños, con el fin de identificar cómo configuraban la experiencia de aprendizaje: en qué consistía la mecánica, la propuesta estética, el sistema de puntuación, la aplicación del contenido.

Posteriormente, se hizo un prototipo de papel para simular la experiencia de juego que se deseaba crear. Al identificar que la mecánica nuclear podría resultar atractiva, se procedió formular la versión inicial del documento de diseño de juego, donde se definieron las reglas, procedimientos, obstáculos, recursos, premisa y la propuesta de progresión de niveles. Concurrentemente, se hizo una propuesta del cronograma de desarrollo y se levantó un diario de trabajo para monitorear todo el proceso del proyecto (PEC 1).

Partiendo del concepto del juego formulado en el documento de diseño, se procedió a elaborar un prototipo digital (PEC 2), cuyo proceso de construcción permitió adquirir valiosos aprendizajes que modificarían el diseño del juego; por ejemplo, se decidió crear ocho niveles en lugar de cinco y se optó por implementar la mecánica de tiempo contrarreloj, a modo de recurso y obstáculo que añadiría mayor desafío a la experiencia y haría del juego un material educativo con interesantes oportunidades de práctica.

Después de construir el primer prototipo digital, que apenas contenía dos niveles, se programó una segunda versión del juego, que no estaba depurada estéticamente, pero contaba con ocho niveles funcionales. Este segundo desarrollo fue sometido a playtesting y permitió recibir feedback para mejorar la usabilidad el juego.

Una vez se crearon y probaron todos los niveles, se realizó el mejoramiento estético de juego. Esto implicó actualizar todo el componente gráfico, utilizando tanto ilustraciones gratuitas como dibujos elaborados por una diseñadora gráfica específicamente para este proyecto. Adicionalmente, se sonorizó el juego, utilizando audios gratuitos.

La construcción de la tercera versión del juego, más pulida estéticamente, se aprovechó para arreglar bugs antes no identificados y también para recibir feedback de una docente de primaria, quien aportó comentarios en cuanto la pertinencia del juego y su usabilidad.

1.4 Planificación del Trabajo

El desarrollo tuvo cuatro hitos principales: diseño del videojuego, desarrollo de la versión parcial, desarrollo de la versión jugable y desarrollo de la versión final (ver Tabla 1).

En el diseño del videojuego se hicieron actividades como definir las reglas, las acciones que el jugador podría realizar, los obstáculos, los recursos, la perspectiva del usuario (desde el tendero), el número de niveles, tipo de ejercicio por nivel, el tipo, cantidad y precio de productos que se podrían utilizar, las pantallas a mostrar y el flujo entre estas. Todas estas definiciones se registraron en el documento inicial de diseño de videojuego. Se habla de documento inicial, porque fue una base que paulatinamente se fue transformando medida que se adquirían aprendizajes durante el desarrollo del proyecto.

La versión parcial consistió en la elaboración de un prototipo digital que ilustrara la dinámica nuclear y el flujo del juego. Para esto se buscaron assets gráficos provisionales, se configuró la escena principal de juego (la tienda), se programó la arquitectura general (gestor de niveles, controlador de clientes, detector de objetos, inventario), el uso del mouse (agarrar y arrastrar), dos niveles del juego (suma de varios objetos y resta mostrando un solo producto) y el feedback general o sistema de recompensas según desempeño.

El desarrollo de la versión jugable requirió programar los niveles restantes, crear un tutorial, actualizar todo el componente gráfico (procurando que se pudiera ajustar adecuadamente a múltiples resoluciones), sonorizar las escenas e interacciones, construir el sistema de salvado de información, pulir la manera de brindar el feedback al ganar o perder el nivel y configurar la activación de niveles según desempeño tanto dentro como fuera de la escena principal de juego (la tienda).

Por último, para crear la versión final del juego, se realizaron revisiones de las mecánicas y flujo del juego, buscando y corrigiendo bugs antes imprevistos, a modo de fase de aseguramiento de la calidad. Durante esta etapa también se elaboró y publicó el showcase del juego en Unity Connect y se realizaron ajustes a la memoria del proyecto.

Tabla 1. Cronograma de trabajo

Actividades	Inicio	Final	Días
Diseño del videojuego	**19/02/2020**	**01/03/2020**	**11**
Depuración de reglas de juego.	19/02/2020	25/02/2020	6
Diseño general de niveles	26/02/2020	29/02/2020	3
Especificación de flujo de juego.	01/03/2020	01/03/2020	1
Documento de diseño del juego	23/02/2020	01/03/2020	7
Versión parcial	**02/03/2020**	**05/04/2020**	**34**
Búsqueda de assets gráficos para versión preliminar del juego	02/03/2020	02/03/2020	1
Creación y configuración de escena principal del juego	03/03/2020	05/03/2020	2
Estructuración del juego y programación de métodos core	06/03/2020	15/03/2020	9
Agarrar y arrastrar, zonas de interacción.	16/03/2020	16/03/2020	1
Múltiples clientes	17/03/2020	17/03/2020	1
Programación de nivel 1 y 2	18/03/2020	01/04/2020	14
Configuración de feedback general de nivel (estrellas según desempeño)	02/04/2020	05/04/2020	3
Versión jugable	**05/04/2020**	**24/04/2020**	**19**
Programación de niveles 3-8	06/04/2020	06/05/2020	30
Creación de escenas complementarias: inicio, créditos, niveles	07/04/2020	12/04/2020	5
Configuración de grabación de progreso.	13/04/2020	19/04/2020	6
Desarrollo de arte gráfico (menús, HUD, personajes, escenario tienda, objetos)	20/04/2020	24/04/2020	4
Definición de música y efectos de sonido	20/04/2020	24/04/2020	4
Asignación de arte gráfico.	20/04/2020	24/04/2020	4
Configuración de música y sonido	20/04/2020	24/04/2020	4
Inicio redacción memoria	29/04/2020	24/05/2020	25
Versión final	**25/05/2020**	**07/06/2020**	**13**
Pruebas de usuario	02/05/2020	12/05/2020	10
Corrección de bugs de mecánicas	27/05/2020	01/06/2020	5
Ajuste de detalles gráficos, de música y sfx	02/06/2020	04/06/2020	2
Ajustes a la memoria de trabajo final	25/05/2020	07/06/2020	13
Publicación del proyecto en Unity Connect	07/06/2020	07/06/2020	1

1.5 Productos obtenidos

Se desarrolló un videojuego educativo sobre operaciones matemáticas básicas (suma, resta e introducción a la multiplicación) hasta dos cifras que simula transacciones simples en el contexto de una tienda. El juego consta de:

- Tres escenas.
- 8 Niveles.
- 1 tutorial.
- Sistema de guardado.
- 16 personajes clientes retomados de Freepik.
- Efectos de sonido y música de acceso libre (FresSounds, Incompetech).
- Arte gráfico original: fondos de pantalla, banners, menús, 16 productos de tienda, dinero, interfaz de tienda, puntuaciones, interfaz de usuario.

1.6 Contenido de la memoria

Los capítulos que siguen realizan una descripción más detallada del proyecto La Tienda de Pepe y sus antecedentes. Los asuntos que se abordarán por capítulo son los siguientes:

Capítulo 2. Estado del arte. Revisión sobre videojuegos de matemáticas dirigidos a estudiantes de educación básica primaria y entornos de desarrollo.

Capítulo 3. Definición del juego. Presentación del diseño del juego, principalmente desde las mecánicas, propuesta estética, narrativa, elementos educativos y motivacionales.

Capítulo 4. Diseño técnico. Explicación de la arquitectura del juego, las herramientas utilizadas para construirlo, los *assets* creados y/o reutilizados.

Capítulo 5. Diseño de niveles. Descripción de los objetivos y las mecánicas propias de cada nivel del juego.

Capítulo 6. Manual de usuario. Instrucciones para el uso del juego.

Capítulo 7. Conclusiones. Principales lecciones aprendidas en el desarrollo del proyecto. Señalamiento de alcances, limitaciones y oportunidades de mejoramiento.

2. Estado del Arte

2.1 Juegos educativos y matemáticas

Juegos educativos

Jesse Schell (2015) define juego como una actividad de solución de problema a la cual nos aproximamos de manera lúdica. Tracy Fullerton (2014) define juego como un sistema formal cerrado que compromete a los jugadores en un conflicto estructurado el cual resuelve su estado de incertidumbre de manera inequitativa. Jane McGonigal (2011) retoma la definición de Bernard Suits y dice que los juegos son un intento voluntario de superar obstáculos innecesarios. También manifiesta que aquello que define un juego son las metas, los obstáculos, el sistema de retroalimentación y la participación voluntaria, pues, ciertamente, no hay juego si no se quiere jugar (lo que es conocido como el círculo mágico).

De acuerdo con Sharon Boller y Karl Kapp (2017), los juegos de aprendizaje buscan que los jugadores desarrollen o fortalezcan conocimientos o habilidades. Estos autores no realizan una distinción entre juegos serios y juegos educativos o instruccionales. Plantean que la meta de un juego educativo es el alcance de un resultado mientras la persona está comprometida o inmersa en el proceso de aprendizaje. También dicen que los juegos podrían abstraer la realidad o tener elementos de fantasía, y por ello no son replicas exactas de la vida real, como a menudo sucede con las simulaciones. En los juegos educativos, la diversión debe relacionarse tanto como sea posible con aquello que se está aprendiendo.

A diferencia de los autores anteriores, Katrin Becker (2018) sí realiza una distinción entre juegos serios y educativos. Ella expone que los juegos educativos son diseñados para permitir el alcance de una meta de aprendizaje específica. La calidad de estos se evalúa principalmente por su efectividad para conseguir que el aprendizaje deseado ocurra. El objetivo principal de los juegos serios, por otro lado, no es solamente entretener, sino transmitir un mensaje que conduzca a un cambio en las actitudes, los comportamientos, la salud, el entendimiento o conocimiento de las personas.

En este proyecto, se entenderá que los juegos educativos son actividades de resolución de problema que buscan motivar al usuario y propiciar el alcance de una meta instruccional para aportar al cierre de una brecha de conocimiento o necesidad de formación. Para los propósitos del proyecto, no es crítica la distinción entre juego serio y juego educativo, en tanto en ambos se busca conseguir un resultado de aprendizaje además del entretenimiento. Así mismo, se entenderá que un juego educativo podría parecerse más a una simulación en la medida que intente replicar con más fidelidad algún contexto o escenario real y mientras no pierda elementos esenciales como juego educativo (meta instruccional, objetivos de aprendizaje, contenido de enseñanza, reglas, procedimientos, conflicto, recursos, objetivos y sistema de feedback).

Juegos de matemáticas

En la web pueden encontrarse innumerable cantidad de videojuegos sobre matemáticas dirigidos a niños y niñas en sus primeros años escolares. A continuación, se describen y analizan solo algunos de los encontrados.

DragonBox Algebra 5+ (Kahoot DragonBox AS 2020)

Figura 1. DragonBox Algebra 5+

Videojuego de algebra para niños. Existe una escena divida en dos partes y el objetivo consiste en aislar la X de la ecuación. Al comienzo del juego, los símbolos son representados por tarjetas con figuras. La X, por ejemplo, es una caja que brilla, donde hay un dragón oculto. Poco a poco se van introduciendo conceptos de algebra, como cancelación de opuestos, balanceo, factorización, simplificaciones. Las ecuaciones se complejizan a medida que el jugador progresa. Ya en los niveles más avanzados las ecuaciones son expuestas explícitamente en formato algebraico estándar. Una de las características más destacables de DragonBox es que enseña álgebra implícitamente mientras el jugador aprende las mecánicas de juego. Al avanzar de nivel, los dragones crecen y el jugador recibe estrellas en función de su desempeño, que consiste

en resolver ecuaciones simplificando y usando la menor cantidad de movimientos posibles.

Mathblaster (JumpStart Games, 2018)

Figura 2. Mathblaster

En un ambiente 3D futurista y espacial, el jugador tiene la posibilidad de acceder a diversos minijuegos. Aquí se describirán dos de ellos. Para comenzar a jugarlos, primero se debe seleccionar el tipo de operación a repasar y el monto de las mismas. Si no se tiene una suscripción de pago, solamente estarán habilitados los niveles más fáciles.

Hyperblast. Desde perspectiva en tercera persona ubicada atrás del avatar, el jugador pilotea un aerodeslizador que navega a través de una pista tubular. El objetivo es sortear obstáculos, destruir enemigos y derrotar a los jefes. Los obstáculos son distintos tipos de barreras que el jugador puede esquivar, pero no destruir. Los enemigos son pequeños seres que aparecen en la pista tubular y pueden ser explotados si se les dispara. Los jefes y minijefes, se derrotan resolviendo correctamente las operaciones matemáticas que aparecen en pantalla. Estos villanos son una especie de pulpo robótico que propone una operación y sostiene en uno de sus brazos la respuesta correcta, la cual debe ser escogida por el jugador haciendo clic en ella. La principal diferencia entre los jefes y minijefes es que los primeros se alejan y tiran bolas de fuego que el jugador debe esquivar.

Figura 3. Minijuego Hyperblast (Mathblaster)

El jugador cuenta con diferentes recursos: escudo, corazones, bombas y balas. El escudo previene el daño que el personaje puede recibir al chocar con enemigos y con obstáculos y se puede recuperar recogiendo ítems de escudo durante el trayecto. Las bombas permiten destruir varios enemigos al mismo tiempo. Estas se recuperan a medida que se derrotan minijefes y jefes. Los disparos corrientes no se agotan. La partida inicia con tres vidas, las cuales se pierden al equivocarse en las operaciones matemáticas que muestran los minijefes y jefes.

Ataque de asteroides. Desde una perspectiva en tercera persona que se ubica tras el avatar, el jugador utiliza el teclado para controlar un cañón que permite destruir los asteroides dirigidos hacia él. Los asteroides tienen números y el jugador debe disparar a aquel que responde correctamente la operación que le aparece en pantalla. Se pierde si se destruyen asteroides incorrectos o si no se explotan aquellos que deben ser eliminados.

Figura 4. Minijuego Ataque de Asteroides (Mathblaster)

Sundae Times Lite (MANGAHIGH - Blue Duck Education Ltd. 2020)

Juego online individual o multijugador donde el usuario debe construir el helado más grande para ganar. Antes de iniciar la partida, el jugador puede elegir la operación y el monto máximo de los ejercicios a realizar. Una vez inicia la partida, en el tiempo disponible, el jugador debe utilizar el teclado numérico para responder a las operaciones que aparecen en pantalla. Si la respuesta es correcta, una bola es añadida al helado. Al final de la partida, el juego brinda diferentes retroalimentaciones al jugador: posición ocupada, cantidad de respuestas correctas, indicador de precisión, puntaje general, preguntas donde se cometieron errores con sus respectivas soluciones.

Figura 5. Sundae Times Lite

Jet Ski Addition (Arcademics, 2020)

Juego de carreras de navegador con opción de partidas multiplayer donde aparece una operación en pantalla y el jugador debe elegir la respuesta correcta entre las opciones disponibles. Gana aquel jugador que primero responda de manera correcta la mayoría de los ejercicios. Al final de cada partida aparece un resumen del desempeño que muestra posición obtenida, tiempo tomado para realizar la carrera, preguntas incorrectas, precisión de las respuestas, tasa de respuestas por minuto.

Figura 6. Jet Ski Addition en pantalla de juego

Happy Burguer (Timestables.com, 2020)

Figura 7. Happy Burguer

Juego 2D para navegador sobre las tablas de multiplicar del 1 al 12. Cada nivel es una tabla de multiplicar y por cada tabla existen días de atención al usuario (rondas). En un día el jugador es visitado por un número de clientes. El jugador asume el rol de vendedor de hamburguesas en primera persona. Los clientes se acercan y proponen un ejercicio de multiplicación de la tabla escogida. En la parte inferior de la pantalla de juego, aparecen los ingredientes de la hamburguesa. La respuesta a la multiplicación aparece de forma aleatoria en alguno de los ingredientes. Al elegir la respuesta correcta, el ingrediente se agrega a la hamburguesa. Si se responden bien todas las multiplicaciones que

propone el cliente, entonces la hamburguesa queda completa y el cliente se va feliz. De lo contrario, se va triste. Al final del día, se muestran aquellos ejercicios donde se cometieron errores, a modo de feedback educativo. Al cambiar de día, nuevos ingredientes se añaden, lo que hace que cada cliente pueda proponer más ejercicios para armar su hamburguesa. Si la mayoría de los ejercicios de cada ronda se contestan correctamente, al cambiar de día se obtiene alguna mejora para la tienda (muebles, pintura de la puerta, etc.).

La Tienda de Pepe y Happy Burguer tienen características similares: perspectiva en primera persona y clientes que se acercan a comprar. En Happy Burguer el pedido son multiplicaciones (no ingredientes), en cambio en la Tienda de Pepe el pedido son objetos y las operaciones resultan de los precios de los mismos y el dinero que tiene disponible el cliente.

Análisis Global

En la mayoría de juegos analizados, se proponen ejercicios matemáticos formalmente expresados cuya correcta resolución posibilita la progresión en el juego, según el mundo propuesto: disminuir el tamaño de una torre de helados que crece continuamente, hacer que un vehículo o una nave se desplace más rápido, evitar algún tipo de invasión, entregar al cliente la hamburguesa y hacerlo feliz. Para superar todos estos juegos, el usuario debe ejercer correctamente las habilidades que le son demandadas: sumar, restar, multiplicar. En este sentido, las actividades son coherentes con el propósito de enseñanza, que consiste en ofrecer oportunidades entretenidas para practicar el cálculo de operaciones matemáticas básicas.

En contraste con estos juegos, la principal diferencia de la Tienda de Pepe es el mundo del juego. El contexto y los resultados de calcular correctamente las operaciones procuran simular lo que podría pasar en una tienda real; por ejemplo, si el usuario no es atendido a tiempo, se aburre y se va; si le cobran más dinero del que debe, se marcha molesto; si le ofrecen más vueltos -suponiendo una cliente ético-, avisa que el jugador (tendero) está haciendo mal sus cálculos. Como se ha mencionado antes en esta memoria, se pensó este diseño del juego por dos motivos principales: primero, con el ánimo de que los niños puedan encontrarle relevancia al contenido que aprenden debido a su probable utilidad en la vida cotidiana; segundo, al utilizar un contexto familiar, se facilita el anclaje de los conocimientos nuevos a los saberes previos.

2.2 Plataformas de desarrollo

Existen muchas plataformas de desarrollo en el mercado, tanto gratuitas como de pago, que posibilitan la creación de interesantes experiencias de juego en 2D. Aquí solamente se describen algunas de estas.

Builbox (AppOnboard, 2020). Entorno de desarrollo gratuito con versión expandida de pago que utiliza programación visual a través de bloques. Permite publicar juegos para iOS, Android, Windows y Steam. Cuenta con numerosas plantillas para iniciar la creación de juegos, mapas mentales para organizar el flujo del juego, mundos 3D o 2D, funciones predefinidas para asignarles a los objetos y herramientas para facilitar la monetización a través de anuncios.

Corona (Corona Labs Inc, 2018). Plataforma gratuita de desarrollo de juegos móviles en 2D. Su promesa de valor consiste en permitir que cualquier persona pueda crear juegos para dispositivos móviles. Corona posibilita la exportación de juegos a múltiples plataformas (iPhone, iPad, Android, Amazon Fire, Mac, Windows). Utiliza el lenguaje de programación Lua. Promociona más de 1000 APIs para facilitar la experiencia de desarrollo. El entorno cuenta con un simulador que permite tener una vista previa inmediata de los cambios realizados en el código. Corona permite realizar pruebas en vivo de los juegos creados directamente en dispositivos.

Construct (Scirra Ltd, 2020). Motor de pago que posibilita programación visual. Utiliza lenguaje de programación JavaScript. Funciona también desde tabletas, móviles e incluso desde el navegador. Permite publicar juegos para Steam, iOS, Android, Windows, Web (Kongregate, Newgrounds, itch.io). Cuenta con línea de tiempo para animaciones, más de 1000 tutoriales, vista previa instantánea, motor de físicas, editor de imágenes y tilemaps, debugger y herramientas para crear juegos multiplayer, entre otras funcionalidades.

GameMaker Studio 2 (YoYo Games, 2020). La propuesta de valor de este motor consiste en poner al alcance de todos el desarrollo de videojuegos de alta calidad. Este es un motor apto para personas con pocos conocimientos de programación. En GMS2 se han desarrollado reconocidos videojuegos en 2D, como Hyper Light Drifter. Tiene un lenguaje de programación propio y, al igual que Unity, permite compilar para múltiples plataformas.

Godot (Juan Linietsky, Ariel Manzur y contribuidores, 2020). Motor gratuito y de acceso abierto. Admite diversos lenguajes de programación. Permite crear juegos en 3D y 2D y herramientas personalizadas. Cuenta con editor de Tilemaps, editor de animaciones, debugger. Funciona en dispositivos móviles. Permite publicar en múltiples plataformas y cuenta con características para facilitar el trabajo en equipo.

Phaser (Photon Storm Ltd, 2020). Entorno gratuito y de acceso abierto que utiliza lenguaje de programación JavaScript o TypeScript y permite programar juegos en HMTL5 para navegadores de escritorio y dispositivos móviles.

Unity (Unity Technologies, 2020). Motor de creación de juegos 2D y 3D que permite compilación para múltiples plataformas (Android, PC, iOS, Windows phone, etc.). El lenguaje de programación que utiliza es C#. Unity tiene múltiples herramientas que lo convierten en uno de los entornos de desarrollo de juegos más versátil y atractivo en el mercado (gestor de cámaras, consola de sonido, servicios en línea, físicas 2D y 3D, editor de terrenos, editor de tilemaps, pathfinding, sistema de partículas, sistema de shaders, entre muchas más funcionalidades). Unity dispone de video tutoriales y una extensa documentación que facilita su aprendizaje, además cuenta con una enorme comunidad que activamente genera contenido explicativo en inglés y en español (tutoriales, documentación), participa en foros y crea herramientas y recursos que agilizan y enriquecen el proceso de desarrollo (como assets gráficos y aplicativos que expanden las funcionalidades del motor).

3. Definición del juego

Descripción breve

La Tienda de Pepe en un juego de un solo jugador sobre las operaciones matemáticas básicas (suma, resta y multiplicación). El jugador desempeña el rol de un tendero que atiende clientes, debe indicarle a estos cuánto suma la compra, cuánto dinero falta o cuánto dinero sobra, así como entregar la cantidad de producto según el dinero que tiene disponible el cliente. Funciona como una simulación de transacciones simples que podrían ocurrir en una tienda real.

Evolución general del concepto

Al comienzo, solo existía la intención de crear un juego de matemáticas para estudiantes de primer o segundo grado de Educación Básica Primaria. No había idea alguna sobre cuál sería la meta del juego, sus mecánicas, propuesta estética, narrativa o elementos formales.

Se revisaron lineamientos oficiales y libros de texto dirigidos a niños y niñas en sus primeros años escolares para identificar qué competencias matemáticas se esperan desarrollar al comienzo de la escuela y qué tipo de ejercicios son propuestos a medida que avanza el currículo.

La inspiración para diseñar la Tienda de Pepe, provino de una cartilla de matemáticas para estudiantes de primer grado publicada por el Ministerio de Educación Nacional (2010) en el marco del modelo Escuela Nueva, una pedagogía activa que en Colombia ha sido implementada para la educación rural. En dicha cartilla, se proponían ejercicios de suma o resta partiendo del escenario de una tienda.

Figura 8. Ejercicio de suma en contexto de una tienda. Imagen de libro de texto dirigido a estudiantes de 1er grado.

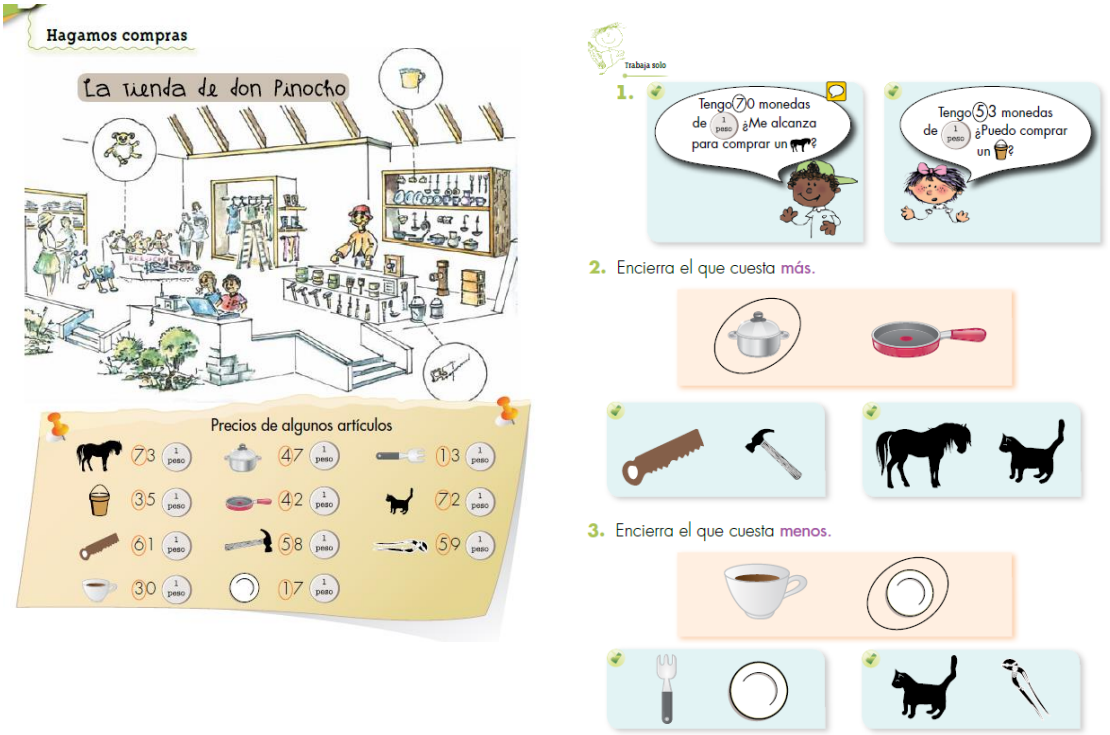

Figura 9. Ejercicios varios en contexto de una tienda. Imagen de libro de texto para estudiantes de 1er grado.

En general, por su propio formato, los libros de texto tradicionales son estáticos: no se pueden cambiar dinámicamente el contenido de los ejercicios, a menos que el docente tenga la iniciativa de proponerlos.

Reconociendo que la actividad de la tienda podría ser interesante por su contexto y pertinencia y entendiendo que en un videojuego los ejercicios a realizar no se agotarían tan prontamente, se decidió diseñar un juego que sirviera a los mismos propósitos educativos, simulando las transacciones a realizar en una tienda, cuya resolución demandaría al estudiante la realización de cálculos matemáticos básicos.

Inicialmente se pensó en una perspectiva en tercera persona, donde habría un indicador de clientes a atender y una cara que mostrara la satisfacción general de estos. Los clientes se acercarían a pedir objetos que el tendero entregaba. El cliente haría la pregunta de cuánto debe y el tendero (jugador) tendría que brindar la respuesta. Esta idea inicial tenía aún muchos vacíos. Por ejemplo, no estaba claro el tema del inventario ni como el jugador tendría que dar su input, ni mucho menos cuál sería la progresión adecuada de niveles.

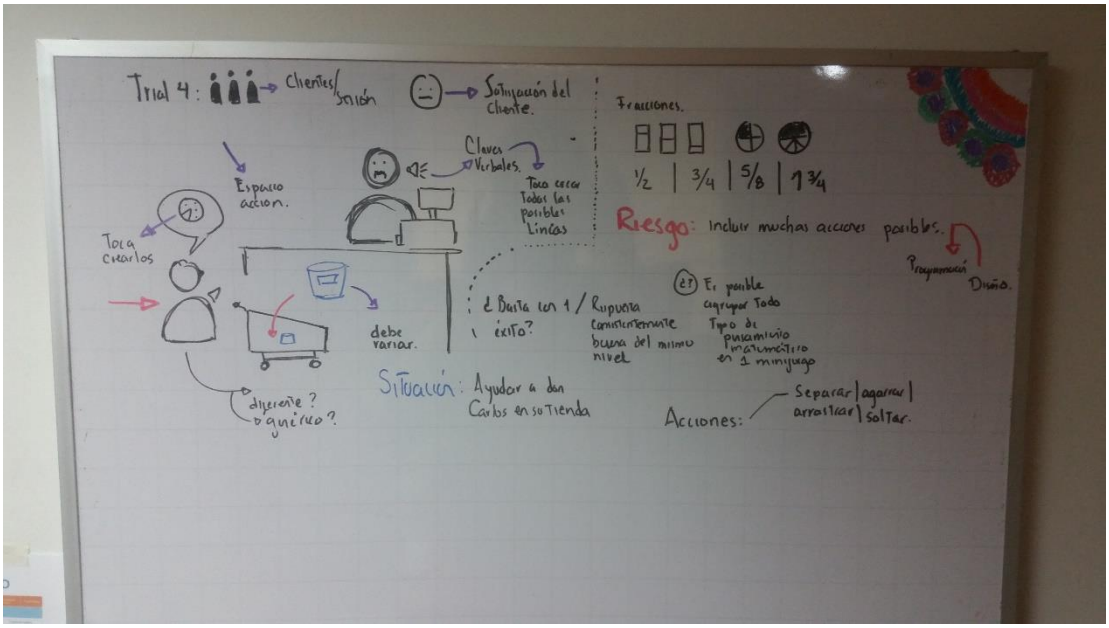

Figura 10. Uno de los primeros bocetos a mano de la Tienda de Pepe

Después, recordando el juego *Papers, Please* (Pope, 2013), surgieron varias ideas para mejorar el concepto inicial. Se adoptaría una perspectiva de juego en primera persona, desde el punto de vista del tendero (Pepe), que sería controlado por el jugador. Los clientes llegarían, harían su pedido, el tendero entregaría los productos solicitados y tendría que utilizar los números en pantalla para responder a las preguntas que harían los clientes.

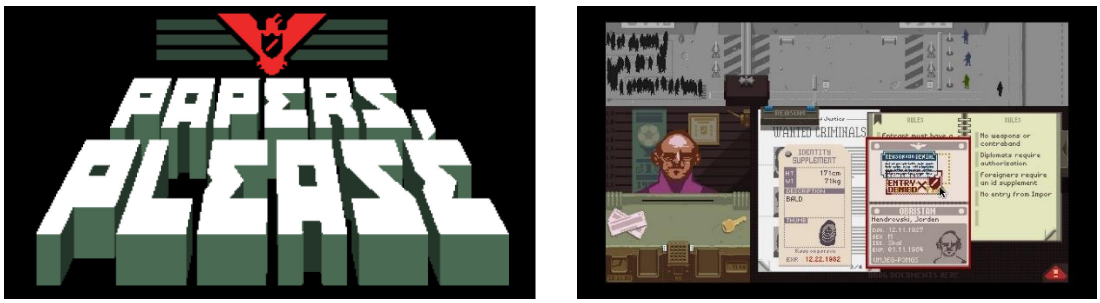

Figura 11. Papers, Please (Pope, 2013)

Ya teniendo una mayor claridad sobre la dinámica nuclear del juego, siguió un proceso de definición de las características del juego, descartando, ajustado o adoptando ideas a medida que se avanzaba en la construcción de los prototipos.

En el camino se tomaron sutiles decisiones de diseño que quedan enmascaradas tras el producto, pero responden a consideraciones analizadas detenidamente. Por ejemplo, se podría pensar ¿Por qué el jugador tiene que agarrar, arrastrar y soltar los productos en el mostrador o devolverlos al inventario si esto es algo que se podría hacer automáticamente, de tal modo que la operación a realizar aparecería lo más pronto posible sin necesidad de esas microgestiones? En este caso, la decisión de diseño (conservar el *drag and drop*) obedeció principalmente a cuatro razones: 1) se siente más real; 2) los antecedentes del aprendizaje de

la suma y la resta parten de la clasificación y del conteo; 3) desde algunas perspectivas de psicología educativa interesa poder evidenciar los estados de entendimiento de quien aprende por las acciones que realiza; y 4) el ejercicio de sumar y restar puede facilitarse si los productos de la tienda pueden manipularse uno a la vez.

Meta instruccional

Al finalizar el juego, el estudiante estará en capacidad realizar mentalmente operaciones matemáticas básicas mentales (suma y resta).

Objetivos de aprendizaje

El estudiante podrá:

- Ejecutar conteos.
- Calcular sumas de números naturales de dos dígitos hasta 99.
- Realizar restas de números naturales con diferencias menores a 99.

Población objetivo

Niños y niñas entre 6 a 9 años que cursa el primer o segundo grado de educación básica primaria y tienen aproximación inicial a la lectura.

Meta del juego

Hacer felices a los clientes, entregándoles los productos que solicitan y calculando correctamente las respuestas a sus preguntas, antes que acabe el tiempo de espera.

Dinámica nuclear

Los clientes llegan a la tienda y hacen pedidos, que demandan al jugador la realización de cálculos mentales rápidos utilizando las operaciones matemáticas básicas (especialmente suma y resta, aunque existe un nivel donde se abordan principios de multiplicación). Con el ánimo de que los niños a quienes está dirigido el juego puedan resolver los problemas propuestos, el monto máximo que puede tener una transacción es $99.

Mecánicas

Reglas y sistema de retroalimentación

El juego tiene 8 niveles, que se diferencian por los contenidos que abordan y por su nivel de dificultad (ver Capítulo 5).

Se pierde el nivel con 5 fallas y se gana realizando 8 ejercicios correctos. El número mínimo de intentos para ganar un nivel es 8 y el máximo es 12.

El monto máximo de la transacción aumenta cada dos ejercicios contestados correctamente. Primero el monto máximo es $20, luego es 50$, $80 y $99. Las denominaciones fueron definidas pensando en que pudiera armarse cualquier número natural entre el 1 y el 99.

La gestión de intentos en general funciona de la siguiente manera: al iniciar el nivel, los dos primeros ejercicios requieren realizar sumas o restas (según el caso) donde el monto máximo de la transacción es $20. Si se contesta correctamente, luego se mostrarán dos ejercicios cuyo monto máximo podría ser hasta $50. No se aumenta el valor límite de la transacción hasta que el jugador complete al menos dos ejercicios con el monto establecido. De esta manera, el jugador debe atender correctamente a 8 clientes (2 para un monto máximo de $20, 2 para $50, 2 para $80 y 2 para $100).

Se optó por implementar esta progresión de los montos dentro del nivel principalmente por dos razones: 1) para brindarle dinamismo a cada nivel, añadiendo dificultad variable; 2) para adaptar los ejercicios al desempeño del usuario.

Los objetos pedidos por el cliente son aleatorios. A medida que aumenta el monto de la transacción, es posible que el cliente pida objetos más costosos, y por esto, dentro del mismo nivel también puede incrementar la dificultad.

Según las acciones del usuario ante el pedido del cliente, el juego brinda diferente feedback:

- Si el jugador no ha puesto objetos en el mostrador, el cliente dice que faltan objetos.
- Si el jugador pone menos objetos que los solicitados por el cliente, este dice que faltan objetos.
- Si se ponen más objetos que los solicitados por el cliente, este dice que hay objetos de sobra.
- Si se ponen en el mostrado la misma cantidad de objetos solicitados por el cliente, pero estos no son exactamente los solicitados, el cliente dice que el pedido no está bien.
- Si se ponen los mismos objetos solicitados por el cliente, este pide la cuenta (suma) o dice cuánto dinero tiene disponible (resta), para que el jugador haga la diferencia.
- Si se cobra un menor valor al correcto, el cliente dice que está cobrando menos.
- Si se cobra un valor mayor al correcto, el cliente dice que se está cobrando más, se retira y los objetos que estaban en el mostrador vuelven al inventario. Estos casos se puntúan como resolución incorrecta del ejercicio y aumentan el indicador de insatisfacción.
- Si se brinda la respuesta correcta, el cliente brinda las gracias, entrega el dinero correspondiente al monto de la transacción y se suma un punto positivo al jugador.

Algunos errores no son penalizados, otros sí. Simulando una tienda, se cuenta como error lo que en afectaría más la satisfacción del cliente (cobrar de más o dar menos vueltos de los que corresponden). Cuando el cliente recibe menos cambio o si se cobra más de lo debido, el cliente se va insatisfecho y el intento cuenta como un error. Si hay tiempo, y el jugador configura erróneamente el pedido, esto no es castigado por el sistema de puntaje. No obstante, si el pedido está incorrectamente entregado cuando finaliza el tiempo, entonces el intento

cuenta como error. Existen algunas equivocaciones (como cobrar menos de lo correcto o dar más vueltos que los debidos) que, mientras haya tiempo, no afectan la satisfacción del cliente, pero sí el puntaje final; es decir, no determinan la victoria del nivel, pero sí impactan la puntuación obtenida, de tal modo que es posible aprobar un nivel sin obtener puntos. Esta decisión es coherente con los planteamientos de Boller y Kapp (2017), quienes plantean que en los juegos educativos no deberían exigirse desempeños perfectos para progresar, ya que podrían crearse experiencias frustrantes en los jugadores que los apartarían del juego para siempre, mientras que si se admiten desempeños no perfectos para avanzar, aún queda un margen para practicar y aprender.

Si el nivel se resuelve sin cometer errores de cálculo o sin dejar clientes insatisfechos porque no se les entregó lo que pedían, se otorgan tres estrellas. Por un (1) error se dan dos estrellas, por dos (2) errores se entrega una estrella y a partir de tres (3) errores no se brindan recompensas. No es posible ganar el nivel en más de 12 intentos, por tanto, si al intento 12 no se completan 8 aciertos, finaliza la jornada y el jugador debe repetir el nivel.

Al final del nivel, se muestra al usuario un mensaje que dice si ganó o perdió y cuántos puntos obtuvo según la cantidad de aciertos y errores. El puntaje más alto obtenido en cada nivel es el que queda grabado y es mostrado en la pantalla de selección de niveles. Un puntaje bajo en un nivel no reemplazará un puntaje más alto conseguido previamente. Debe aprobarse un nivel (así sea sin estrellas), para que el siguiente nivel sea desbloqueado.

Acciones

Las acciones que más frecuentemente realiza el jugador son (a) agarrar, arrastrar y soltar y (b) señalar y hacer clic. El jugador puede agarrar, arrastrar y soltar objetos en el mostrador. Esto sucede cuando el cliente pide mercancía de la tienda y el jugador se la entrega. Por otro lado, señala y hace clic cuando configura y confirma su respuesta; por ejemplo, si el cliente pide A y B productos y pregunta cuánto cuesta, el jugador tendrá que marcar, borrar o confirmar el valor de la transacción utilizando las teclas disponibles en pantalla.

Objetos

A continuación, se resumen los principales objetos del juego y la manera como se relacionan entre sí, indicando las acciones que pueden realizar.

Tabla 2. Descripción de objetos principales (atributos, estados y acciones)

Objetos	Atributos	Estados	Acciones
Clientes	Apariencia Tipo de diálogo (pedido objeto, propuesta de problema, retroalimentación)	Dentro o fuera de tienda.	Propone cálculos, hace pedidos al jugador y brinda retroalimentación.

Dinero	Forma y color (moneda o billete) Valor ($50, $20, $10, $5, $2, $1)	En caja, en cliente, en mostrador.	Es mostrado por el cliente para indicar la cantidad que tiene disponible para la compra. También es entregado por el jugador como vuelto o es recibido como pago por los productos que adquiere el cliente.
Productos o artículos de la tienda.	Tipo (fruta, aseo personal, papelería, ropa) 4 ejemplares por cada categoría. Forma. Precio.	En inventario. En mostrador. En cliente. En retorno al inventario. En tránsito.	Son puestos por el jugador en el mostrador para ser entregados al cliente. En caso de respuesta o posicionamiento incorrecto de los objetos, estos vuelven al mostrador.
Display	Número	Con números Sin números	En el display se muestran los valores que el jugador escoge como respuesta numérica a las preguntas realizadas por el cliente.
Botón borrar	Forma y color Capacidad de ejecutar acción al ser presionado	Ejecutado / No ejecutado	Al ejecutarse borra un número del display
Botón confirmar	Forma y color Capacidad de ejecutar acción al ser presionado	Ejecutado / No ejecutado	Al ejecutarse confirma la entrega del pedido o el valor ingresado como respuesta
Botón número	Forma y color Capacidad de ejecutar acción al ser presionado	Ejecutado / No ejecutado	Al ejecutarse ingresa un valor en el display que indica la respuesta al problema propuesto por el cliente.
Mostrador	Detecta colisiones.	Con o sin los objetos correctos. Con dinero o sin dinero del cliente.	Detecta qué objetos colisionan y reporta si la configuración es correcta o incorrecta de acuerdo con el pedido del cliente.

Cada artículo tiene un precio fijo. Los artículos y sus respectivos precios están listados en la Tabla 3. Estos artículos pueden ser agarrados, arrastrados y soltados para entregárselos al cliente. También existe dinero, que automáticamente es dispuesto en la caja, es devuelto al cliente o es puesto en el mostrador. Adicionalmente, el jugador dispone de una “calculadora” que permite indicarle al cliente los resultados de los cálculos realizados. Además,

existe un botón para confirmar la respuesta y otro para borrar los números ingresados.

Tabla 3. Productos de la Tienda de Pepe

ID de la categoría	Categoría	ID del producto	Producto	Precio
1	Frutas	11	Manzana	5
		12	Pera	4
		13	Limón	1
		14	Banano	2
2	Elementos de aseo	21	Jabón	7
		22	Cepillo	9
		23	Crema dental	10
		24	Shampoo	20
3	Papelería	31	Regla	8
		32	Lápiz	3
		33	Cuaderno	20
		34	Tijeras	15
4	Ropa	41	Jean	50
		42	Camisa	40
		43	Buso	60
		44	zapatos	80

Conflicto

En la Tienda de Pepe, existen dos fuentes generadoras de conflicto: los ejercicios matemáticos y el tiempo de espera de los clientes. En la medida que el jugador pueda realizar las operaciones matemáticas más fácil y rápidamente, la intensidad del conflicto disminuirá.

Recursos

Tiempo. Los clientes tienen un tiempo de espera para realizar su compra. Si el tiempo de espera se acaba y no han recibido los productos que pidieron o no se ha resuelto correctamente el ejercicio, el cliente se va y el intento es considerado fallido. En algunos niveles, el tiempo de espera es mayor, dado que el jugador

tiene que hacer más micromanejo, por ejemplo, el tipo de ejercicio dónde primero el cliente pide unos objetos y al darse cuenta de que no tiene suficiente dinero, solicita quitar algunos de ellos.

La mecánica de tiempo límite no estaba incluida en el diseño inicial del juego. Se pensaba que el jugador debería tener cuantos minutos necesitara para realizar las operaciones. Así, el desafío principal estaba solamente en poner los objetos correctos y resolver acertadamente la operación. Se añadió la mecánica del tiempo con el fin de incrementar el reto y hacer el juego más interesante, pues se percibió que, sin tiempo en contra, las habilidades requeridas para resolver el problema podrían exceder considerablemente el reto propuesto, lo que provocaría una experiencia de aburrimiento y apatía, en lugar de concentración y disfrute.

Intentos. El jugador tiene hasta 12 intentos para ganar cada nivel. Debe acertar 8 ejercicios para ganar y, si comete 5 errores que generan insatisfacción al cliente (dar menos vueltos de los correctos, cobrar más de lo debido, no entregar el pedido en el tiempo definido), entonces pierde. Existe un indicador que comunica al jugador su desempeño (cara feliz y triste). La cara feliz se completa con los aciertos y la cara triste con los desaciertos. Si bien este indicador no dice específicamente el número de clientes que faltan por atender, si muestra qué tanto falta para ganar o perder.

Elementos narrativos

Historia, ambientación y trama

El jugador es el cajero de la Tienda de Pepe, a quien le tocará realizar mentalmente los cálculos para atender satisfactoriamente a los clientes. Todo el juego se desarrolla en la tienda de Pepe. El jugador tiene la perspectiva del cajero, es decir, observa de frente a los clientes. A su alcance tiene una “calculadora” para indicar las respuestas y los productos para entregarlos a los usuarios.

Personajes

El jugador es el cajero. Los otros personajes son los clientes, quienes acuden a la tienda para comprar sus víveres.

Estética

Propuesta gráfica.

Dado que el juego está dirigido a niños y niñas que inician su vida escolar, se eligió una propuesta estética con características acordes a la población objetivo, con colores cálidos, alegres, llamativos. Se eligieron personajes caricaturescos y fuentes orgánicas, amplias, legibles, con curvas y rellenas.

La siguiente imagen muestra el primer boceto digital del aspecto estético esperado.

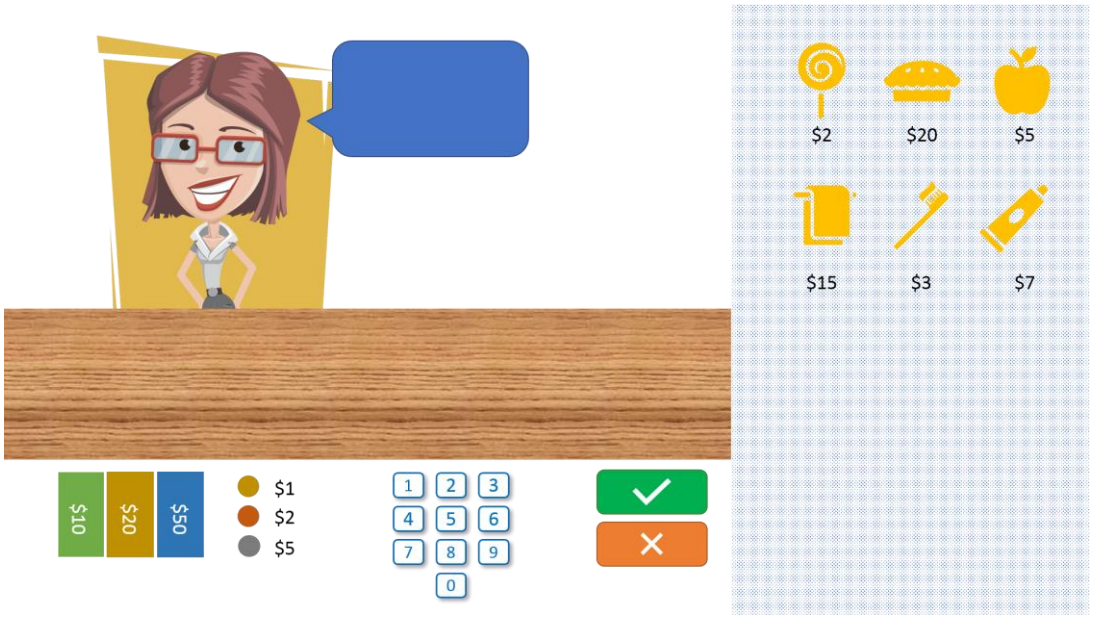

Figura 12. Boceto digital de la tienda.

Para la elaboración del prototipo digital, se optó por una propuesta gráfica funcional, que permitiera comunicar y experimentar la dinámica nuclear del juego.

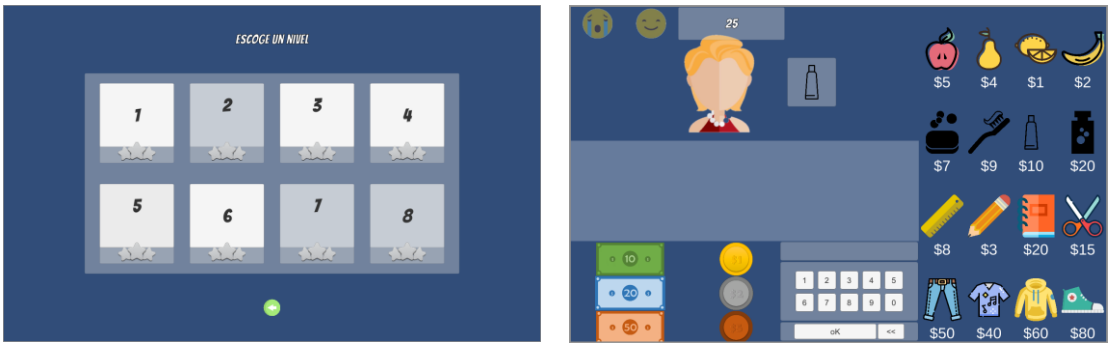

Figura 13. Propuesta gráfica preliminar

Finalmente, para construir la versión alpha del juego, se contrataron los servicios de una diseñadora gráfica que elaboró las piezas que terminarían implementándose en el juego.

Figura 14. Propuesta estética final de la Tienda de Pepe

Sonorización

Desde el diseño del juego, se pensó que los efectos de sonido debían reforzar la idea de que el personaje principal se encontraba en una tienda. Por eso se asignarían sonidos a acciones como agarrar, soltar y presionar teclas. Por supuesto, para generar una sensación de mayor dinamismo, también se agregarían sonidos para indicar el buen y mal desempeño en los intentos y para reforzar las condiciones de pérdida o victoria al final de los niveles. También se pensó en utilizar solamente dos pistas de audio, una para las pantallas de inicio del juego y escogencia de niveles y otra para la tienda. En cuanto al género de las pistas, desde el comienzo se buscó que resultarán alegres, cómicas, animadas.

Instrucciones y textos

Pensando que la población objetivo probablemente apenas se estaría iniciando en procesos de lectura, se tomaron algunas decisiones para facilitar el uso del juego: navegación simple entre pantallas (sin muchas ramificaciones), oraciones cortas, vocabulario simple.

4. Diseño técnico

4.1 El entorno de desarrollo y sus requerimientos técnicos

Para desarrollar el videojuego la Tienda de Pepe, se escogió el motor Unity. Esta elección se justifica principalmente en razones personales del autor del proyecto, quien durante el curso de la Maestría adquirió más experiencia en dicho entorno. Las restricciones en términos de los recursos temporales disponibles hacían inviable la posibilidad de familiarizarse con otro motor, por más atractivo que pareciera.

En el sitio oficial de Unity se mencionan los siguientes requerimientos de sistema para crear o jugar juegos desarrollados en este entorno.

Para el desarrollo

- **Sistema operativo**. Windows: 7 SP1+, 8, 10, solo versión 64-bit. Las versiones de servidor de Windows no han sido probadas.
- **CPU**. Conjunto de instrucciones de apoyo SSE2.
- **GPU**. Tarjeta gráfica con capacidades DX10 (shader model 4.0).
- Si se desea desarrollar para dispositivos Android o Windows, se requiere:
- **Android**. Android SDK y Java Development Kit (JDK). IL2CPP scripting backend requiere Android NDK.
- **Universal Windows Platform.** Windows 10 (64-bit), Visual Studio 2015 con componente de C++ Tools o superior y Windows 10 SDK.

Dependiendo de la complejidad del proyecto, se puede necesitar otros requerimientos.

Para ejecutar juegos creados en Unity

Las aplicaciones creadas en Unity pueden ejecutarse en la mayoría de los dispositivos y sistemas. Lo bien que se ejecuten las aplicaciones creadas dependen de la complejidad del proyecto. Algunos requerimientos son

Computadores de escritorio.

- **Sistema operativo.** Windows 7 SP1+
- **CPU**. Conjunto de instrucciones de soporte SSE2.
- **GPU**. Tarjeta gráfica con capacidades DX10 (shader model 4.0).

Android.

- OS 4.1 o superior.
- ARMv7 CPU con soporte NEON o Atom CPU
- OpenGL ES 2.0 o superior.

WebGL

- Cualquier versión reciente de navegador de escritorio como Firefox, Chrome, Edge o Safari.

Universal Windows Platform

- Windows 10 y tarjeta gráfica con capacidad para DX10 (shader model 4.0).

4.2 Herramientas.

Durante el desarrollo del proyecto se utilizaron diversas herramientas, además de Unity. Las principales fueron:

- **Bitbucket**. Repositorio en línea usado para gestionar las versiones del proyecto.
- **Microsoft Word.** Se utilizó para llevar un Diario de trabajo, así como para redactar entregas parciales y la memoria del proyecto.
- **Excel.** Utilizado para gestionar el cronograma.

- **Sourcetree.** Software utilizado para administrar el repositorio del proyecto.
- **Audicity.** Programa de grabación y edición de audio que se usó para editar las pistas de audio y efectos de sonido.
- **Microsoft Visual Studio 2017.** Entorno de desarrollo donde se escribió todo el código del juego.

4.3 Assets

A continuación, se listan los principales assets utilizados en la Tienda de Pepe.

Tabla 4. Listado general de assets de la tienda de Pepe

Asset	Tipo	Descripción u Origen
data.json	Json	Archivo donde se guarda el progreso, el cual es cargado al iniciar el juego.
levelItemsData.json	Json	Define los productos disponibles por nivel. Los cuales son cargados al iniciar el mismo.
Berlin Sans.ttf	Fuente	Tipografía utilizada en todo el juego en sus variaciones (regular, negrilla).
Prefabs de todos los productos y dinero.	Prefab	Los productos son instanciados al iniciar nivel y destruidos cuando se entregan al cliente. El dinero de caja se instancia al iniciar nivel y es destruido cuando se lo lleva el cliente. El dinero del cliente se instancia cuando este muestra la cantidad de dinero que tiene disponible y puede ir a caja o retorna al cliente dependiendo de si la respuesta es correcta o incorrecta.
Main.unity	Escena	Pantalla de inicio del juego.
Levels,unity	Escena	Pantalla donde se escoge nivel.
level1.unity	Escena	Pantalla donde se carga cada uno de los niveles del juego.
CashData.asset	Scriptable Object	Guarda la información que corresponde a cada denominación de dinero.
CustomerPrefabs.asset	Scriptable Object	Guarda los sprites de los clientes.

Asset	Tipo	Descripción u Origen
LevelPoinsImages.asset	Scriptable Object	Guarda los sprites de las puntuaciones que se muestran en la pantalla de escogencia de nivel.
ProductsData.asset	Scriptable Object	Guarda la información que corresponde a cada uno de los productos.
StarsImages.asset	Scriptable Object	Guarda los sprites que se muestran al finalizar los niveles.
BoxColliderResizer.cs	Script	Cambia el tamaño de los colliders del mostrador y de los productos según si existe un ajuste en la resolución.
ButtonData.cs	Script	Cuando los números de la calculadora son presionados, comprueba si existen objetos en el mostrador y envía la información de la tecla al display.
ButtonLevelActivator.cs	Script	En la pantalla niveles, se encarga de activar los botones de nivel según si ya han sido superados o no.
CustomerManager.cs	Script	Controla las entradas y salidas de los clientes. Inicia y finaliza los intentos dentro del nivel.
DragDropScript.cs	Script	Controla el comportamiento de agarrar, arrastrar y soltar.
FindManagers.cs	Script	Encuentra el GameManager y el SoundManager. En general, se usa para asignar los destinos de botones y los sonidos que deben reproducirse.
GameManager.cs	Script	Controlador principal del juego. Gestiona las escenas, los menús, el controlador de sonido.
ILevels.cs	Script	Interfaz para controlar los scripts de cada nivel.
ItemConstructor.cs	Script	Se encarga de llenar el inventario de la tienda, leyendo primero el archivo levelItemsData.json.

Asset	Tipo	Descripción u Origen
Level_0.cs	Script	Código que controla el comportamiento del nivel introductorio (nivel 1).
Level_1.cs	Script	Código que controla el comportamiento del nivel 2.
Level_2.cs	Script	Código que controla el comportamiento del nivel 3.
Level_3.cs	Script	Código que controla el comportamiento del nivel 4.
Level_4.cs	Script	Código que controla el comportamiento del nivel 5.
Level_5.cs	Script	Código que controla uno de los tipos de ejercicios que se propone en el nivel 6.
Level_5b.cs	Script	Código que controla uno de los tipos de ejercicios que se propone en el nivel 6.
Level_6.cs	Script	Código que controla el comportamiento del nivel 7.
Level_7a.cs	Script	Código que controla uno de los tipos de ejercicios que se propone en el nivel 8.
Level_7b.cs	Script	Código que controla uno de los tipos de ejercicios que se propone en el nivel 8.
Level_7c1.cs	Script	Código que controla uno de los tipos de ejercicios que se propone en el nivel 8.
Level_7c2.cs	Script	Código que controla uno de los tipos de ejercicios que se propone en el nivel 8.
Level_7d.cs	Script	Código que controla uno de los tipos de ejercicios que se propone en el nivel 8.
LevelManager.cs	Script	Script muy importante que controla el comportamiento general de cualquiera de los niveles: intentos, aciertos, errores, condiciones de pérdida o victoria, tiempos de espera, cuenta regresiva, comprobaciones de inputs)

Asset	Tipo	Descripción u Origen
LevelPointsIdentifier.cs	Script	En pantalla niveles, hace visibles los mayores puntajes obtenidos por nivel.
Money.cs	Script	Asigna las características a las monedas instanciadas en función de los valores especificados en el respectivo scriptable object
NextLevelActivator.cs	Script	Dentro de la pantalla de juego, en la tienda, habilita del botón para acceder al siguiente nivel, en caso de se haya ganado el actual.
ObjectDetector.cs	Script	Identifica la cantidad y el tipo de producto puesto en el mostrador, para compararlo con el pedido realizado por el cliente. También devuelve los pedidos al inventario o los entrega al cliente según respuesta correcta o incorrecta del jugador.
PriceOutput.cs	Script	Controla lo que se muestra en el display de la “calculadora”.
Product.cs	Script	Script que asigna a los productos los valores definidos en el respectivo scriptable object. Guarda las posiciones iniciales de los objetos para que estos puedan volver a su lugar original cuando son mal puestos o cuando el jugador responde incorrectamente.
SaveSystem.cs	Script	Gestiona el sistema de guardado. Solo se graba un puntaje si es superior al anterior, cuando se gana nivel. Al comienzo del juego, este script además se ocupa de cargar los resultados obtenidos por el jugador en sesiones de juego previas.
SoundManager.cs	Script	Reúne todos los audioclips del juego y tiene los métodos que otros scripts referencian para reproducirlos.
TutorialManager.cs	Script	Controla el comportamiento del tutorial.

Asset	Tipo	Descripción u Origen
Imágenes: fondos (pantalla inicio, pantalla niveles, pantalla juego), mesón, estante, productos, calculadora, dinero, banners (créditos, pausa, ganaste, perdiste), botones (empezar, quit, créditos, siguiente, volver, salir de menú, estrellas, acceso a niveles, índice, reiniciar nivel, ayuda), candado, reloj, indicadores.	Sprites	Los sprites fueron creados por AliceGames específicamente para el proyecto la Tienda de Pepe, https://agpublicistas.com/
Clientes	Sprites	Imágenes creadas por Freepik: Colorful collection with great variety of avatars https://www.freepik.com/free-vector/colorful-collection-with-great-variety-avatars_1258263.htm
Sountrack opening.	Música	Fluffing a Duck by Kevin MacLeod Link: https://incompetech.filmmusic.io/song/3766-fluffing-a-duck License: http://creativecommons.org/licenses/by/4.0/
Soundtrack en pantalla de juego.	Música	The Show Must Be Go by Kevin MacLeod Link: https://incompetech.filmmusic.io/song/4509-the-show-must-be-go License: http://creativecommons.org/licenses/by/4.0/ La sección reproducida es un extracto del original.
sfx_drop	Efecto de sonido	Leather Bag Drop Gneube https://freesound.org/people/gneube/sounds/315838/
sfx_beep1	Efecto de sonido	microwave beep.wav KeyKrusher https://freesound.org/people/KeyKrusher/sounds/154953/

Asset	Tipo	Descripción u Origen
sfx_beep2	Efecto de sonido	Door chime Azumarill https://freesound.org/people/azumarill/sounds/395213/
sfx_lose	Efecto de sonido	lose.wav milton https://freesound.org/people/milton./sounds/86865/
sfx_ok2	Efecto de sonido	Powerup/success.wav GabrielAraujo https://freesound.org/people/GabrielAraujo/sounds/242501/
sfx_ok	Efecto de sonido	Success Jingle JustInvoke https://freesound.org/people/JustInvoke/sounds/446111/
sfx_drag	Efecto de sonido	Pick-1.wav Free-Rush https://freesound.org/people/Free-Rush/sounds/336992/
sfx_register	Efecto de sonido	Cash Register Purchase Zott820 https://freesound.org/people/Zott820/sounds/209578/
sfx_select	Efecto de sonido	Select menú icyjim https://freesound.org/people/icyjim/sounds/436884/
sfx_select_Level	Efecto de sonido	success.wav Grunz https://freesound.org/people/grunz/sounds/109662/
sfx_key	Efecto de sonido	t key uEffects https://freesound.org/people/uEffects/sounds/180970/

Asset	Tipo	Descripción u Origen
sfx_ui_button	Efecto de sonido	Selection Sound notyermom https://freesound.org/people/notyermom/sounds/434833/
sfx_win	Efecto de sonido	game-win.mp3 Mickleness https://freesound.org/people/mickleness/sounds/269198/
sfx_wrong	Efecto de sonido	Error.wav Deathscyp https://freesound.org/people/Deathscyp/sounds/404021/

4.4 Flujo del juego

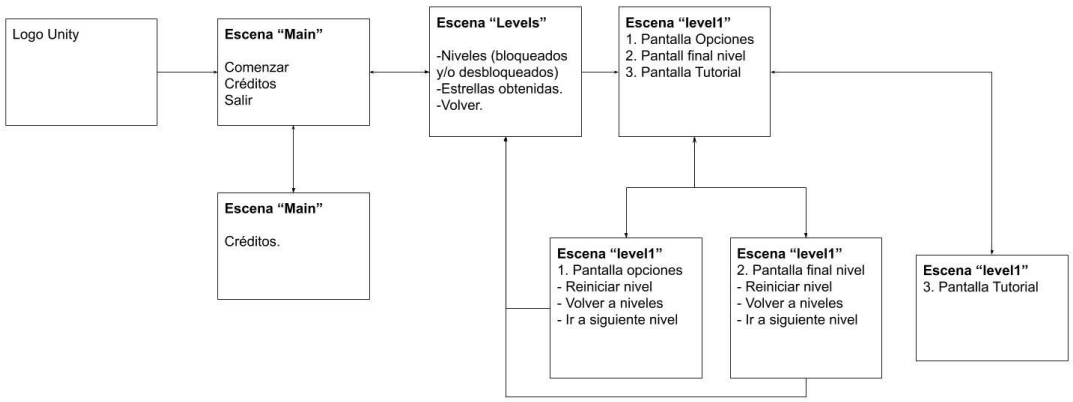

Figura 15. Flujo del juego

La Tienda de Pepe solamente consta de tres escenas: "Main", "Levels" y "level1". Main es la escena de inicio. Desde esta es posible ir a la escena Levels, ver los créditos o salir del juego.

Figura 16. Escena "Main"

Desde la escena Levels se puede acceder a los niveles desbloqueados, aquellos cuyo nivel previo ya ha sido aprobado con 0 a 3 puntos. El único nivel que aparece desbloqueado por defecto es el primero. Desde Levels también se puede volver a la escena Main.

Figura 17. Escena "Levels"

Todos los niveles del juego se cargan en la escena level1. Lo que cambia por nivel es el tipo de ejercicio que se propone al usuario. Esta es la escena donde ocurren las principales interacciones del jugador: entrega de productos e indicación de la respuesta las preguntas que realizan los clientes.

Figura 18. Escena "level1".

Al interior del nivel de juego en la escena level1, se puede abrir la Pantalla Opciones, que brinda la posibilidad de volver a la escena Levels, reiniciar el nivel o ir al siguiente nivel (solo si el nivel actual ya ha sido aprobado). Al desactivar esta pantalla se sigue jugando.

Figura 19. Pantalla pausa

Al final del nivel, aparece la Pantalla Final Nivel para indicarle al jugador si ganó o perdió el nivel y qué puntuación obtuvo. También se brindan las opciones de reiniciar nivel, volver a la escena Levels o ir al siguiente nivel (solamente si el nivel actual ya se ha ganado).

Figura 20. Pantalla de final de nivel

Desde la escena level1 también se puede abrir la pantalla de ayuda, donde están la secuencia de imágenes y las instrucciones que dicen cómo se juega la Tienda de Pepe. Al cerrar esta pantalla, se retorna al nivel de juego.

4.5 Descripción general del funcionamiento técnico del juego

En general, la posición y el tamaño de la mayoría de elementos se adaptan al tamaño de la pantalla, esto pensando en que el juego pueda visualizarse bien en diferentes resoluciones.

Se ha creado un sistema de guardado que salva los puntajes obtenidos por el jugador en los diferentes niveles. La información se guarda en un archivo Json. Solo se salva la mayor puntuación obtenida por nivel. En el momento el juego está configurado para guardar los datos de un solo usuario. Pero, con la estructura de datos creada, se podría tener más usuarios, que podrían crearse y cargarse en una pantalla posterior a la de inicio y anterior a la de niveles.

Solamente se puede avanzar al siguiente nivel cuando el previo ha sido aprobado. El acceso al siguiente nivel puede ser desde la escena Levels o de desde la escena level1, tanto en la Pantalla Opciones como en la Pantalla final nivel. En la escena Levels se muestra la puntuación máxima obtenida por nivel. Al final de cada nivel también se indica que puntuación se obtuvo.

Para implementar los niveles, se utilizaron interfaces. Se recurre a unos métodos que cumplen propósitos similares, aunque en contenido sean diferentes. Es decir, en cada nivel ocurren eventos semejantes, como la entrada y salida de personajes y la configuración de operaciones, pero, la mecánica de cada nivel en general es diferente, unas veces se tendrán que configurar ejercicios para que el jugador sume, y en otras ocasiones para que reste o haga operaciones mixtas.

El inventario y la caja de la tienda se arma partiendo de un archivo json, que sirve para determinar qué objetos de la tienda se muestran por nivel (dinero y productos). Esto posibilita realizar una progresión paulatina de la cantidad de ítems que aparecerán entre niveles. Adicionalmente, se utilizaron scriptable objects para configurar la información de los objetos que componen la tienda (dinero y productos) (cantidad, nombre, valor, sprite) y para los clientes (sprites).

En los niveles creados, por cada intento se escoge aleatoriamente un sprite de cliente que ingresa a la tienda y hace un pedido. Los ejercicios propuestos al jugador no son preestablecidos, sino que se crearon algoritmos para proponer al jugador diferentes valores en cada intento. El juego asigna el monto máximo de la transacción y busca en el inventario disponible los productos que se adapten a la misma, y si no los encuentra, vuelve a realizar una nueva búsqueda hasta que haya una propuesta de ejercicio para el usuario.

El jugador puede agarrar objetos de la estantería para ponerlos en el mostrador. Cuando los objetos no se ponen sobre el mostrador, estos retornan al inventario. Esta posición se actualiza si existe un cambio de resolución de la pantalla, de tal modo que los objetos no vuelvan a un lugar equivocado. Los objetos también retornan a la estantería cuando el cliente se va insatisfecho, sea porque no tiene más tiempo de espera o porque se responde incorrectamente el problema. Si la solución del problema es correcta, los objetos se entregan al cliente y luego son destruidos.

El dinero de caja se crea al iniciar nivel y el dinero del cliente es instanciado cuando el cliente muestra cuánta plata tiene. Esto último pasa especialmente en algunos ejercicios de resta. El dinero de caja se desplaza hacia el cliente cuando se entregan vueltos. El dinero del cliente se puede desplazar al mostrador o a la caja. En algunos ejercicios de resta, se desplaza a mostrador para enseñar al jugador cuánta plata tiene (esta información aparece en texto y en imagen). Si se responde incorrectamente al ejercicio, el dinero vuelve al cliente. Si finalmente se responde correctamente, el dinero del cliente, en cualquiera de los ejercicios, se desplaza a la caja.

Se escribió un script que permite detectar si se ponen correctamente en el mostrador los objetos que pide el cliente (ObjectDetectos.cs). Este script realiza varias comparaciones para brindar al jugador diferente feedback. Se comprueba, por ejemplo, si no hay productos en el mostrador, si hay más o menos objetos, si la cantidad de productos es correcta pero no está bien configurada. Si la configuración de los objetos está bien realizada, entonces el cliente procede a realizar la pregunta problema (por ejemplo, “¿Cuánto debo?”, “¿Cuánto falta?” “¿Cuánto sobra?”).

El jugador puede utilizar la “calculadora” para indicar su respuesta. Dentro del código de cada nivel se determina cuál debería ser la respuesta correcta en función de los objetos pedidos y del dinero asignado al cliente (esto último especialmente en casos de resta). Si no se atiende al cliente en el tiempo límite y si se cobra más de lo debido o se brindan menos vueltos de lo que eran, el cliente se va insatisfecho, se aumenta el indicador de fallas (cara triste), se reproduce sonido de error, y se brinda cliente el feedback que aplique (“Estás cobrando más”, “Son más vueltos”, “Tengo que irme”). También se detectan respuestas incorrectas que no afectarían la satisfacción del cliente mientras este

aún tiene tiempo de espera, por ejemplo, si se devuelve más dinero del correcto o si se cobra menos de lo debido. Cuando se responde correctamente, el cliente brinda las gracias, se va satisfecho, aumenta el indicador de felicidad, se reproduce sonido de logro y, si aplica, se eleva el posible montó máximo de la transacción.

4.6 Arquitectura

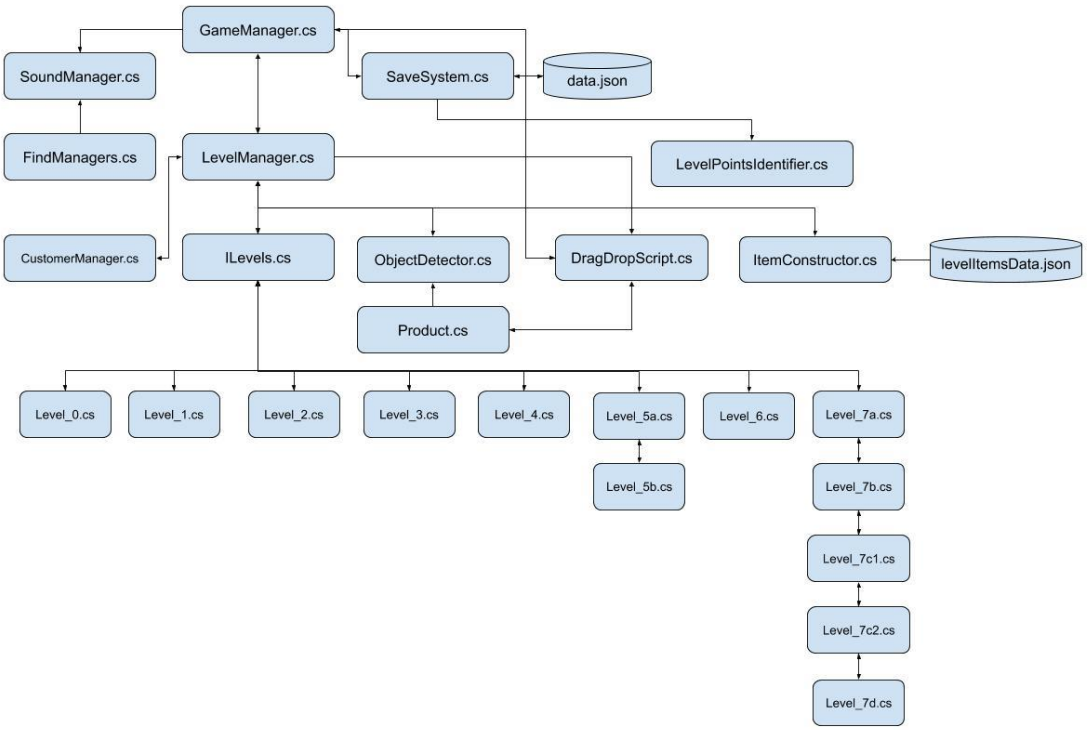

Figura 21. Relación entre scripts.

GameManager es el gestor general del juego. Persiste en todas las escenas y posee algunas variables estáticas que controlan el comportamiento de otros scripts, como el DragDropScript (que define la funcionalidad del mouse dentro del nivel). En ocasiones, el GameManager es el canal para indicarle al SoundManager (gestor de audio) qué sonidos reproducir. También es el canal para acceder al SaveSystem, el script que controla la carga y guardado de progreso y el que informa los puntajes al LevelPointsIdentifier con el fin de determinar qué niveles se activan según los resultados.

SoundManager también recibe instrucciones de reproducción de audio desde FindManagers, que se asigna a objetos de UI y busca los gestores para realizar las acciones requeridas (cargar nivel o agregar un listener).

GameManager le informa al LevelManager el estado de algunas de sus variables para que este último se comporte de la manera que corresponda. Por ejemplo al seleccionar nivel, esta información queda registrada en el GameManager y es consultada por el LevelManager para activar el script del nivel que se necesite. El LevelManager también utiliza al GameManager como canal para indicar que sonidos reproducir al SoundManager o para acceder SaveSystem.

LevelManager tiene una relación muy estrecha con el CustomerManager (que controla la entrada y salida de los clientes y administra el inicio o final de los intentos) y con cada uno de los scripts que define los ejercicios de los niveles. Existen acciones de lectura y escritura entre estos scripts. Por ejemplo, al inicio de nivel, el LevelManager le indica al CustomerManager que inicie la entrada del cliente a la tienda, y una vez el cliente está adentro, el CustomerManager le dice al LevelManager que inicie el conteo regresivo y accede hasta el script de nivel para que establezca el monto de la ronda. De manera similar, el script de nivel, utiliza el LevelManager como canal para indicarle al CustomerManager que ya es momento de sacar al cliente, para dar paso a un nuevo intento, si aún el nivel no ha finalizado.

ObjectDetector tiene como misión informarle al LevelManager y a los scripts de nivel cuáles son los objetos que se encuentran en el mostrador. Especialmente con el fin de poder realizar las comparaciones que permiten saber si los objetos puestos en el mostrador son los correctos. Pero, también recibe órdenes, cuando se trata de entregar los productos al cliente o de regresarlos a la estantería.

Product es un script asociado a cada producto y brinda información a ObjectDetector, DragDrop y a los scripts de nivel para poder controlar su comportamiento. Por ejemplo, permite verificar aspectos como ¿Qué objeto está en el mostrador? ¿El objeto está siendo arrastrado? ¿El objeto ya fue escogido por el cliente?

DragDrop controla el funcionamiento del mouse dentro de los niveles. Para funcionar, necesita información de diversos scripts. Utiliza el LevelManager como canal para leer el CustomerManager y saber si el cliente está fuera o dentro de la tienda, lee el GameManager para saber si está activo algún menú que inhabilite la posibilidad de agarrar y arrastrar objetos, también utiliza el GameManager como canal para llegar a SoundManager y reproducir sonidos de agarrar y soltar y escribe en Product para cambiar el estado del objeto si está siendo agarrado.

ItemConstructor lee el nivel escogido de LevelManager para saber qué productos deben instanciarse. ItemConstructor también utiliza a LevelManager como canal para indicarle a ObjectDetector a dónde deben regresar los objetos en caso de respuesta incorrecta del jugador.

Dentro de los scripts de nivel (Por ejemplo, Level_0.cs, Level_1.cs, etc.), se definen los ejercicios a proponer al usuario. El diseño de estos niveles se describirá con más detalle en el siguiente capítulo.

5. Diseño de niveles

El juego tiene 8 niveles, cada nivel es un día de trabajo. La dificultad aumenta progresivamente, de tal manera que el primer nivel es más fácil que el último. Inicialmente solo se proponen ejercicios de suma o resta. Posteriormente aparecen ejercicios que combinan las operaciones. A continuación, se describe el diseño de niveles tanto desde su estructura técnica como desde su jugabilidad.

Nivel 1: Introducción a la Tienda de Pepe

Al comienzo del juego, cuando aún no se ha ganado el primer nivel, aparece un tutorial que indica cómo jugar la Tienda de Pepe.

Figura 22. Tutorial

En este nivel el jugador solamente debe poner un objeto en el mostrador y decir cuál es el valor correspondiente. El monto máximo de las transacciones es 20. Aparecen la mitad de los objetos que puede tener la estantería. Solo se requieren 4 intentos correctos para ganar y hasta 5 errores para perder. El tiempo además es mucho mayor que en los demás niveles (60 segundos). Es un nivel diseñado para no ser perdido, su función es principalmente introductoria.

Figura 23. Nivel 1 en acción.

Por cada intento sucede lo siguiente:

1. El cliente entra a la tienda.
2. El contrarreloj empieza a contar. Se detiene cuando algún menú es activado (opciones / ayuda).
3. Se define el monto de la ronda.
4. Se define el número de objetos a buscar, que es uno.
5. Se debe buscar un objeto cuyo monto sea menor o igual que el monto de la ronda. El valor del objeto es guardado.
6. Se muestra al jugador el objeto encontrado.
7. El jugador debería poner en el mostrador el objeto correcto. Tan pronto como el producto entra o sale del mostrador, este es comparado con el objeto antes encontrado y se brinda el respectivo feedback.
8. Si se ingresa el objeto correcto, el cliente pregunta cuánto debe.
9. El jugador deberá ingresar un valor y confirmar.
10. Se comprueba si el valor ingresado es igual al valor del objeto antes encontrado. Según la respuesta, suceden diversas consecuencias.
    a. En caso de éxito: el cliente agradece y se retira, aumenta el indicador de intentos correctos en nivel, el producto escogido se dirige al cliente, los sonidos de intento correcto y de caja son reproducidos, se instancia dinero en función del valor del producto y este se desplaza hacia la caja. Si aplica, se eleva el monto de la ronda.
    b. En caso de error (si se cobra de más): el cliente se retira diciendo al jugador que se está cobrando más, aumenta el indicador de fallas, se reproduce el sonido de error y el producto vuelve a la estantería.
    c. En caso de cobrar menos de lo correcto y cuando aún hay tiempo de espera: el cliente informa que se está cobrando menos, se resta uno a los puntos por nivel. Los indicadores de éxito o error permanecen estables (cara feliz o triste). El jugador tiene la oportunidad de borrar la respuesta dada e ingresar otra nueva.
11. Cuando el tiempo acaba, se evalúa el intento en función de lo realizado. Si el objeto puesto en mostrador y el valor ingresado son correctos, ocurre la consecuencia descrita en el paso 10.a. De lo contrario, el cliente se retira, aumenta el indicador de fallas, se reproduce el sonido de error y el producto vuelve a la estantería.

De acuerdo con los aciertos y desaciertos, al final del nivel se muestra si el jugador gana o pierde nivel, enseñando los puntos obtenidos.

Nivel 2: Suma

En este nivel, el jugador debe poner en el mostrador tantos objetos como pida el cliente y luego debe indicar cuánto suman todos sus valores. El monto máximo de las transacciones es 99. Todos los objetos de la estantería ya están habilitados. Se requieren 8 intentos correctos para ganar y 5 para perder. El tiempo de espera del cliente es 30 segundos. El nivel de dificultad es mayor porque deben llevarse más objetos al mostrador y porque deben realizarse sumas cuyos montos pueden ser mayores, comparados con los del nivel 1.

Figura 24. Pedido de cliente en Nivel 2

Por cada intento sucede lo siguiente:

1. El cliente entra a la tienda.
2. El contrarreloj empieza a contar. Se detiene cuando algún menú es activado (opciones / ayuda).
3. Se define el monto de la ronda.
4. Se define aleatoriamente el número de objetos a buscar, entre 2 y 10.
5. Se debe buscar la cantidad de objetos definida cuya suma sea menor o igual que el monto de la ronda. El valor de la suma de los objetos es guardado.
6. Se muestran al jugador los objetos encontrados.
7. El jugador debería poner en el mostrador los objetos pedidos por el cliente. Cuando se pone la cantidad correcta de objetos en el mostrador, se compara si los objetos puestos son los solicitados, para brindar el respectivo feedback.
8. Si se ingresan los objetos correctos el cliente pregunta cuánto debe.
9. El jugador deberá ingresar un valor y confirmar.
10. Se comprueba si el valor ingresado es igual a la suma de los precios de los objetos antes encontrados. Según la respuesta, suceden diversas consecuencias.
    a. En caso de éxito: el cliente agradece mientras se retira, aumenta el indicador de intentos correctos en nivel, los productos se dirigen al cliente, los sonidos de intento correcto y de caja son reproducidos, se instancia dinero en función del valor de los productos y estos se desplaza hacia la caja. Si aplica, se eleva el monto de la ronda.
    b. En caso de error (si se cobra de más): el cliente se retira diciendo al jugador que se está cobrando más, aumenta el indicador de fallas, se reproduce el sonido de error y los productos vuelven a la estantería.

c. En caso de cobrar menos de lo correcto y cuando aún hay tiempo de espera: el cliente informa que se está cobrando menos, se resta uno a los puntos por nivel. Los indicadores de éxito o error permanecen estables (cara feliz o triste). El jugador tiene la oportunidad de borrar la respuesta dada e ingresar otra nueva.

11. Cuando el tiempo acaba, se evalúa el intento en función de lo realizado. Si los objetos y el valor ingresado son los correctos, ocurre la consecuencia descrita en el paso 10.a. De lo contrario, el cliente se retira, aumenta el indicador de fallas, se reproduce el sonido de error y los productos vuelven a la estantería.

De acuerdo con los aciertos y desaciertos, al final del nivel se muestra si el jugador gana o pierde nivel, enseñando los puntos obtenidos.

Nivel 3: Resta - ¿Cuánto falta?

En este nivel, el jugador debe poner en el mostrador el objeto que pida el cliente y luego debe indicar cuánto dinero falta para completar el precio del producto escogido. El monto máximo de las transacciones es 99. Todos los objetos de la estantería están habilitados. Se requieren 8 intentos correctos para ganar y 5 para perder. El tiempo de espera del cliente es 30 segundos.

Figura 25. Pregunta de cliente en Nivel 3

Por cada intento sucede lo siguiente:

1. El cliente entra a la tienda.
2. El contrarreloj empieza a contar. Se detiene cuando algún menú es activado (opciones / ayuda).
3. Se define el monto de la ronda.
4. Se define la cantidad de objetos a buscar (1).
5. Se debe buscar un objeto cuya suma sea menor o igual que el monto de la ronda. El valor del objeto es guardado.

6. Se muestra al jugador el objeto encontrado.
7. El jugador debería poner en el mostrador el producto solicitado por el cliente. Cuando se pone un objeto en el mostrador, se compara si el objeto puesto es el solicitado, para brindar el respectivo feedback.
8. Si se ingresa el objeto correcto, se define el dinero que tiene disponible inicialmente el cliente: este es un valor número natural aleatorio entre el .2 y el .9 del precio del objeto encontrado.
9. Se define el valor que corresponde a la respuesta correcta del ejercicio (la diferencia entre el valor del objeto seleccionado y el dinero disponible del cliente)
10. Se buscan las denominaciones necesarias para luego instanciar el dinero inicial del cliente y ponerlo en el mostrador.
11. El jugador deberá ingresar un valor y confirmar.
12. Se comprueba si el valor ingresado es igual a la diferencia entre precio del producto y dinero inicial del cliente. Según la respuesta, suceden diversas consecuencias.
    a. En caso de éxito: el cliente se retira agradeciendo, aumenta el indicador de intentos correctos en nivel, los productos se dirigen al cliente, los sonidos de intento correcto y de caja son reproducidos, se instancia el dinero que faltaba. Todo el efectivo se dirige a la caja. Si aplica, se eleva el monto de la ronda.
    b. En caso de error (si se cobra de más): el cliente se retira diciendo al jugador que está cobrando más, aumenta el indicador de fallas, los productos vuelven a la estantería, se reproduce el sonido de error y el dinero inicial del cliente vuelve a este.
    c. En caso indicar que falta menos dinero del correcto y cuando aún hay tiempo de espera: el cliente informa que se está cobrando menos, se resta uno a los puntos por nivel. Los indicadores de éxito o error permanecen estables (cara feliz o triste). El jugador tiene la oportunidad de borrar la respuesta dada e ingresar otra nueva.
13. Cuando el tiempo acaba, se evalúa el intento en función de lo realizado. Si los objetos y el valor ingresado son los correctos, ocurre la consecuencia descrita en el paso 12.a. De lo contrario, el cliente se retira, aumenta el indicador de fallas, se reproduce el sonido de error, los productos vuelven a la estantería y el dinero inicial del cliente vuelve a este.

De acuerdo con los aciertos y desaciertos, al final del nivel se muestra si el jugador gana o pierde nivel, enseñando los puntos obtenidos.

Nivel 4: Resta - ¿Cuánto sobra?

En este nivel, el jugador debe poner en el mostrador el objeto que pida el cliente y luego debe indicar cuánto dinero le sobra al cliente. El monto máximo de las transacciones es 99. Todos los objetos de la estantería están habilitados. Se requieren 8 intentos correctos para ganar y 5 para perder. El tiempo de espera del cliente es 30 segundos. Se trata de un nivel de resta con un nivel de dificultad similar al del nivel anterior. Aquí debe hallarse la diferencia entre el dinero de más que tiene el cliente y el precio del producto.

Figura 26. Pregunta de cliente en Nivel 4

Por cada intento sucede lo siguiente:

1. El cliente entra a la tienda.
2. El contrarreloj empieza a contar. Se detiene cuando algún menú es activado (opciones / ayuda).
3. Se define el monto de la ronda.
4. Se define la cantidad de objetos a buscar (1).
5. Se debe buscar un objeto cuya suma sea menor o igual que el monto de la ronda. El valor del objeto es guardado.
6. Se muestra al jugador el objeto encontrado.
7. El jugador debería poner en el mostrador el producto solicitado por el cliente. Cuando se pone un objeto en el mostrador, se compara si el objeto puesto es el solicitado, para brindar el respectivo feedback.
8. Si se ingresa el objeto correcto, se define el dinero que tiene disponible inicialmente el cliente: este es un número natural aleatorio mayor o igual que el precio del objeto seleccionado y menor o igual que el monto de la ronda.
9. Se define el valor que corresponde a la respuesta correcta del ejercicio (la diferencia entre dinero inicial del cliente y el valor del objeto pedido por este).
10. El dinero inicial del cliente es buscado, definido, instanciado y puesto en el mostrador.
11. El jugador deberá ingresar un valor y confirmar.
12. Se comprueba si el valor ingresado es igual a la diferencia entre el dinero inicial del cliente y el precio del producto. Según la respuesta, suceden diversas consecuencias.
    a. En caso de éxito: el cliente se retira agradeciendo, aumenta el indicador de intentos correctos en nivel, los productos se dirigen al cliente, los sonidos de intento correcto y de caja son reproducidos, se instancia el dinero que sobra y se le entrega al cliente. El

efectivo que estaba en el mostrador entra a la caja. Si aplica, se eleva el monto de la ronda.

b. En caso de error (si se dice que sobra menos): el cliente se retira diciendo al jugador que está dando menos vueltos, aumenta el indicador de fallas, los productos vuelven a la estantería, se reproduce el sonido de error y el dinero inicial del cliente vuelve a este.
c. En caso de indicar que sobraba más dinero que el correcto y cuando aún hay tiempo de espera: el cliente informa que se sobra menos dinero, se resta uno a los puntos por nivel. Los indicadores de éxito o error permanecen estables (cara feliz o triste). El jugador tiene la oportunidad de borrar la respuesta dada e ingresar otra nueva.

13. Cuando el tiempo acaba, se evalúa el intento en función de lo realizado. Si los objetos y el valor ingresado son los correctos, ocurre la consecuencia descrita en el paso 12.a. De lo contrario, el cliente se retira, aumenta el indicador de fallas, los productos vuelven a la estantería, se reproduce el sonido de error y el dinero inicial del cliente vuelve a este.

De acuerdo con los aciertos y desaciertos, al final del nivel se muestra si el jugador gana o pierde nivel, enseñando los puntos obtenidos.

Nivel 5: Suma y Resta – “Tengo menos dinero”

En este nivel, el jugador debe poner en el mostrador los objetos que pide el cliente e indicar el valor total de la compra. Luego, el cliente se da cuenta que tiene menos dinero del solicitado, por lo que pedirá al jugador que retire uno o varios objetos. Después de quitar los objetos que el cliente ya no quiere, este preguntará cuánto debe. Para resolver el ejercicio, el jugador debe realizar la suma inicial, y cuando el cliente pide retirar productos, puede elegir entre restar el valor de los objetos escogidos al monto inicial o sumar el valor de los objetos que quedan en el mostrador. El monto máximo de las transacciones es 99. Todos los objetos de la estantería están habilitados. Se requieren 8 intentos correctos para ganar y 5 para perder. El tiempo de espera del cliente es 45 segundos, que es mayor que los niveles anteriores (excepto por el Nivel 1), dado que el nivel requiere más microgestiones, como devolver los productos a la estantería. Esto último podría hacerse simplemente retirando los objetos del mostrador o llevándolos hasta la estantería. Es un nivel con mayor dificultad que los anteriores, en tanto se requiere más movimiento de varios productos y se combinan sumas y restas con más de un objeto.

Figura 27. Pregunta de cliente en Nivel 5

Por cada intento sucede lo siguiente:

1. El cliente entra a la tienda.
2. El contrarreloj empieza a contar. Se detiene cuando algún menú es activado (opciones / ayuda).
3. Se define el monto de la ronda.
4. Se define aleatoriamente el número de objetos a buscar, entre 2 y 10.
5. Se debe buscar la cantidad de objetos definida cuya suma sea menor o igual que el monto de la ronda. El valor de la suma de los objetos es guardado.
6. Se muestran al jugador los objetos encontrados.
7. El jugador debería poner en el mostrador los objetos pedidos por el cliente. Cuando se pone la cantidad correcta de objetos en el mostrador, se compara si los objetos puestos son los solicitados, para brindar el respectivo feedback.
8. Si se ingresan los objetos correctos el cliente pregunta cuánto debe.
9. El jugador deberá ingresar un valor y confirmar.
10. Se comprueba si el valor ingresado es igual a la suma de los precios de los objetos antes encontrados. Según la respuesta, suceden diversas consecuencias.
    a. En caso de respuesta correcta.
        i. El cliente dice que tiene menos dinero.
        ii. Se borra el precio puesto en el display de la calculadora.
        iii. Se define un número de objetos a sustraer del pedido.
        iv. Se buscan los objetos a sustraer.
        v. Se pone una X encima de los objetos a sustraer.
        vi. Se calcula el nuevo total de la compra (sin los precios de los objetos quitados).
    b. En caso de error (si se cobra de más): el cliente se retira diciendo al jugador que se está cobrando más, se reproduce el sonido de error, aumenta el indicador de fallas y los productos vuelven a la estantería.
    c. En caso de cobrar menos de lo correcto y cuando aún hay tiempo de espera: el cliente informa que se está cobrando menos, se resta uno a los puntos por nivel. Los indicadores de éxito o error permanecen estables (cara feliz o triste). El jugador tiene la oportunidad de borrar la respuesta dada e ingresar otra nueva.
11. El jugador debe quitar del mostrador los objetos señalados por el cliente. Cuando queda en el mostrador la cantidad correcta de objetos, se compara si los objetos puestos son los solicitados, para brindar el respectivo feedback.
12. Si los objetos son los correctos, el cliente pregunta cuánto debe.
13. El jugador debe ingresar un valor y confirmar.
14. Se comprueba si el valor ingresado es igual a la suma de los precios que quedaron en el mostrador. Según la respuesta, suceden diversas consecuencias.
    a. En caso de éxito: el cliente agradece mientras se retira, aumenta el indicador de intentos correctos en nivel, los productos se dirigen

al cliente, los sonidos de intento correcto y de caja son reproducidos, se instancia dinero en función del valor de los productos y estos se desplaza hacia la caja. Si aplica, se eleva el monto de la ronda.

b. En caso de error (si se cobra de más) sucede la consecuencia señalada en 10b.
c. En caso de cobrar menos de lo correcto y cuando aún hay tiempo de espera sucede lo señalado en 10c.

15. Cuando el tiempo acaba, se evalúa el intento en función de lo realizado. Cuando ya se han retirado los objetos que señaló el cliente y se tiene en el mostrador los objetos correctos ingresando la respuesta correcta, ocurre la consecuencia descrita en el paso 14.a. De lo contrario, el cliente se retira, se reproduce el sonido de error, aumenta el indicador de intentos fallidos y los productos vuelven a la estantería.

De acuerdo con los aciertos y desaciertos, al final del nivel se muestra si el jugador gana o pierde nivel, enseñando los puntos obtenidos.

Nivel 6: Sumas y restas - ¿Cuánto debo? / ¿Cuánto falta?

En este nivel, el jugador debe poner en el mostrador los objetos que pide el cliente y luego debe indicar cuánto dinero le falta o le sobra a este. El monto máximo de las transacciones es 99. Todos los objetos de la estantería están habilitados. Se requieren 8 intentos correctos para ganar y 5 para perder. El tiempo de espera del cliente es 30 segundos.

Se trata de un nivel donde se proponen ejercicios cuya resolución demanda operaciones de suma y resta. Es un nivel con un grado de dificultad similar al anterior, que requiere menos microgestión, pero reúne y complejiza los tipos de ejercicios propuestos en los Niveles 3 y 4, en tanto no se pide un solo objeto sino varios, entonces el jugador tendrá que sumar los valores de estos para poder hallar cuánto dinero tiene de más o menos. Se inicia el nivel siempre con un ejercicio que indaga cuánto dinero falta. Al final del intento, al azar se sortea el tipo de ejercicio que sigue, si toca hallar el dinero que falta o sobra. La secuencia lógica es similar a la de los niveles 2 y 3, lo que varía principalmente es el número de objetos pedidos por el cliente, que no será 1, sino 2 hasta 10 objetos.

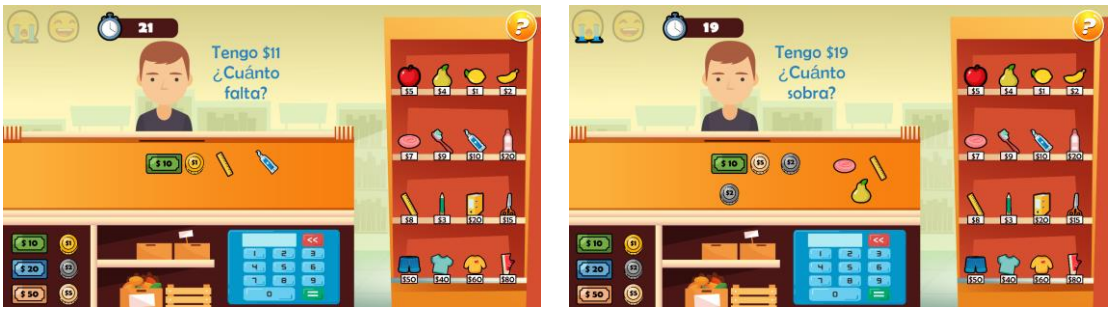

Figura 28. Nivel 6

Nivel 7: Introducción a la multiplicación

En este nivel, el jugador debe poner en el mostrador tantos objetos como pida el cliente y luego debe indicar cuánto suman todos sus valores. Este nivel funciona como una introducción a la multiplicación (cantidades iguales sumadas tantas veces), en el sentido de que se pide un mismo tipo de objeto en cantidades que van de 2 hasta 10, mientras su suma no supere el monto de la ronda. El monto máximo de las transacciones es 99. Todos los objetos de la estantería están habilitados. Se requieren 8 intentos correctos para ganar y 5 para perder. El tiempo de espera del cliente es 30 segundos. Tiene un nivel de dificultad similar al Nivel 2. La variación principal está en la forma de proponer el ejercicio que podría realizando multiplicaciones mentales.

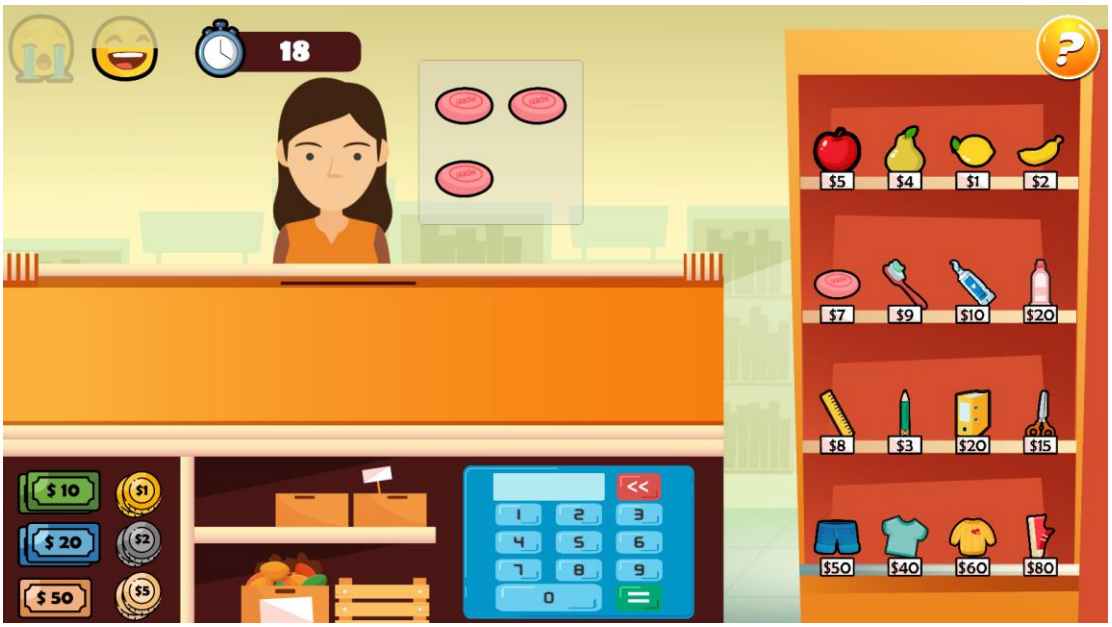

Figura 29. Nivel 7

Por cada intento sucede lo siguiente:

1. El cliente entra a la tienda.
2. El contrarreloj empieza a contar. Se detiene cuando algún menú es activado (opciones / ayuda).
3. Se define el monto de la ronda.
4. Aleatoriamente se define qué producto buscar.
5. Se comprueba si el precio del objeto encontrado es menor que el monto de la ronda y si tampoco sobrepasa este monto al multiplicarlo por 2.
6. Si se supera el monto máximo de ronda, continúa la búsqueda.
7. Si se haya un objeto cuyo precio es menor que el monto de la ronda, así sea multiplicado por 2, se verifica si el producto está disponible en el inventario. Si al menos hay 2 ejemplares del mismo, el producto puede ser escogido.
8. Si no existen suficientes ejemplares, entonces se reinicia la búsqueda.
9. Si existen suficientes ejemplares, se busca un factor por el cual multiplicar el precio del producto. Este factor es encontrado entre la cantidad de ejemplares disponibles del producto. Si la multiplicación del precio del

producto por el factor hallado sobrepasa el monto de ronda, se vuelve a hacer la búsqueda de otro factor hasta encontrarlo.

10. Cuando ya se ha encontrado un factor por el cual multiplicar el precio del producto seleccionado sin sobrepasar el monto de la ronda, entonces se guarda el resultado de esta multiplicación, dado que este será el valor correcto que se comparará con la respuesta que después brindará el usuario.
11. El cliente pide el objeto encontrado tantas veces como el factor definido en el paso 10.
12. El jugador debería poner el objeto escogido en las cantidades pedidas por el cliente. Cuando se pone la cantidad correcta de objetos en el mostrador, se compara si los objetos puestos son los solicitados, para brindar el respectivo feedback.
13. Si se ingresaron los objetos correctos, el cliente pregunta cuánto debe.
14. El jugador deberá ingresar un valor y confirmar.
15. Se comprueba si el valor ingresado es igual al producto de la multiplicación previamente guardada en el paso 10. Según la respuesta, suceden diversas consecuencias.
    a. En caso de éxito: el cliente agradece mientras se retira, aumenta el indicador de intentos correctos en nivel, los productos se dirigen al cliente, los sonidos de intento correcto y de caja son reproducidos, se instancia dinero en función del valor de los productos y estos se desplaza hacia la caja. Si aplica, se eleva el monto de la ronda.
    b. En caso de error (si se cobra de más): el cliente se retira diciendo al jugador que se está cobrando más, aumenta el indicador de fallas, se reproduce el sonido de error y los productos vuelven a la estantería.
    c. En caso de cobrar menos de lo correcto y cuando aún hay tiempo de espera: el cliente informa que se está cobrando menos, se resta uno a los puntos por nivel. Los indicadores de éxito o error permanecen estables (cara feliz o triste). El jugador tiene la oportunidad de borrar la respuesta dada e ingresar otra nueva.
16. Cuando el tiempo acaba, se evalúa el intento en función de lo realizado. Si los objetos y el valor ingresado son los correctos, ocurre la consecuencia descrita en el paso 15.a. De lo contrario, el cliente se retira, aumenta el indicador de fallas, se reproduce el sonido de error y los productos vuelven a la estantería.

De acuerdo con los aciertos y desaciertos, al final del nivel se muestra si el jugador gana o pierde nivel, enseñando los puntos obtenidos.

Nivel 8: Mix de ejercicios

Este es el nivel más difícil del juego y también el más divertido, principalmente porque el jugador debe estar atento al tipo de ejercicio que se propone. Todos los niveles previos han sido una preparación para este nivel. Aquí se proponen ejercicios de los niveles anteriores: sumas de los precios de múltiples objetos (Nivel 2), sumas de los precios mismos objetos (Nivel 7), hallar el valor que sobra o falta con múltiples objetos (Nivel 6), agregar los productos indicados, indicar su suma y luego retirar los que pida el cliente para encontrar la nueva sumatoria

de la compra (Nivel 5). Solo para el ejercicio que implicar quitar objetos (como el tipo de problema propuesto en el Nivel 5), el tiempo de ronda es de 45 segundos. El tiempo para los otros ejercicios es de 30 segundos. La cantidad de objetos que pide el cliente varía entre 2 y 10 productos. El monto máximo de las transacciones es 99. Todos los objetos de la estantería están habilitados. Se requieren 8 intentos correctos para ganar y 5 para perder.

El nivel siempre comienza con una suma y al final del intento se elige aleatoriamente qué otro ejercicio de los disponibles puede aparecer. Si aplica, al final del intento se realiza el aumento de monto de ronda. La estructura lógica del intento depende del ejercicio propuesto. Si es una suma, funcionará como el Nivel 2, si se trata de hallar el dinero que falta o sobra, funcionará como el Nivel 6, si se trata de una suma repetida del mismo tipo de objeto, funcionará como el Nivel 7, si primero hay que realizar una suma para luego retirar ciertos objetos, restando su valor, entonces el ejercicio se comportará como el Nivel 5.

6. Manual de usuario

En el momento, el juego está desarrollado para ser jugado en PC, en sistema operativo Windows. Para acceder al mismo se debe extraer todos los ficheros dentro del archivo zip y luego realizar doble clic sobre el ejecutable del juego La Tienda de Pepe.exe.

Todo el juego se controla principalmente con el ratón y la tecla Escape (esta se usa para salir de escenas o cerrar menús). En una versión posterior se habilitará el teclado como mando para jugar. En la Figura 30, se muestran las instrucciones para jugar.

Figura 30. Tutorial completo del juego

7. Conclusiones

El desarrollo proyecto la Tienda de Pepe ha sido un continuo y exigente proceso de aprendizaje sobre diseño y programación de videojuegos, tanto desde el ámbito técnico como personal.

El definir un alcance acotado es lo que permitió desarrollar una versión completa del juego, en el sentido que es funcional (se puede usar sin ayuda) y en general tiene pocos errores identificados que atasquen su jugabilidad o creen lagunas que alteren la experiencia de juego (como atajos que brinden ventajas inesperadas). Esta estrategia de apuntar a algo concreto, pero abordable, fue lo que hizo posible que pudieran programarse ocho niveles en lugar de los cinco propuestos inicialmente. Tuvo que renunciarse a muchas características deseadas.

El proceso de investigación previo a la realización del juego, consultando documentos oficiales sobre la enseñanza de las matemáticas en los primeros años escolares y explorando libros de texto dirigidos a estudiantes de primer grado, así como llevar continuamente un diario de trabajo, donde se anotaban reflexiones, problemas, lista de deseos, entre otros temas, permitió analizar continuamente qué características se implementarían en el juego. Esto fue el rasero para definir qué sería crítico introducir y qué no, en aras de tener un juego finalizado con los conocimientos y recursos temporales disponibles.

Diseñar y desarrollar la Tienda de Pepe permitió vivir en carne propia las diversas fases del proceso de creación de juegos que a menudo se leen en la literatura sobre la materia: la investigación de referentes, la generación de ideas, el

prototipado físico y digital, la escritura y reescritura del concepto del juego, las validaciones continuas para refinar poco a poco el juego.

Todo el proceso permitió adquirir y fortalecer aprendizajes sobre varios temas, como el uso de interfaces para controlar diversos niveles desde una misma escena, la carga y registro de datos, el uso de *scriptable objects*, el manejo de las herramientas de UI de Unity y la creación de lógicas que simulen, aunque básicamente, las interacciones en escenarios reales.

La búsqueda de bugs es un proceso muy dispendioso, riguroso y minucioso que requiere una cantidad considerable de tiempo que debe planearse. Hay errores que solamente se perciben después de numerosas iteraciones. Cuando se intenta corregir un error, se corre el riesgo de alterar otra funcionalidad del juego. El videojuego es un sistema, y como tal, un cambio en la parte puede afectar el todo, lo que hace que deba probarse todo el sistema una y otra vez hasta cerciorarse de que un ajuste no ha causado un efecto inesperado en otra parte del juego.

En versiones posteriores podrían implementarse muchas mejoras para refinar la experiencia del juego. Desde el punto de vista de las mecánicas, se podrían implementar diversas dificultades, por ejemplo, que varíen aleatoriamente los objetos que aparecen por nivel, así como su posición y precio, disminuyendo también los tiempos de espera. También se podrían agregar servicios en línea para crear un tablero de puntuaciones (a modo de juego tipo arcade) que se reinicie diariamente y les permitan a los jugadores comparar sus desempeños, por ejemplo, en términos de velocidad de sus respuestas. Al final del nivel podría agregarse un panel que indique al jugador cuáles fueron las operaciones donde se cometieron errores, en aras de que pueda mejorar en próximos intentos. Desde el componente estético, podrían agregarse animaciones a las interfaces, a los personajes y efectos con partículas que se reproduzcan cuando los intentos y los niveles son aprobados o fallados. Desde la funcionalidad del juego, podrían implementarse opciones para controlar la música y el sonido, múltiples idiomas, así como una pantalla para gestionar usuarios y sus datos, en el caso de que algún docente quiera usar el juego en sus clases y desee obtener reportes de desempeños. De cara a una expansión del proyecto, también podrían crearse más niveles que aborden otros temas (multiplicación, división, fracciones) y admitan denominaciones mayores de las transacciones.

Es necesario realizar sesiones de playtesting con la población objetivo. Las pruebas realizadas fueron realizadas con adultos, uno de los cuáles era docente de estudiantes de primaria. Aunque el feedback recibido fue muy valioso, es clave analizar el juego siendo jugado por la población objetivo. Por ejemplo, debe validarse si agarrar y arrastrar es un tipo de interacción que resulta sencilla para los niños y las niñas, ya que requiere una mayor psicomotricidad fina que aún está en proceso de desarrollo. Aunque se intentaron elaborar instrucciones y diálogos cortos con lenguaje comprensible, también es necesario validar si las instrucciones escritas son suficientes o si en cambio se requiere añadir consignas, diálogos y retroalimentaciones orales.

Durante el diseño y desarrollo del juego debe decidirse qué funcionalidades finalmente se implementan. Tenía que escogerse entre características críticas y aquellos desarrollos que resultaban muy interesantes y atractivos, pero no

esenciales. Esta toma de decisiones implicó realizar renuncias, siempre cotejando la meta esperada con los recursos disponibles para alcanzarla. A modo de metáfora: “la prioridad siempre fue terminar la carrera en pie, no dejándola a mitad de camino por exceso de perspectiva”.

8. Referencias

AppOnboard (2020). Buildbox. Recuperado de: https://www.buildbox.com/

Arcademics (2020) Jet Ski Addition. Recuperado de: https://www.arcademics.com/games/jet-ski

Becker, K (2018). What’s the difference between serious games, educational games, and game-based learning?. Recuperado en diciembre de 2019 de: http://minkhollow.ca/beckerblog/2018/02/03/whats-the-difference-between-serious-games-educational-games-and-game-based-learning/comment-page-1/

Blue Duck Education Ltd. (2020). Sundae Times Lite. Recuperado de: https://www.mangahigh.com/en/games/sundaetimeslite

Boller, S & Kapp, K. (2017). Play to Learn: Everything you need to know about designing effective learning games. Alexandria: ATD Press.

Chou, Y (2015). Actionable Gamification: Beyond points, badges, and leaderboards. California: Octalysis Media.

Corona Labs Inc (2018). Corona [software]. Recuperado de: https://coronalabs.com/

Csíkszentmihályi, M. (1990). Flow: The psychology of optimal experience. New York: Harper y Row.

Fullerton, T. (2014). Game design workshop (Third edition). Boca Raton: CRC Press Taylor & Francis Group.

Juan Linietsky, Ariel Manzur y contribuidores (2020). Godot [software]. Recuperado de: https://godotengine.org/

JumpStart Games (2018). Mathblaster [videojuego]. Disponible en mayo de 2020 en: http://www.mathblaster.com/

Kahoot DragonBox AS 2020 (2020). *DragonBox Algebra 5+.* Recuperado en mayo de 2020 de: https://dragonbox.com/products/algebra-5

McGonigal, J. (2011). Reality is broken: Why games make us better and how they can change the world. London: Jonathan Cape.

Ministerio de Educación Nacional (2006). Estándares Básicos de Competencias en Lenguaje, Matemáticas, Ciencias y Ciudadanas: Guía sobre lo que los estudiantes deben saber y saber hacer con lo que aprenden. Recuperado en mayo de 2020 de: http://cms.mineducacion.gov.co/static/cache/binaries/articles-340021_recurso_1.pdf?binary_rand=1223

Ministerio de Educación Nacional (2010). Matemáticas 1: Primera Cartilla. Bogotá: Recuperado de: http://redes.colombiaaprende.edu.co/ntg/men/archivos/Referentes_Calidad/Modelos_Flexibles/Escuela_Nueva/Guias_para_estudiantes/MT_Grado01_01.pdf

Ministerio de Educación Nacional y Universidad de Antioquia (2016). Derechos Básicos de Aprendizaje: Matemáticas. Recuperado en mayo de 2020 de: http://aprende.colombiaaprende.edu.co/sites/default/files/naspublic/DBA_Matem%C3%A1ticas.pdf

Photon Storm Ltd (2020). Phaser [software]. Recuperado de: https://phaser.io/

Pope, L. (2013). Papers, Please [videojuego].

Schell, J. (2015). The art of game design: A book of lenses. Boca Raton: CRC Press Taylor & Francis Group.

Scirra Ltd (2020). Construct [software]. Recuperado de: https://www.construct.net/en

Timestables.com (2020). Happy Burger [videojuego]. Recuperado de: https://www.timestables.com/happy-burger.html

Unity Technologies (2020). Unity [software]. Recuperado de: https://unity.com/es

YoYo Games Ltd (2020). GAMEMAKER STUDIO 2 [software]. Recuperado de: https://www.yoyogames.com/gamemaker