Estudio del *Trucotron*. Aportes para la historia del desarrollo de videojuegos en Argentina.

Gustavo Del Dago

Universidad Nacional de José C. Paz
Pcia. de Buenos Aires
Argentina
gdeldago@gmail

Resumen En este trabajo se presentan los primeros resultados sobre el análisis del videojuego *Trucotron* desde la arqueología computacional, teniendo como principal objeto de estudio el código de programa. Así mismo, se describen algunos hallazgos, en particular aquellos producto de la reescritura del código fuente y su posterior análisis. Mediante la exposición de un caso de estudio concreto, se intenta evidenciar el valor potencial de la metodología empleada a fines de producir nuevo conocimiento sobre los objetos digitales preservados, una forma de realizar aportes para la historia de los videojuegos.

Palabras claves: Videojuegos · Preservación · Emulación · Arqueología computacional · MAME

1 Introducción

Una de las plataformas asociadas al origen de la industria de los videojuegos es la máquina recreativa. Una configuración que lejos de ser una novedad del campo computacional, es fiel heredera de otros sistemas de entretenimientos basados en esquemas de operación mediante fichas o monedas. Se trata de un tipo de artefacto concebido para funcionar en locales públicos de entretenimiento. En Argentina, las empresas prestadoras de los servicios de entretenimiento comenzaron incorporando máquinas recreativas importadas. El proceso comercial generó un espacio para la aparición de diversos procesos industriales mediante los cuales se ensayaron intervenciones sobre los equipos y sistemas importados. Si bien no es el propósito de este trabajo analizar el proceso de génesis y desarrollo de la industria nacional de los videojuegos, es importante señalar que los procesos industriales se orientaron a proveer soluciones relacionadas con los aspectos constructivos de los equipos y componentes. Algo que sucedió en menor medida con el desarrollo de programas. Existe gran cantidad de programas cuyas intervenciones demuestran un alto grado de dominio sobre las tecnologías con las que están desarrollados. Un caso excepcional, si se probara que el diseño y desarrollo de la plataforma son de factura nacional, podría ser el que abordamos en este trabajo. Aquí tomaremos como objeto de estudio exclusivamente

los programas que determinan la lógica de funcionamiento de la máquina recreativa *Trucotron*. El abordaje que emplearemos es el que hemos denominado arqueología computacional y que hemos aplicado en trabajos anteriores [4],[5] y [2]. Las técnicas, herramientas y procedimientos empleados son similares a los presentados por *Aycock*[1]. El propósito de este trabajo es presentar los primeros hallazgos producto de este enfoque. El conocimiento generado mediante el proceso aporta, desde nuestra perspectiva, nuevos elementos para la historia de los videojuegos.

2 Objetos preservados y método de trabajo.

El presente trabajo tiene como punto de partida dos objetos principales: el emulador de la plataforma *Trucotron* disponible en el entorno *MAME* y los archivos conteniendo la imagen digital del código almacenado en las memorias ROM de la máquina original. Al conjunto de objetos mencionado se suman las fotografías de las placas de circuito impreso correspondientes a dos máquinas distintas[1] y el testimonio de uno de los creadores del juego publicado en la revista Replay[10]. En el entorno *MAME*, cada máquina emulada se implementa mediante una serie de módulos de programa que, en conjunto, permiten recrear cada plataforma o arquitectura hardware concreta. A dicho conjunto de módulos de programa se lo denomina "driver". El código fuente de cada *driver* constituye un documento que, empleando un lenguaje formal (el lenguaje de programación *C++*), describe de manera precisa la arquitectura de cada máquina emulada. Se trata de un objeto muy fértil que permite lograr niveles de comprensión profundos sobre equipos y dispositivos a los que, por una diversidad de motivos, no se tiene posibilidad de acceder. Es necesario señalar que el conocimiento contenido en el código fuente de cada *driver*, es producto de diversos trabajos de ingeniería inversa realizados, generalmente, sobre los equipos y dispositivos físicos. El objetivo principal del proyecto *MAME* está centrado en la preservación de los equipos y dispositivos. En este sentido, la posibilidad de ejecutar los programas es, en términos de los propios responsables, un efecto secundario[8]. Se trata de un criterio que prioriza el desarrollo de emuladores donde el grado de fidelidad respecto de los equipos físicos constituye una clave. El criterio se aplica aún en los casos donde no se logre, por diversos motivos, ejecutar los programas preservados. Lo anterior nos permite suponer que la forma de documentación a la que nos hemos referido (el código fuente de los drivers) guarda estrecha relación con los equipos y dispositivos emulados, con aquello que denominados "plataforma". Al mismo tiempo, podría constituir un factor limitante cuando se quiera, como en este caso, analizar los programas en tiempo de ejecución. Como se señala en [7], el proyecto *MAME* se focaliza en la preservación de las máquinas recreativas en tanto artefactos culturales significativos. Nuestro análisis, que toma como objeto central a los programas contenidos que determinan la lógica de funcionamiento de las plataformas, tiene la intención de aportar elementos que favorezca el estudio de las producciones culturales que llamamos videojuegos.

[1] Se trata de las placas con números de serie 0222 y 0239.

El código de programa constituye el objeto más importante desde la perspectiva de este trabajo. Partiendo del código binario y mediante un proceso de desensamblado se logró reescribir el código fuente del programa. Se trata de una nueva versión del código fuente que nos permite alcanzar niveles de comprensión suficientes para develar algunas cuestiones y aportar nuevos elementos de estudio. El código fuente resultante, diferente al que escribieron los autores originalmente, permite generar el mismo código binario que se encuentra en las memorias de la máquina. En otras palabras, el código fuente que se ha obtenido y sobre el que se realiza el estudio del programa es una nueva forma de representación del código contenido en la plataforma preservada.

3 Secretos industriales y mecanismos de protección.

La posibilidad de ejecutar los programas en un entorno de emulación nos permite, entre otras cosas, lograr una aproximación desde la perspectiva del usuario final o destinatario. Algo que llama inmediatamente la atención es el despliegue de un extenso texto, presentado en letras rojas sobre fondo verde, que incluye información sobre los derechos de propiedad. El mensaje advierte, al mismo tiempo, sobre las posibles sanciones para quines infrinjan los mencionados derechos. Podemos suponer que este mensaje tiene el propósito de disuadir a potenciales competidores, interpretándolo como un primer testimonio que da cuenta de algunas preocupaciones que tenían los desarrolladores y productores de tecnología en el sector de máquinas recreativas. El aviso legal refiere al juego con el nombre "TRUCOTRON" mientras que el circuito impreso de las placas halladas indican "TRUCO-TRON". Los programadores del *driver* de *MAME* optaron por la denominación "Truco-Tron" una preferencia que puede comprenderse atendiendo al objetivo principal del proyecto *MAME* centrado, como se ha dicho, en la preservación de las plataformas hardware. Aquí, interesados principalmente en los aspectos relacionados con el programa, nos inclinamos por emplear la versión registrada en el código binario (sin el guión medio)[2].

Roberto Fresca, uno los programadores del *driver* de *MAME*, nos comenta las dificultades a la hora de estudiar la arquitectura de la máquina[3]. Los circuitos integrados de las placas a las que tuvo acceso se encontraban rayados de tal modo que resultó imposible la lectura de sus códigos identificatorios. De este modo el trabajo de ingeniería inversa comienza con la identificación, mediante pruebas de caja negra, de cada uno de los componentes. Un factor adicional que dificulta la tarea de análisis y replicación de la plataforma, es la utilización de una memoria RAM estática alimentada mediante una batería. Sobra decir que, cualquier intento de manipulación sobre este componente ocasiona la pérdida de la información en él contenida. El contenido de dicha memoria es validado en distintos momentos durante la ejecución del programa y, en caso de existir diferencias, la máquina asumirá un estado de bloqueo. La solución implementada

[2] A fines de facilitar la lectura preferimos no utilizar letras mayúsculas.

[3] Intercambio de correos electrónicos mantenidos con el autor durante el primer semestre de 2020.

por los programadores del driver de *MAME* consistió en forzar el contenido de las posiciones de memoria con los valores que el programa espera encontrar durante la etapa de verificación. Se trata de una técnica que, aún cuando no permite explicar el modo de funcionamiento del sistema de protección o el propósito del mismo, favorece la ejecución de los programas.

El mecanismo de protección que acabamos de describir tiene algunas consecuencias que vale la pena analizar. Cuando se pierde el contenido de la memoria RAM estática (lo que puede ocurrir cuando la batería que le provee potencia se agota) la máquina queda en un estado de bloqueo. En este trabajo se demostrará que existen otras formas que, aún cuando la batería esté funcionando, la memoria pierda su contenido. La consecuencia directa de ésto es que la máquina tiene un tiempo de operación limitado a la vida útil de la batería, un componente que indefectiblemente dejará de proveer energía. Se trata de una situación que merece ser considerada por diversas cuestiones. En principio nos permite suponer que o bien existía un vínculo activo entre los proveedores de la tecnología y los usuarios (posiblemente implementado mediante un esquema de prestación de servicios) o bien, los usuarios tenían los medios para prestar servicio a las máquinas y conocían la existencia del sistema de protección. Por otra parte, los sistemas de protección de esta naturaleza, hacen que los actuales propietarios de los dispositivos, no estén en condiciones de impedir el bloqueo de los mismos, perdiendo el derecho de continuar utilizándolos. Estamos frente a una situación en la que el sistema deja de operar no por presentar fallas (algo que es imposible de evitar) sino por funcionar de acuerdo a lo programado.

El análisis de código nos ofrece nuevas evidencias sobre los esquemas de protección implementados a nivel programa. Se incorpora, como parte de la secuencia de inicialización, el cálculo de una suma de verificación. Esta suma se realiza sobre el espacio de memoria del programa. El programa se almacena en memorias que, cuando se encuentran instaladas en el dispositivo final (la máquina recreativa propiamente dicha) no pueden ser alteradas. Son dispositivos de sólo lectura. Un dato curioso es que la suma de verificación se computa sobre un espacio cuyo tamaño de palabra es de 16 bits del que sólo se verifican los 8 bits de menor peso. En cualquier caso, la suma de verificación funciona como una barrera cuando se quiere modificar el código de programa o los datos almacenados. Si la verificación no tiene resultado favorable, el equipo se bloquea emitiendo una señal audible de carácter intermitente. La ausencia de un sistema de autodiagnósito, algo muy común en las máquinas recreativas, nos permite suponer que, en este caso, la suma de verificación tiene el propósito de evitar la modificación deliberada de los programas y datos antes que garantizar el correcto funcionamiento del equipo. Una de las modificaciones que dicha suma de verificación puede detectar es la modificación de cualquiera de los textos de la interfaz de usuario. La modificación de los textos, que permite alterar la apariencia del juego, no requiere mayores conocimientos sobre el código de programa, lo que permitiría copiar el programa y modificar, por ejemplo, el nombre del juego y el mensaje con los derechos de propiedad. La suma de verificación actúa en este caso como un mecanismo de seguridad que impide realizar cambios cuando

no se advierte su existencia y modo de funcionamiento. Algo que, como puede suponerse, exige un esfuerzo adicional.

Considerando la temática del juego, basada en la versión argentina del juego de naipes Truco, es dado suponer que los potenciales competidores, a quienes van dirigidos los mensajes y sistemas de protección, se encontraban en la plaza local.

4 Un intento de datación.

De acuerdo a los testimonios recogidos en [10] la primera partida de máquinas *Trucotron* estuvo lista el 10 de septiembre de 1991. La sensación que se experimenta al ejecutar *Trucotron*, cuando se lo compara con los videojuegos más populares, es la de estar frente a un producto de los primeros años ochenta. Tanto los gráficos (resolución y paleta de colores) como el sonido, se asemejan a los de las primeras máquinas recreativas. Es necesario decir que estamos presentando una sensación y, en consecuencia, se trata de algo absolutamente subjetivo. Los intentos por determinar a qué año o época pertenecer un videojuego dependerá de la experiencia y conocimiento del observador. En el proyecto *MAME* se realizó una datación que consideramos contextualizada en la medida que se determinó a partir del conocimiento sobre una gran cantidad de máquinas y plataformas cuyas fechas de producción están confirmadas mediante elementos que no dejan lugar a dudas. De este modo, se logró inferir con alto grado de certeza un periodo temporal. Como puede verse en el código fuente del *driver* de *MAME*, no se arriesgó un año concreto sino que se prefirió emplear un rango, ubicando a la máquina *Trucotron* en la década de los ochenta. Cabe señalar que se trata de una década de desarrollo acelerado en la industria de los videojuegos, en consecuencia las producciones de inicios de los ochenta muestras rasgos y características muy diferentes a los de finales. La información publicada deja claro que se trata de una estimación sin confirmar (198?). Una datación que tiene como principal parámetro el estado del arte del campo de la electrónica, los procesos productivos del sector y la disponibilidad de algunos componentes en el mercado. Aquí presentaremos argumentos para sostener que la tecnología utilizada en el diseño y construcción de la plataforma *Trucotron* obedece a razones de distinta índole y posiblemente complementarias: a) la aplicación de un criterio de suficiencia y b) el dominio que poseían los desarrolladores sobre las herramientas y tecnología empleada en su producción. Para sostener el argumento relacionado con el criterio de suficiencia, debemos decir que la producción final está muy bien lograda cuando se la analiza desde aquello que denominamos experiencia de usuario. Se trata del resultado de una programación que logra sacar provecho de una plataforma que es sin dudas muy modesta cuando se la compara con las máquinas recreativas de los años noventa. Respecto del segundo argumento, recordemos que, de acuerdo al testimonio de uno de los desarrolladores[10], la elaboración del programa comenzó a principio de los años ochenta. De acuerdo al relato, se puede suponer que se trató de un proceso continuo que comenzó a principios de la década del ochenta y concluyó en el año 1991. Sin embargo, atendiendo a la

arquitectura de la máquina recreativa y a algunas características del videojuego podemos afirmar que, en este proceso de desarrollo y producción fue necesario realizar adecuaciones y ampliaciones al programa original. El análisis del código fuente nos acerca un dato que apoya la hipótesis de estar frente a un programa más antiguo que la plataforma de la cuál forma parte en las máquinas *Trucotron*. En el código de programa, los textos de las interfaz de usuario se almacenan empleando un identificador de fin de cadena. Se trata de un método que exige reservar un valor para dicho identificador. En *Trucotron* se emplea el valor $0x04$. Diremos que se trata de una elección curiosa por dos motivos: 1) para el año 1990 la utilización del valor $0x00$ era ya una práctica muy extendida (operando como un estándar de facto) y 2) se trata de un valor que coincide con la utilizada en el código fuente de *monitor de sistema ASSIT09* incluido en el manual del microprocesador *Motorola 6809*[9], el mismo que se emplea en *Trucotron* y que posiblemente tuviera la máquina en la que se desarrolló el programa embrionario del que suponemos se derivó el contenido en la máquina recreativa que estamos analizando. Esto permite suponer que o bien los programadores habían adquirido la costumbre de emplear dicho delimitador y continuaron haciéndolo hasta principios de los años noventa sin asumir las prácticas más difundidas para ese momento o bien los programas, en particular las rutinas orientadas al tratamiento de textos estaban programadas desde años anteriores. En este punto es necesario anticipar que, de acuerdo a los resultados de nuestro análisis, el código binario no fue producido por un compilador de un lenguaje de alto nivel. Algo que se observa en el tratamiento de parámetros durante la invocación de rutinas y en la gestión de los registros del microprocesador: en particular los relacionados con el *stack* y el área de memoria de trabajo. Desde nuestra perspectiva este puede ser un nuevo elemento que apoya el supuesto de estar frente a un programa elaborado originalmente para un sistema de computación hogareño y luego portado a la plataforma de la máquina recreativa.

5 Adecuación al mundo de las máquinas recreativas.

Aceptando que el programa de la plataforma *Trucotron* tiene su origen en un programa anterior elaborado empleando una computadora hogareña, analizaremos algunas características que, de acuerdo a las conjeturas que compartimos en este trabajo, hayan sido incorporadas para adaptarlo al mundo de las máquinas recreativas. En cada apartado ofreceremos evidencias obtenidas mediante el análisis del objeto desde diversas perspectivas, favoreciendo aquellas obtenidas mediante el trabajo de arqueología computacional. La presentación de los hallazgos estará acompañada eventualmente, de nuevas conjeturas e interrogantes.

5.1 La interfaz de usuario.

Las máquinas recreatvas incluyen, por lo general, un modo de operación denominado “modo atracción” o “modo desmostración”. Se trata del modo de ejecución en el que se encuentran cuando no estan en uso. Las máquinas recreativas se

emplazan normalmente en locales públicos donde los concurrentes y potenciales usuarios tienen la posibilidad de elegir las de su preferencia. Las máquinas, en lugar de estar simplemente inactivas, operan en "modo atracción", el nombre remite a la idea de atraer la atención de los visitantes. Una estrategia para lograrlo consiste en mostrar de manera dinámica algunas de las características del videojuego. El "modo atracción" está compuesto por una serie de secuencias que incluye, entre otras: El despliegue del título, la presentación de los elementos y personajes principales, la simulación de algunas instancias del juego en funcionamiento, un cuadro de puntuaciones y las iniciales de quienes las alcanzaron y breve tutoriales que permiten aprender las mecánicas principales. Todo esto ocurre, casi invariablemente, con un despliegue gráfico y sonoro de amplificada estridencia. Una muy buena descripción del clima característico de los salones de videojuegos de los años ochenta, producto de máquinas recreativas operando de manera concurrente se encuentra en[3]. *Trucotron* incluye un modo atracción que cumple con las características que hemos resumido. El objetivo que se persigue mediante la inclusión de dicho modo, nos permite suponer que éste no se encontraba presente en la versión originalmente elaborada para una computadora hogareña. En el proceso de conversión que estamos considerando, la adecuación de la interfaz de usuario representa el primer desafío de diseño. La plataforma de la máquina recreativa posee solamente dos dispositivos de entrada: una palanca de mando que reconoce tres estados (libre, dirección arriba y dirección abajo) y un botón que reconoce dos estados (pulsado o libre). La solución implementada en *Trucotron* es tan ingeniosa como interesante. Diremos que la interfaz de usuario propone una forma de narrativa dinámica, implementada mediante una serie de menúes sensibles al contexto. De este modo, en cada una de las instancias del juego se presentan al usuario menúes conformados por el subconjunto de las opciones posibles. Algunos menúes tienen la capacidad de extender sus opciones de manera dinámica, incluyendo las cartas que tiene el jugador en su poder. Se trata de un sistema interactivo basado en diálogos en los que el programa emplea el voseo y algunos términos del lunfardo como formas de retar o desafiar al jugador. El vocabulario del programa constituye una de sus características distintivas. El análisis del código fuente permite confirmar la existencia de 32 menúes con opciones que cubren cada momento en el desarrollo de la partida. El espacio de memoria necesario para almacenar los menúes se mantiene bajo control mediante la técnica de itemización de los textos correspondientes a cada opción de menú. Todos los mensajes de la interfaz de usuario están en letras mayúsculas, algo que dificulta la lectura, aún cuando se trate de textos y mensajes breves. La plataforma no cuenta con circuitería orientada a la generación de caracteres. El análisis del código de programa, permite confirmar que se encuentra implementado un conjunto reducido de carácteres[4], donde las letras están sólo en mayúsculas. En principio, se puede suponer que la decisión obedeció a criterios de optimización en el uso de la memoria. Los sonidos y las melodías se generan sin asistencia de circuitería específica. La generación de sonidos se logra

[4] Una decisión de diseño muy común en plataformas de videojuego de los primeros años ochenta.

mediante el control de un sólo puerto de salida digital conectado a un circuito analógico que regula la exitación de un pequeño parlante. La rutina encargada de generar sonidos, que es la misma para todo el programa, se ocupa de conmutar el estado del bit de control de acuerdo a la frecuencia necesaria para lograr las distintas notas musicales y efectos sonoros. Esta solución implica que durante la generación de sonidos el microprocesador de la máquina se encuentra abocado por completo a esta tarea.

5.2 Tiempo de juego.

La explotación comercial de las máquinas recreativas exige considerar el tiempo de juego por unidad de dinero; regla que ha definido la dinámica de los videojuegos destinados a este tipo de plataforma. Los videojuegos impiden la inmovilidad: si no se toman acciones, se pierde, es decir se agota el tiempo de juego. Es un mecanismo inherente al modelo de monetización de las máquinas recreativas. Los jugadores intentarán extender el tiempo de juego, algo que será posible en la medida que se adquieran destreza y experiencia. La adquisición de experiencia, por lo general, se logrará a partir de aprendizajes logrados durante el tiempo de juego. Así, podemos pensar que los jugadores, al enfrentarse a un nuevo juego, realizan una inversión inicial. Dicha inversión podría capitalizarse en la medida que se alcanza un grado de dominio que permite extender el tiempo de juego en futuras partidas. Es dado suponer que, para quien disponen las máquinas y cobran una unidad de dinero por cada partida, serán convenientes partidas de menor duración en la medida que el tiempo de explotación de la máquina es finito. En el sector de máquinas recreativas, uno de los parámetros que permiten evaluar una máquina particular es el promedio de recaudación por unidad de tiempo. El análisis exige considerar la necesidad de un equilibrio que, sin llegar al extremo de desanimar a los jugadores, dificulte el desarrollo de partidas que monopolicen el tiempo de máquina sin generar rédito económico. Los juegos interactivos resuelven el problema mediante una serie de mecánicas que demandan acciones constantes por parte del jugador. La variación en los niveles de dificultad para realizar dichas acciones permite regular el tiempo de juego. Una dinámica muy diferente presentan los juegos por turnos. Los juegos de naipes no imponen un tiempo límite para realizar las jugadas, de hecho son los mismos participantes quienes arbitran el desarrollo de las partidas. Como puede verse, se trata una característica que exige adecuaciones si se desea mantener bajo control el tiempo de juego en un videojuego cuya temática es un juego de naipes. La solución implementada en *Trucotron* se basa en la inclusión de un temporizador que limita el tiempo de cada turno. Esta modalidad de juego, que no es propia del Truco, está indicada mediante un mensaje al inicio de la partida. La expresión de dicho mensaje guarda relación con el resto de la interfaz de usuario[5]. Durante el desarrollo del juego se presentan mensajes de aviso empleando un ciclo de colores cambiantes acompañados por señales audibles que dan cuenta del

[5] El mensaje que se presenta es: <<OJO NO SEAS MUY LERDO, QUE PERDES LA FICHA, JUGA BIEN Y NO ARRUGUES ! ! !>>

paso del tiempo. Agotado el tiempo, sin importar en que instancia del partido se esté o los puntajes parciales, el juego se da por terminado.

6 ¿Máquina recreativa o juego de apuestas?

Hasta ahora nos hemos referido a *Trucotron* situándolo en la categoría videojuego. De acuerdo al testimonio de un empresario del sector de entretenimientos[11] sabemos que la máquina *Trucotron* prestó servicio en múltiples salones de videojuegos. El juego, por su propia temática, parece estar destinado a un público adulto. No contamos con elementos para definir el perfil de los jugadores. Una posible conjetura, tendiendo en cuenta las franjas etarias de la concurrencia de los salones de videojuegos, sobre todo en los destinos turísticos, consiste en suponer que *Trucotron* ofrecía una alternativa más acorde a los intereses de un público menos familiarizado con los videojuegos de acción.

Un incentivo que se emplea en gran cantidad de máquinas recreativas consiste en otorgar recompensas orientadas a extender el tiempo de juego [6]. En el caso de *Trucotron* la recompensa se plantea en términos de obtener un partido gratis[7]. La relación entre fichas y partidos es evidente en la medida que se requiere poner una ficha para poder iniciar el juego[8]. Ganar el partido es equivalente a ganar una ficha y jugar lento implica perder una ficha. Podemos resumir todo lo anterior diciendo que la propuesta está centra en apostar una ficha. Sabemos que se trata de un argumento discutible porque las fichas no pueden recuperarse ya que la máquina no incorpora un mecanismo de devolución de fichas. Sin embargo, nos parece importante señalarlo teniendo en cuanta la conjetura que hemos realizado sobre el posible público destinatario.

El análisis del código de programa develó la existencia de un modo de juego basado en créditos. Se trata de un modo de operación al que no se puede acceder desde la interfaz de usuario pero que se encuentra disponible en las todas las máquinas. Es decir, se trata de un modo de comportamiento potencial. Activarlo requiere únicamente configurar un valor determinado en una de las posiciones de la memoria estática (la misma donde residen los códigos de protección y otras configuraciones). Vamos a referirnos a este modo de juego como "modo apuestas" para distinguirlo del modo videojuego o modo recreativo. En la interfaz de usuario del modo apuestas no son visibles algunos aspectos propios de la versión recreativa [9]. Cuando la máquina opera en el modo apuestas, el usuario puede agregar tantos créditos como desee. Adicionalmente puede decidir, en cada mano, el nivel de riesgo que asume. El nivel de riesgo se determina

[6] Los mecanismos más comunes son las vidas extra o la concesión de extensiones al límite de tiempo disponible para alcanzar de terminados objetivos.

[7] El mensaje que se muestra al iniciar una partida es: <<SI ME GANAS,TE DOY UN PARTIDO GRATIS>>

[8] El mensaje que se muestra al inicar una partida es: <<1 FICHA=1 PARTIDO A 15 TANTOS>>

[9] No aparecen los mensajes que retan al jugador, no está activo el modo atracción ni el marcador de tantos.

mediante una escala numérica en el intervalo [1-9] y constituye un multiplicador sobre la puntuación de cada mano. Los puntos se computan de acuerdo a las reglas del Truco pero no existe el concepto de partido sino que se juegan manos individuales. Al final de cada mano se tiene la posibilidad de retirar los créditos acumulados. La interfaz de usuario hace referencia a algunos controles que no están disponibles en la máquina recreativa[10], lo cuál nos permite confirmar que el modo de apuestas requiere ajustes o extensiones al mueble o gabinete de la máquina. Este descubrimiento nos permite plantear nuevas hipótesis respecto de las aplicaciones que pudo haber tenido o que pudieron haber sido proyectadas para *Trucotron*.

7 Cartas sueltas y herramientas de depuración.

Otro hallazgo, producto del estudio del código fuente, es la existencia de un modo de operación que permite especificar las cartas en poder de cada jugador al inicio una mano. Durante el desarrollo de una partida en el modo recreativo, el programa selecciona las cartas de manera pseudoaleatoria. Los intentos por modelar un juego de naipes, desde las perspectivas actuales del desarrollo de software, remiten al concepto de mazo. Así, el mazo constituye una entidad con determinadas propiedades. De acuerdo a las reglas del Truco, la unicidad de los naipes es una propiedad necesaria. En *Trucotron* la solución no está basada en la definición de un mazo que contenga o agrupe a la totalidad de las cartas sino que las cartas se generan sólo cuando son necesarias. Se trata de una solución ingeniosa que no requiere almacenar el mazo completo pero presenta, al mismo tiempo, la necesidad de verificar que las cartas generadas no se repitan. En un partido de Truco son requeridas en cada mano sólo seis cartas. Cuando concluye una mano, las cartas empleadas se vuelven a incorporar al mazo. El elemento de azar se obtiene mediante la mezcla del mazo antes de cada mano. El algoritmo implementado en el programa de *Trucotron* emplea un generador de números pseudoaleatorios al que se invoca de manera iterativa en dos ciclos anidados. El primer ciclo está formado mediante una repetición fija que coincide con la cantidad de naipes necesarios. El segundo, en cambio, se implementa mediante una repetición condicional. La condición de este segundo ciclo se ocupa de cotejar que cada nueva carta generada no coincida con ninguna de las cartas generadas anteriormente. Puede verse que la probabilidad de obtener cartas únicas durante la primera iteración del segundo siglo es menor en la medida que se avanza a lo largo del primer ciclo. Durante una de las fases del modo atracción, el programa despliega un conjunto de veinticuatro naipes seleccionados al azar. El análisis de código permite asegurar que los naipes nunca se repetirán. Creemos que se trata de un detalle muy bien cuidado porque favorece en los usuarios la idea de un mazo de naipes completo y sin repeticiones, algo acorde a los modelos explicativos que podemos tener a mano cuando describimos un juego de naipes. No se ha avanzado en el análisis necesario para determinar la probabilidad de aparición de cada uno de los naipe de acuerdo al algoritmo implementado en

[10] La interfaz de usuario hace referencia a la tecla ENTER y al ingreso de una llave.

Trucotron. Sin embargo, se puede adelantar que, dado el sistema de codificación empleado para los naipes[11] no se trata de un escenario de equiprobabilidad y, en consecuencia algunos naipes tendrán mayores probabilidades de aparición. Volviendo al modo de operación especial, se ha probado que el sistema permite seleccionar mediante menúes especiales, las seis cartas intervinientes en una mano. Luego, el juego se desarrolla del modo normal. Cada vez que comienza una mano se vuelve a invocar al modo de reparto manual. Este modo de operación, al igual que el modo que hemos denominado modo apuestas, se activa mediante la configuración de un valor específico en determinada posición de la memoria de trabajo. Podemos suponer que estamos frente a un modo de operación destinado al análisis del comportamiento del programa durante el proceso de desarrollo. La posibilidad de configurar el escenario inicial puede constituir una ventaja cuando se intenta corroborar el funcionamiento de un programa que opera con valores pseudoaleatporios. Se trata de una herramienta potente cuya interfaz de usuario se programó a partir de las mismas rutinas de servicio que se emplean en el resto del juego. Esto nos da algunas pistas sobre las posibles características del sistema de desarrollo con el que se elaboró *Trucotron*. En primer lugar parece quedar claro que se trató de un sistema sin capacidad de ejecutar el programa resultante, cosa que justifica la inclusión del modo de operación que permite repartir las cartas de modo manual. La existencia de este modo de operación cobra un renovado sentido cuando se tiene en cuenta el modo de operación orientado a juego de apuestas descripto anteriormente.

8 Juego justo.

Es esperable que los usuarios, o por lo menos aquellos con miradas críticas respecto de los dispositivos de lógica programable, se formulen algunas preguntas acerca de su funcionamiento. Del posible universo de preguntas nos interesa responder en primer lugar aquellas relacionadas con lo que denominaremos juego justo. Todo intento por responder este tipo de interrogantes exigirá aceptar que el programa tiene conocimiento de los naipes en poder del usuario. Los juegos de naipes, en su amplia mayoría, no son de información completa. En este sentido, las cartas en poder de los demás jugadores constituyen un conjunto de información con la que no se puede contar. Es frecuente, sin embargo, que durante el desarrollo de una partida se obtenga, de modo lícito, información que permita estimar con diversos grados de probabilidad, las chances de que un jugador tenga en su poder algunas cartas concretas. No vamos a entrar en un análisis

[11] Cada naipe se codifica en dos medias palabras: la primera asigna los dos bits de menor peso para el palo (bastos, copas, espadas y oros), la segunda se emplea para el valor del naipe que está comprendido en el intervalo $[0, 9]$. Puede apreciarse que la distribución no es homogénea aún cuando el generador de números pseudoaleatorios operara generando secuencias de densidad homogénea. La solución implementada para generar una carta, consiste en obtener valores en el intervalo $[0, 255]$ hasta que se tenga uno cuya palabra de mayor peso esté en el intervalo $[0, 3]$ y su palabra de menor peso en el intervalo $[0, 9]$.

pormenorizado sobre las posibilidades de emplear estas estrategias en el juego que nos ocupa. Baste decir que en el Truco las posibilidades de estimar las caras de otros participantes son más bien acotadas. En primer lugar, porque los naipes utilizados se vuelven a incorporan al mazo luego de finalizar cada mano; en segundo lugar, porque la información se obtiene a medida que avanza el juego y, en muchos casos, carece de valor una vez obtenida. En todo caso se trata, como hemos dicho, de formas de juego lícitas. Por el contrario, no es lícito obtener, sin importar el medio o método, información sobre las cartas en poder de los demás jugadores. Cuando se quiere conocer de manera profunda o pormenorizada la lógica de funcionamiento implementada en un programa de computación, no hay mejor registro que el propio código de programa. Aún tratándose de un trabajo en curso, se tienen elementos suficientes para asegurar que *Trucotron* emplea como uno de los criterios para determinar sus jugadas, información sobre las cartas en poder del jugador. La lógica que se ha analizado de manera completa corresponde al juego denominado "envido" que constituye una instancia temprana, y opcional, de cada mano en el juego del Truco. Luego de "repartir" las cartas, de manera pseudoaleatoria y sin mediar otras condiciones para la asignación que no sea la ya mencionada que permite garantizar la unicidad de los naipes en juego, el programa realiza una serie de cómputos. Uno de los valores calculados es correspondiente a los "puntos de envido" de cada participante. Se trata de puntos potenciales, es decir aquellos que si se invita y acepta la jugada se otorgarían al ganador. La estrategia de juego comienza determinando, mediante una alternativa condicional, si se utilizará la información sobre las cartas en poder del usuario. No se ha completado el análisis pormenorizado que permita establecer las características del generador de secuencias pseudoaleatorias y, en consecuencia, asumiremos que todos los números del intervalo tienen idéntica probabilidad de aparición. En estas condiciones, podemos afirmar que el 50 % de las veces el programa decide su jugada de manera ilícita, una forma de hacer trampa. El otro 50 % de las veces emplea una serie de criterios basados en información lícita (las cartas propias). La rutina encargada de determinar las jugadas, cuando opera sin consultar los naipes en poder del jugador, aplica condiciones sobre valores obtenidos de manera pseudoaleatoria y, en consecuencia, es menos previsible. Con todo lo anterior se logra camuflar un comportamiento que, de otro modo, produciría resultados sospechosos y no tardaría en generar una ventaja significativa en favor de la máquina. Pensamos que el conocimiento aportado mediante el análisis del código desensamblado, habilita nuevas formas de abordaje para estudiar el objeto preservado. Ahora tenemos la posibilidad de plantear nuevos interrogantes cuyas respuestas exigen atender a cuestiones éticas.

9 Puertas traseras y mensajes secretos.

En el mundo de los videojuegos, el primer mensaje secreto del que se tiene registro es el incorporado por *Warren Robbinett* es su juego *Adventure* para la plataforma *Atari VCS*. Los videojuegos destinados a sistemas de entretenimien-

to hogareño, se convirtieron rápidamente en una nueva forma de producción cultural. A diferencia de otros productos culturales como libros, historietas o películas; en los videojuegos no era usual incluir información sobre la autoria. Los programadores de Atari comenzaron a manifestar su malestar ante esta situación. *Robbinet* decidió tomar cartas en el asunto incluyendo una sala secreta a la que sólo se puede acceder luego de realizar una serie precisa y ordenada de acciones. El primer paso es recoger un objeto particular cuya representación en pantalla es sencillamente un punto de color gris. *Robbinet* llamó a este objeto "el punto"[6]. En la sala secreta se puede apreciar el mensaje <<Created By Warren Robbinet>> que se muestra coloreado de manera dinámica con todos los colores de la paleta disponible. Un recurso mediante el cuál el autor tiene la posibilidad de presentarse al público. El descubrimiento de la sala secreta, lejos de convertirse en un hecho condenado, habilitó una nueva dimensión en los videojuegos. Ahora el mundo del juego se muestra como un lugar con zonas por descubrir, lugares a los que sólo llegan quienes se dedican a explorarlo a fondo. Diremos antes de terminar que las llaves de acceso a las zonas secretas nunca son sencillas de obtener. El trabajo de arqueología computacional aplicado a videojuegos nos permite explorar el código de programa en búsqueda de las vías de acceso a dichas zonas secretas. Durante el análisis de *Trucotron* se encontró no sólo un mensaje secreto sino la existencia de dos puertas traseras[12]. En *Trucotron*, cada una de las "puertas secretas" conduce a la ejecución de rutinas especiales. Ambas se activan mediante la introducción de breves secuencia donde se combinan los controles disponibles en la interfaz de usuario (palanca y botón). Una de las secuencias tiene dos efectos muy distintos de acuerdo a una condición adicional. Si al momento de introducir la secuencia se cuenta con ocho créditos[13] el sistema mostrará un mensaje secreto con información de autoría. La cadena de caracteres que contiene este mensaje está cifrada[14]. El mensaje, al mismo tiempo, requiere ser interpretado[15]. El caso es que la misma secuencia de comandos, cuando no se cumpla la condición sobre la cantidad de créditos invocará a una función que elimina los códigos de seguridad contenidos en la memoria RAM estática. El efecto de esta operación es dejar la máquina en estado no operativo. La segunda secuencia tiene un efecto similar ya que elimina otro conjunto de códigos de seguridad. Cuando se activa esta secuencia la máquina ya no responde al ingreso de fichas y, en consecuencia, seguirá operando en modo atracción. El ingreso correcto de ambas secuencias destructivas, cuyos efectos no serán apreciables hasta que termine el partido en curso, se confirman

[12] Con puerta trasera nos referimos a una forma de acceso que permite acceder a sistemas de computación cuando se encuentran en entornos productivos, por lo general sorteando los subsistemas de seguridad y activadas mediante secuencias especiales de comandos.

[13] Como la secuencia se ingresa mientras está en curso una partida, para alcanzar la condición se requieren nueve fichas

[14] Si bien se trata de un sistema de cifrado muy elemental, es suficiente para que el mensaje no sea descubierto mediante el volcado de la memoria en formato ASCII.

[15] El mensaje que se muestra es: <<ESTE PROGRAMA DE JUEGO DE TRUCO PERTENECE A LOS PROPIETARIOS DE LA MARCA LANZA PERFUME>>

mostrando un "punto rojo". El descubrimiento de las secuencias destructivas nos plantea una serie de interrogantes que ayudarían a comprender el contexto en el que se desarrolló este producto y las motivaciones y preocupaciones de quienes prepararon estos modos de operación: ¿quienes conocían estas secuencias?, ¿con qué finalidad las prepararon?, ¿qué tensiones en el modelo de producción y comercialización llevarían a estas personas a preparar secuencias destructoras capaces de dejar fuera de servicio las máquinas recreativas?.

10 Próximos pasos.

Hemos señalado que cuando la máquina *Trucotron* pierde el contenido de sus memoria RAM estática (que puede suceder por manipularla de manera incorrecta) por agotamiento de la batería o por el ingreso de la secuencia de comandos destructiva quedará en un estado no operativo. Parte del esfuerzo de este trabajo de investigación se orientó a buscar una forma de recuperar las máquinas bloqueadas. Uno de los estados de bloqueo presenta en pantalla un código numérico y queda a la espera del ingreso de un código complementario o correspondiente. Analizando el código del programa se logró determinar el algoritmo generador del código complementario. Cuando se realiza la operación de desbloqueo, el programa escribe los valores necesarios en la memoria RAM estática y, en consecuencia, el equipo queda nuevamente en condiciones normales de operación. Algo que nos permite suponer un escenario de prestación de servicios mediante el cual la recuperación de las máquinas, desde el punto de vista técnico, consiste simplemente en ingresar el código de validación. Al momento de escribir este trabajo no se ha concluido el análisis que permita confirmar la posibilidad de desbloquear los sistemas en todos los modos de bloqueo detectados. Gran parte del código desensamblado está a la espera de las tareas de análisis, interpretación y eventual reescritura. Aquí se han compartido los primeros resultados de un trabajo que se encuentra en curso.

Referencias

1. Aycock, J.: Retrogame Archeology. Exploring Old Computer Games. Springer International Publishing Switzerland (2016)
2. Chesini, E., Del Dago, G., Wolovick, N.: MS101, la maquinita de bazán. Anales del V Simposio de Historia de la Informática en América Latina y el Caribe. Río de Janeiro/Brasil (2018)
3. Collins, K.: Before the crash : early video game history, chap. One-Bit Wonders : Video Game Sound before the Crash. Wayne State University Press (2012)
4. Del Dago, G.: Creación de un ecosistema donde preservar el primer lenguaje y compilador argentino: Un caso de arqueología computacional. Anales del II Simposio de Historia de la Informática en América Latina y el Caribe, en el XXXVIII CLEI. Medellín/Colombia (2012)
5. Del Dago, G.: Innovación tecnológica en la argentina de los años sesenta. estudio del *SIM1401*. Anales del III Simposio de Historia de la Informática en América Latina y el Caribe. Montevideo/Uruguay (2014)

6. Kent, S.L.: The Ultimate History of Video Games: from Pong to Pokémon–the story behind the craze that touched our lives and changed the word. Three Rivers Press (2001)
7. Kocurek, C.A.: Coin-Operated Americans : Rebooting Boyhood at the Video Game Arcade, chap. The Arcade Is Dead, Long live the Arcade nostalgia in an Era of Ubiquitous Computing. University of Minnesota Press (2015)
8. MAME: About mame (2020), https://www.mamedev.org/about.html
9. Motorola, Inc.: MC6809-MC6809E 8-Bit Microprocessor Programming Manual (1981)
10. Rondan, S.: Truco. Revista Replay **2**(7) (octubre 2017)
11. Rondan, S.: Breve Historia de los videojuegos en Argentina. Playland. Revista Replay **2**(9) (febrero 2018)