JUGANDO A LA INTERFAZ

La estética de la interfaz de usuario en los videojuegos

RESUMEN

Los videojuegos como parte de los nuevos medios influencian y son influenciados por la cultura; algunos de ellos utilizan la interfaz de usuario como recurso mecánico y estético, estresando los límites de lo que se entiende como simuladores o juegos serios.

"*La forma en que los ordenadores hacen disponible su uso, y las suposiciones hechas sobre qué posibles interacciones pueden desarrollarse, son ambas fundamentalmente culturales.*" (Fuller, 2003)

Este proyecto establece las bases para un análisis de las características del uso de la interfaz de usuario como recurso en los videojuegos, a partir de identificar usos comunes y alternativos en otros contextos; para describirlos y valorarlos propiamente. La motivación de esta investigación es encontrar cómo estos videojuegos repiten, reviven y expanden la apropiación de la estética de la interfaz y del código fuente.

Palabras clave: VIDEOJUEGO, INTERFACE, GAME STUDIES, SOFTWARE STUDIES, NET ART.

ABSTRACT

Video games as part of new media are influenced and being influenced by culture; some of them uses the user interface as a mechanical and aesthetic resource, stressing the limits of what is known as simulators or serious games.

"*The way the computers makes available such us, and the assumptions made about what possible interactions might develop, are both fundamentally cultural.*" (Fuller, 2003)

This project stablished a base analysis of the user interface features when used as a resource in video games, by identify common and alternative uses in other contexts; to describe and appraise them better. Its part of this research to find out how this video games repeat, revive and extends the aesthetic appropiation of the user interface and sourcecode.

Keywords: VIDEO GAMES, INTERFACE, GAME STUDIES, SOFTWARE STUDIES, NET ART.

1. INTRODUCCIÓN

En esta sociedad con fácil acceso a la tecnología, estamos acostumbrados a utilizar los dispositivos que nos rodean a través de convenciones de la interfaz de usuario: tocar botones, cerrar ventanas, etc. Esta serie de estándares y regularizaciones se han establecido de forma progresiva. Siendo los ordenadores capaces de construir muchos tipos de interacción, la pregunta no es cómo o porqué llegamos a los usos actuales, sino lo que significan para nosotros, cómo los entendemos, usamos y apropiamos.

El uso de la interfaz en los videojuegos que nos interesan no se limita a simular interfaces gráficas contemporáneas, también utiliza interfaces de línea de comando y se atreve con interfaces híbridas de fantasía. Gracias a esta apropiación, será de particular interés incluir trabajos en la línea del software art, net art, form art, code poetry y codework.

Esta investigación sobre la interfaz de usuario en los videojuegos no se basará ni repetirá aproximaciones hechas por la disciplina del diseño y game studies. Aunque se reconoce la claridad práctica y capacidad narrativa que les concede Chris Crawford en *The Art of Interactive Design* (2002), y la relevancia de los sentidos que les confiere Steve Swink en *Game Feel* (2008), estas referencias, además de los otros análisis del discurso o enfoques semioticos quedarán de lado, no por desestimarlos, sino por realizar un recorrido distinto que considera el estudio crítico de las interfaces de usuario. Igualmente este análisis no se centrará en la interfaz de tipo HUD, no por ser menos interesantes, sino por que su objetivo suele ser funcional, no por forma ni recurso estético.

Ya que el estudio se realiza sobre videojuegos, y aunque se utilicen referencias de otras áreas, conviene utilizar terminología seleccionada de los gamestudies para acotar los objetos de estudio. En este sentido consideraremos la clasificación de interfaz de Kristine Jorgensen en *Gameworld interfaces* (2013) y el modelo de análisis *Mechanics-Dynamics-Aesthetics* de LeBlanc, Hunicke, Zubek (2004).

2. MARCO CONCEPTUAL

La ruta elegida para establecer un puente entre los videojuegos y las interfaces de usuario pasa por el terreno de la informática, el diseño y el arte, y la cruzaremos sobre el vehículo de los estudios del software. El recorrido busca identificar usos canónicos y alternativos de la interfaz, con una parada sobre la línea de comandos y el código fuente.

2.1 LA INTERFAZ

Comenzando con la definición más simple, una interfaz es aquello que está entre dos cosas y que sirve como mediador; enmascara a uno con intención práctica, para facilitar una interacción entre distintos; traduce la intención de uno en la instrucción al otro, muestra la acción de uno en un lenguaje que entienda otro.

Cuando los grandes ordenadores se volvieron comunes en los centros de investigación de las universidades, con la promesa de resolver las necesidades de viejas industrias (militares, transformación, etc), los investigadores identificaron nuevas oportunidades y problemas de estas nuevas máquinas, pensándolas como artefactos creadores y mediadores de la comunicación, no solo como ordenadores y procesadores de datos. Algunas de las soluciones que diseñaron fueron: el dispositivo *Memex* de Bush (1945) un artefacto para la gestión de información basada en "mecanismos" cerebrales de asociación de memorias; el *Sketchpad* pionero en el diseño por computador que su autor Sutherland (1963) bautizó como "*Un sistema de comunicación gráfica hombre-máquina*"; y la metáfora del dispositivo como un vehículo dinámico (multifuncional) de Engelbart (1968) imaginando

computadores personales que aumentaran el intelecto humano. Estas ideas extendían los usos de la máquina a partir de metáforas creando representaciones numéricas (Manovich, 2001) con la preocupación legítima de construir una interfaz para el usuario.

Los primeros ordenadores modernos heredaron el mecanismo de tarjetas perforadas del telar de Jacquar como interfaz de entrada y de salida. El osciloscopio (con orígenes en el radar) fue una de las primeras adiciones como interfaz de salida para desplegar información, y con ella llegaron sus usos creativos. Se pueden datar las figuras abstractas *Oscillons* de Ben Laposky a 1952, y en 1959 Willy Higginbotham programó el proto-videojuego *Tennis for Two*, que mostraba una perspectiva horizontal del juego y cada jugador 'golpeaba' la pelota virtual moviendo girando una perilla individual y presionando un botón, es relevante mencionar que este juego no mostraba puntaje y los jugadores no eran representados en la pantalla.

Cuando el ordenador personal se convirtió en producto de consumo, la industria estableció un estándar de interfaz: dispositivos de entrada (teclado, discos) y salida (pantalla, impresora), y unidad de procesamiento, de tamaño tal que quedaban dispuestos sobre una mesa. En sus inicios la interfaz del software estaba conquistada por el texto. La interfaz por linea de comando (CLI) era el primer punto de contacto de los usuarios, la computadora no hacía nada hasta que el usuario le indicara qué programa ejecutar, todas las aplicaciones debían iniciarse tecleando su nombre, y las más populares mostraban precarios dibujos para organizar la información. El software de consumo más solicitado era el procesador de texto y las novedosas hojas de cálculo como Visical, un programa informático creado en 1979 que simulaba una cuadrícula de papel en la que cada celda podía llenarse con números, texto o simples fórmulas matemáticas (fig 1).

Fig 1. Visical (1979)

Pintores, ingenieros y diseñadores como Jean-Pierre Hébert, Paul Brown, Manfred Mohr (fig), Frieder Nake y Georg Nees en los 70's hacían dibujo asistido por computador programando algoritmos que traducían valores numéricos fijos o aleatorios para seleccionar secuencias de figuras, tamaños, el denominado *algoritmic art*. Es importante aclarar que la pieza se materializaba sobre papel usando plotters, y solo se exponía la imagen impresa, como si fueran pinturas, el código no se mostraba.

2.2 Una interfaz más gráfica

Extendiendo las nociones de hypertexto y construido sobre el paradigma de objetos con la intención de incrementar la utilidad de los ordenadores para el usuario, la interfaz se hizo gráfica. Para Alan Kay, figura clave en el diseño y crítica de los ordenadores modernos, su frase de 1987 *"Doing with Images makes Symbols"*, se basa en los modos de representación de Bruner para establecer cómo la manipulación (ratón) de imágenes (iconos y ventanas) construye abstracciones complejas (resolución de tareas, o programación) (Kay, 2001). Nacido en Xerox PARC, e implementado por todo tipo de fabricantes de ordenadores y compañías de software, la interfaz gráfica de usuario (GUI) estableció un nuevo estándard, el WIMP que en sus siglas enlista sus elementos comunes sin importar la marca del fabricante: windows, icons, menus, pointers.

Tradicionalmente el estudio de las interfaces es competencia del HCI (human

computer interaction) y estaba adscrita a las ciencias de la computación, cuyo objetivo es (y sigue siendo) el diseño y construcción de interfaces optimizadas para la producción y centradas en que sus usuarios cumplan eficientemente objetivos funcionales.

2.3 NET.ART

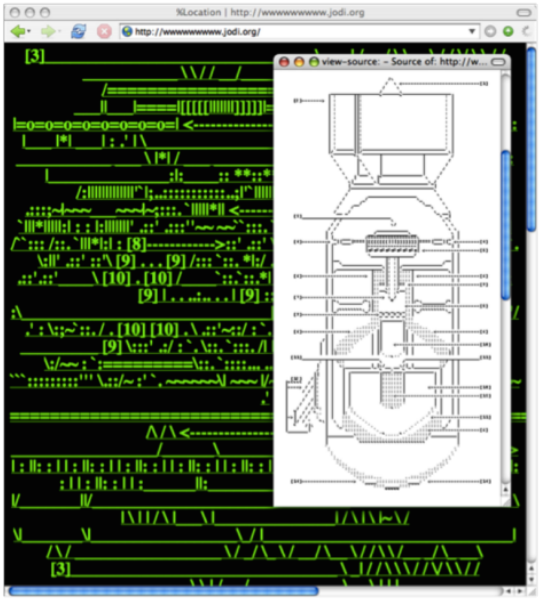

Fig 2. jodi.org (2001)

"*This is the fatal endpoint of the standard mode of HCI. It empowers users by modelling them, and in doing so effects their disappearance, their incorporation into its models*" (Fuller, 2001).
Entendemos net.art como cualquier obra de arte creada para internet y no tenga sentido en ningún otro medio, sobre esto Brea es más propositivo en su definición, y considera solo a "*aquél que invierte el total de su energía en la producción "de" dicho media*" (Brea, 2001).

El duo de artistas conocido como JODI (fig 2) es referencia al hablar de net.art y sus derivados. Su estilo queda de manifiesto en su propio sitio web, que sin indicación para el usuario, es tanto un catálogo como una obra en si, manipulándose a sí misma, teniendo enlaces sin identificar, etc. Obras como *geogoo* donde el mapa de google maps es manipulado para realizar una serie de dibujos usando marcadores sobre el mapa, o como *This folder contains* que entre otras cosas consiste en la proyección de un webcrawler que se manipula a si mismo a partir de los trozos de páginas que encuentra a su paso mostrando el resultado de las páginas manipuladas junto al código que lo hace transformarse.

2.4 REGRESO AL GÉNESIS, LA LÍNEA DE COMANDO

Si bien la sofisticación de la interfaz de usuario avanza en favor de incrementar las capacidades de interacción por lo general aumentando y sofisticando sus características visualmente; en los entornos usados por programadores prevalece un privilegio del texto. Las características más desarrolladas en estos entornos integrados de desarrollo (IDE) simplifican la administración del código, automatizan procesos y permiten encontrar errores; sin embargo siempre son accesorios adicionales del lenguaje de programación.

Es por ello que los lenguajes de programación y su "toolchain" (herramientas para depuración, compilado, ejecución y despliegue) prefieren interoperabilidad de las herramientas de consola.

Como muestra, el servicio de repositorio de código remoto github crea herramientas de linea de comando basadas en convenciones que permiten la automatización por medio de procedimientos igualmente programados y distribuidos en la red. Esto es el perfecto ejemplo de la característica de modularidad de Manovich (2001) que deriva en una estructura fractal de dependencias sobre la que está construida internet.

2.5 SOURCECODE Y EL DESVELAR

Even source code is a kind of interface, an interface into a lower level set of libraries and operation codes. (Galloway, 2007)

El screen-essensialism tiende a estudiar solo aquello que se produce en la pantalla, e ignora la materia que lo compone (Monfort, 2007). Entendiendo esta crítica, en sus estudios sobre la estética del código fuente, Cox y McLean (2013) señalan: *"Como la poesía, el valor estético del código radica en su ejecución, no solo en su forma escrita. Para apreciarlo completamente, necesitamos ver el código para comprender qué es lo que estamos experimentando y construir un entendimiento de las acciones del código".*

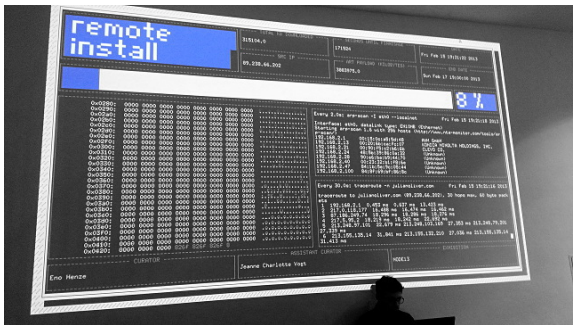

Fig 3. Remote Install (2013) Julian Oliver

La producción artística de Julian Oliver incluye obras que resaltan por su estética de interfaz de linea de comando. En *Men in grey* (2009-2014) una serie de dispositivos evocan herramientas de vigilancia, entre ellos un maletín que supuestamente despliega comunicaciones capturadas furtivamente en las redes cercanas. Por otro lado, *Remote install* (2013) un ordenador que de forma autónoma instala su sistema operativo y descarga archivos de la web del artista, mientras despliega en pantalla un volcado de los datos recibidos y las comunicaciones establecidas (fig 3).

Adentrándonos más en las capacidades específicas del código, no solo en lo que significa su representación, los estudios de estética y generatividad del código de Cox y McLean (2013) se pueden destilar cuatro características:

- **Flexibilidad**. El lenguaje es usado en forma muy controlada y con matices sutiles. "*En términos de forma, cualquier identación y otro patrón visual es una técnica para visualizar la lógica de flujo, aunque el mismo código pueda ser expresado en cualquier forma o distribución y pueda ejecutar la misma salida*" (Ibid)
- **Generatividad**. El hecho de que el código se ejecute varias veces (en ciclos, loops) y que puedan escribirse condicionales para cuando se alcance cierto estado, produce generatividad, en el sentido de que se desenvuelve en tiempo real. "*Toda poesía puede ser vista como generativa pues siempre se encuentra en proceso de ser*". En el ejemplo: `$walk1_beat = ++$walk1_beat % 16;` la interpretación de este código en el contexto de un ciclo o recursión no solo alude al autoincremento de la variable `$walk1_beat` (por el símbolo `++`), la misma línea también indica la producción de un resultado distinto cada 16 repeticiones.
- **Intencionalidad**. Los comandos se pueden expresar de forma distinta. "*(en la poesía) el orden de las palabras ayudan a expresar lo más importante en un enunciado - la condición o la acción*". El código `return if (++$beats < $aTime)` devuelve un valor binario (true o false) según el resultado de la verificación y un incremente, que bien podría escribirse como `++$beats; if ($beats < $aTime){ return true; } else { return false;}`, pero el programador ha preferido dar protagonismo a la respuesta del bloque (return) sobre la condición o el incremento.
- **Intraducibilidad**. Para interpretar el código, no puede separarse la forma del contenido. Aunque el código pueda ser entendido por el ordenador como código fuente, código máquina, raw bytes, para el hombre solo es interpretable en su lenguaje original como fue escrito en su contexto.

En este sentido, Cox invita a que el código y su ejecución deben experimentarse en paralelo. En contraste, Paul Graham en su ensayo sobre la estrecha relación entre la pintura y la programación, rescata una cita de Estructura e Interpretación de Programas Computacionales: "*Los programas deben ser escritos para que la gente los lea, y sólo incidentalmente para que las máquinas los ejecuten*" (Graham 2003).

"Lo correcto siempre se establece en lo que está delante de nosotros (...) Solo allí donde acontece tal desocultar, acontece lo verdadero."(Heidegger 1954)

Al hablar de juegos que resaltan el código, sale a la luz un fenómeno interesante: a pesar de que los ordenadores modernos ya no presentan las mismas limitaciones técnicas que restringían a los desarrolladores de antaño, hoy algunos juegos son populares por utilizar la nostalgia como estética, lo "retro" por elección. Características como el pixel-art, low-poly y el glitch hoy son estilos y técnicas, más que soluciones o consecuencias de limitaciones de memoria, velocidad de cálculo o despliegue gráfico.

En el juego de 1982 *Yars' Revenge* Howard Scott Warshaw necesitaba más espacio de memoria que lo ofrecido por el Atari 2600 para las características de jugabilidad, así que para ahorrar bytes dirigió la memoria de video al stack donde está el código, consiguiendo además un efecto de tipo ruido gráfico que usó para representar una zona de enemigos y un efecto de explosión (Warshaw, 2003). Difícilmente algún jugador podría de darse cuenta de que está viendo el código codificado como pixeles de colores. Aunque Warshaw muestra el código, su intención es la economía de recursos, y el resultado estético nada tiene que ver alguna estrategia de desocultar entendido por Heidegger o Cox.

2.6 La interfaz en los videojuegos

Como marco general para abordar las interfaces en los videojuegos, partiremos del análisis de Kristine Jorgensen en Gameworld interfaces (2013). Para ella, el rol de estas interfaces "*is to facilitate communication between the game system and the player, making interaction easier*" (Jorgensen, 2013), e identifica tres capas en que pueden ser divididas:

"digital game interface can be divided into three layers: (1) the physical hardware interface consisting of input and output devices; (2) the traditional WIMP interface devices that allow users to interact directly with the objects visualized on screen and (3) the gameworld environment itself." (Ibid)

En su libro, la autora desarrolla categorías considerando los dos últimos niveles, abandonando al hardware por estar constreñido a estándares industriales. En su estudio encuentra tres características definidas por opuestos:

- **Integradas** o **superimpuestas**. Refiriéndose a la forma en que se muestran al usuario, si como parte del mundo de juego, o sobrepuestas como una capa externa de HUD (heads up display).
- **Ecológicas** o **empáticas**. Son ecológicas si sus elementos interactúan con otros elementos dentro del sistema de juego, o solo los resaltan sin participar del juego.
- **Ficticias** o **lúdicas**. Cuando la interfaz es coherente con la ficción planteada en el juego, u obedece al sistema de juego.

2.7 Para diseccionar un videojuego

Como medio interactivo, los videojuegos son aquellos que permiten el juego. Un modelo de análisis (particularmente ludólogo) que permite descomponer un videojuego tanto para su análisis, diseño y estudio es el modelo de análisis MDA (LeBlanc, Hunicke, Zubek, 2004) sobre el que estructuraremos nuestras categorías. Este marco de trabajo sugire valorar por separado: las reglas y acciones individuales que permite el juego (mecánicas), la forma en que estas son usadas por el jugador durante el juego (dinámicas), y la experiencia del jugador diseñada por el desarrollador (estética).

Una gran ventaja del modelo MDA a la hora de analizar videojuegos está en la clara delimitación de los elementos a analizar; por el mismo motivo LeBlanc (2000) acotó ocho tipos de diversión, que determinan la aesthetic de su modelo, y que se resumen en: sensación (juego como sensación-placer), fantasía (juego como hacer-vivir), narrativa (juego como desarrollo de historia), reto (juego como ruta de obstáculos), camaradería (juego como entorno social), descubrimiento (juego como territorio inexplorado), expresión (juego como "soap box") y sumisión (juego como pasatiempo sin sentido).

Mecánicamente podemos identificar acciones concretas que realiza el usuario como: introducir datos, presionar botones, manipular recursos (archivos, datos), automatizar procesos (enlazar acciones), personalizar entornos (mover o redimensionar ventanas, configurar preferencias).

En el nivel de dinámica, podemos identificar que las interfaces tienden a plantear situaciones de control en la capacidad manipular recursos de la interfaz a partir de representaciones indirectas, como los mods de JODI o Leandre.

Sobre las experiencias del jugador, podemos incluir en las características estéticas aquellas que evocan situaciones más complejas como la complejidad, la vigilancia y la datificación. Así que cuando nos refiramos a estética, estaremos hablando de la experiencia del jugador lograda a partir de recursos gráficos y sonoros, producida por la emergente interacción de las reglas de juego.

3. ANÁLISIS LÚDICO

Con los elementos rescatados en el marco conceptual: los tipos de interfaz, las varias apropiaciones, la clasificación de tipos de interfaz en el juego, y un modelo de trabajo en juegos, proponemos un método de análisis que permite establecer relaciones para identificar características relevantes.

La metodología para categorización subjetiva propone una matriz de análisis: 1) seleccionar un grupo de videojuegos de distintas épocas y géneros que utilizan la interfaz de forma particular (están fuera del campo de análisis las interfaces tipo HUD (heads up display) que solo muestran o resaltan información por muy integradas que estén al juego). 2) Para cada juego, documentar los siguientes elementos:

- **Tipo de interfaz**. Basados en la clasificación de Jorgensen, identifica la forma en que utiliza la interfaz de usuario dentro del mundo de juego.
- **Nivel de implicación**. Para cada capa del modelo MDA, documenta la forma en que participa cualquier tipo de interfaz.

Y 3) establecer relaciones entre ellos, que revelen obviedades, excepciones y sorpresas.

3.1. JUEGOS SELECCIONADOS

- ***SCRAM***, 1981. Un juego de Crawford para Atari donde tomas el papel de un ingeniero que debe mantener en funcionamiento una planta nuclear. Seleccionado por ser un simulador sin objetivo serio, sino lúdico, de narrativa emergente y crítico.
- ***Real lives***, 2004. Juego de simulación a partir de estadísticas de desarrollo mundial. El jugador es llevado a través de la vida de un avatar simulado al que le ocurren eventos (enfermedades, accidentes, migración, guerras), las intervención del jugador están limitadas a cómo distribuir su ingreso, cómo involucrarse socialmente, etc. además de recibir información estadística y de organizaciones involucradas.
- ***Hacknet***, 2015. Como representante de los llamados *hacker simulators,* este es un juego de aventura donde cumples tareas de hacking rompiendo seguridad, entrando en

ordenadores y manipulando archivos desde la línea de comando; y donde la estética visual de tu interfaz de usuario es exagerada cual serie de televisión.

• ***Her story***, 2015. Es una ficción interactiva donde el jugador analiza los video archivos policiales en un caso de asesinato, interactuando directamente con un ordenador de la policía y su software de búsqueda.

• ***Pony island***, 2016. Este juego que inicia como un inocente juego de ponis, hasta que un virus diabólico se hace del control de todo el ordenador poniendo trampas y acertijos en la interfaz para evitar que el jugador recupere el control.

• ***Glitchspace***, 2016. Se trata de un juego de acción en primera persona en que el jugador tiene control sobre la programación del juego y los niveles, teniendo acceso a una interfaz de programación visual (tipo pure data) que permite manipular objetos y reglas de juego para abrirse paso.

• ***Killing time at lightspeed****,* 2016. Es una aventura a través de la interfaz de una red social. Juegas como el pasajero de un crucero espacial donde cada minuto de viaje corresponde a un año en la tierra, así que tu interacción social es limitada mientras ves pasar la vida de tus conocidos.

4. CONCLUSIONES

En este punto con la metodología descrita, se realizará el análisis de los juegos y se documentarán los hallazgos.

5. BIBLIOGRAFÍA

Brea, José Luis. 2001. La era postmedia: acción comunicativa, prácticas (post)artísiticas y dispositivos neomediales. Salamanca: Consorcio Salamanca 2002 : Centro de Arte de Salamanca.

Cox, Geof, Alex McLean, and Adrian Ward. 2016. “The Aesthetics of Generative Code.” Accessed September 26. http://generative.net/papers/aesthetics/.

Cox, Geoff, and Alex McLean. 2013. Speaking Code: Coding as Aesthetic and Political Expression. Software Studies. Cambridge, Mass: The MIT Press.

Fuller, Matthew. 2003. Behind the Blip: Essays on the Culture of Software. Brooklyn, NY: Autonomedia.

Graham, Paul. 2003. “Hackers Y Pintores.” http://paulgraham.es/ensayos/hackers-y-pintores.html.

Heidegger, Martin, Francisco Soler, and Jorge Acevedos. 1997. Filosofía, ciencia y técnica. 2003rd ed. Santiago de Chile: Editorial Universitaria.

Hunicke, Robin, Marc LeBlanc, and Robert Zubek. 2004. “MDA: A Formal Approach to Game Design and Game Research.” https://www.cs.northwestern.edu/~hunicke/MDA.pdf.

Jorgensen, Kristine. 2013. Gameworld Interfaces. Cambridge, Massachusetts: The MIT Press.

Key, Alan. 2002. “User Interface: A Personal View (1989).” In Multimedia: From Wagner to Virtual Reality, edited by Randall Packer and Ken Jordan, Expanded ed. New York: Norton.

Manovich, Lev. 2005. El lenguaje de los nuevos medios de comunicación: la imagen en la era digital. Paidós comunicación 163. Barcelona: Paidós Ibérica.

Warshaw, Howard Scott. 2003. Once Upon Atari. Documentary.