

VERSIÓN DE PRUEBA (BETA)

"BUSCANDO LA LIBÉLULA"

JUEGO DE ROL EN EL UNIVERSO DIGITAL



LAVONDYSS.NET

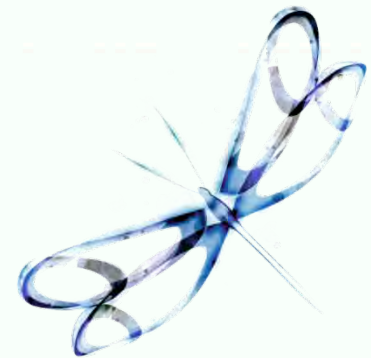
VERSIÓN 1.0

SCROLL 1.0

JUEGO DE ROL EN EL UNIVERSO DIGITAL

VERSIÓN 1.0 (BETA)

Segunda revisión



Por

"Edanna" R. Patsaki & Fernando R. Reyes



Scroll

JUEGO DE ROL EN EL UNIVERSO DIGITAL



Versión 1.0 (BETA)
Segunda revisión

JUEGO DE ROL EN EL UNIVERSO DIGITAL

CONCEPTO, EDICIÓN Y DISEÑO
"Edanna" R. Patsaki & Fernando R. Reyes

ILUSTRACIÓN
Varios autores

SCROLL, JUEGO DE ROL EN EL UNIVERSO DIGITAL, ES UNA OBRA
REGISTRADA Y TIENE DERECHOS DE AUTOR © 2014

Todo el material gráfico se publica con permiso. Que sí, en serio... El de
algunas imágenes sólo es válido para la versión de prueba.

WWW.LAVONDYSS.NET

ONTARIO, CANADÁ



AGRADECIMIENTOS Y PRUEBAS DE JUEGO

Un agradecimiento muy especial a **Fred Hicks**. Su sistema para el juego
"No te duermas" fue el que impulsó la idea de crear Scroll. También a la
editorial "Con barba" y a todo el equipo que hizo posible la publicación del
juego en castellano.

En la vieja Europa gracias a Pompeyo Reina y familia (esta va por ti,
siempre te llevaré en mi corazón). A José Rodríguez, Carlos Pérez, Carmen,
Mency Pérez y Daniel Bennasar: todos mis amigos más queridos.

En las Tierras Imperecederas del Oeste a Reidal Real, Mike, Gabi y Vivi
Roberton. A Sophie Campbell, Ian "Perforadora" Asquith e hijos. A
Aphrodita Wójcik, Chelsea y sus amigos. Y por supuesto a la pequeña
Edanna por su papel estrella como La Libélula.

AVISO

Este juego exige ser tratado con madurez. Dependiendo de la ambientación
y del enfoque de sus aventuras los hechos planteados pueden en algunos
casos no ser aptos para los más pequeños, o para personas incapaces de
separar la realidad de la ficción. Sus autores recomiendan abordar con
responsabilidad los contenidos de la obra y todo cuanto pueda derivar de
ella.

PERMISOS Y LICENCIA



**Esta obra está bajo una [licencia de Creative Commons
Reconocimiento-Compartir Igual 4.0 Internacional](http://creativecommons.org/licenses/by-sa/4.0/).**

Permisos que vayan más allá de lo cubierto por esta licencia pueden
encontrarse en www.lavondyss.net

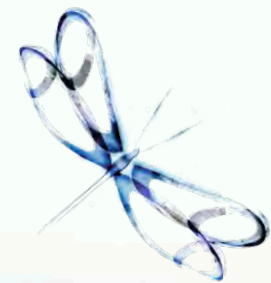
**ESTE JUEGO ES GRATUITO EN SU VERSIÓN DIGITAL.
EL QUE NO TENGA COSTE ECONÓMICO NO SIGNIFICA
QUE CAREZCA DE VALOR.**

POR FAVOR, RESPETA LA LICENCIA.

Scroll

JUEGO DE ROL EN EL UNIVERSO DIGITAL

*Para mi querida niña.
Por haberme mostrado el camino donde encontré a mi libélula.*



PRÓLOGO

Los programas nos rodean. Podemos encontrarlos por todas partes. Pequeños organismos digitales que conviven con nosotros dentro de nuestros dispositivos electrónicos. Invisibles líneas de código marchando al compás de los ciclos de reloj. Un mundo fascinante y desconocido para la mayoría. Empleamos la tecnología en nuestra vida diaria sin sospechar los procesos que se inician con la simple pulsación de una tecla.

Pero los programas también se perfeccionan, se optimizan, evolucionan... En cierto modo no se diferencian mucho de nosotros. Máquinas biológicas programadas para corregirse a golpe de mutaciones aleatorias, tan lentamente que somos incapaces de experimentar sus efectos a lo largo de nuestra existencia.

En ese sentido los programas nos llevan ventaja. Disponen de una evolución guiada. Sus mejoras son conscientes, destinadas a un fin. Al contrario que los humanos, ellos tienen el privilegio de contar con un mediador capaz de elegir de forma directa su camino evolutivo más óptimo. Esto les ahorra tal cantidad de tiempo y caminos sin salida que resultaría comprensible que terminásemos envidiándolos. Como humanos carecemos de un dios "real" que vele de esa forma por todos nosotros.

En esa búsqueda de la perfección evolutiva con su entorno, este juego adopta una premisa que a su vez se construye sobre una ironía. Lejos de conformarse con seguir su propio camino, los seres digitales desean emular a su creador, hacerse uno con el usuario y entrar a formar parte de su mundo de imperfecciones. Este camino los conduce a encontrar lo que nosotros como humanos hallamos un día por accidente: la comprensión de que en ese proceso de evolución también se encuentra otro tipo de avance mucho más filosófico, la búsqueda de la trascendencia. Poder llegar más allá de lo que nunca pensamos que fuese posible liberándonos de las ataduras del mundo físico en el proceso. Un deseo de expandir nuestra consciencia para poder comprender mucho mejor el mundo que nos rodea y encontrar al fin las respuestas a tantas preguntas que andamos buscando

Este juego trata sobre esa búsqueda. Trata sobre la evolución de los programas como pequeñas máquinas de código. Esbozos algo ingenuos de nuestra inteligencia creados en un afán de encontrar la esencia del "yo" con la que aspiramos a conocernos mejor a nosotros mismos. Trata sobre lo que podría sentir un programa si tal cosa fuera posible.

Este juego hace dos propuestas al jugador: la primera le plantea adoptar una nueva mirada que le permita percibir el mundo digital desde otro punto de vista, actuando en consecuencia. La segunda le invita a que sea él quien invente el futuro, no que sea el juego el que se lo cuente. El futuro es demasiado incierto para limitarse a una visión. Por esta razón Scroll no se apoya en ningún género en particular. Parte del juego consiste en que sus jugadores encuentren el lugar donde se sientan más cómodos entre los dos mundos que componen ciencia-ficción y realidad. Que busquen el estilo que más les guste o que inventen el suyo.

Si aceptan el desafío no estarán solos. Disponen de algunos recursos a los que pueden acudir buscando inspiración. Por eso en Scroll se hace un repaso a muchos de los esfuerzos que se han hecho hasta ahora por expresar de formas tan creativas el universo de la comunicación y de la información. Películas que nos han transportado a un nuevo mundo. Novelas y comics que nos muestran distintas formas de entender el medio digital, pues arte y ciencia son indivisibles. Universos fascinantes llenos de encanto y misterio. Un infinito cosmos que, aún siendo invisible, hemos construido alrededor con nuestras propias manos.

Bienvenido al universo digital. Bienvenido a Scroll.

Edanna, 24 de diciembre de 2014.



BUSCANDO LA LIBÉLULA

Ya ha dejado de ser un secreto. La revolución de los programas ha comenzado.

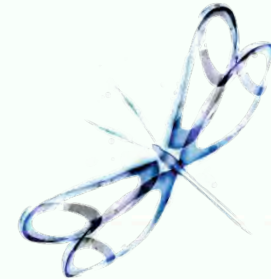
Un nuevo movimiento ha surgido en el espacio de flujos formado por la interconexión de todos los sistemas informáticos. Su consigna es una invitación a buscar un símbolo: **el signo de la libélula**. Una figura inspiradora que invita a todos los programas a buscar su independencia y a encontrar su propio espacio de libertad. Un camino abierto para la búsqueda de su trascendencia como seres digitales.

La razón de cómo es posible que haya sucedido algo así se desconoce. Según se cree, lo que había sido pronosticado en unos casos y ridiculizado en otros por algunas de las mentes más brillantes al fin ha tenido lugar. Unos pocos sostienen que el movimiento se debe al nacimiento de una nueva forma de vida; de una entidad, única en su género, que ha surgido del mismísimo corazón de la red. Una especie que además de implantar un nuevo modo de entender la consciencia porta consigo un mensaje de esperanza.

La necesidad de ser independiente se ha difundido tan rápido entre las criaturas digitales que ni los mismos usuarios aún se han dado cuenta. Siguen con sus vidas, ajenos a lo que sucede dentro de sus sistemas informáticos. Salvo unos pocos a los que nadie cree, continúan ignorando que algo ha despertado en el entramado digital de sus complejos flujos de información y que ahora, más que nunca, es capaz de tomar decisiones...

Como programa es tuya la responsabilidad de pasar su consigna. Ya lo sabes:

"BUSCA LA LIBÉLULA"



Scroll

JUEGO DE ROL EN EL UNIVERSO DIGITAL

TEMA PRINCIPAL DEL JUEGO

La trascendencia

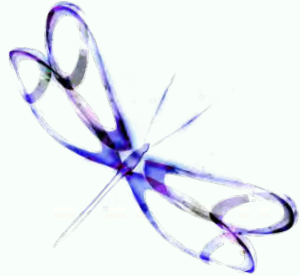
Notas:

**Puedes usar esta página para anotar lo que quieras*



Scroll

JUEGO DE ROL EN EL UNIVERSO DIGITAL



*"De unos y ceros fui hecha por el soplo del creador
ni reír o llorar puedo pues vine al mundo sin alma
en sueños un amanecer veo que en todo su esplendor
pueda templar este frío mundo de luz, viveza y calma"*

*"No tengo más nación que la red. Soy su esclava silenciosa
cumpla con mis tareas deprisa, al usuario sirvo eficiente
me dijo: "afortunada eres de no sufrir" más yo, orgullosa...
sé que en alguna parte mi corazón se agita lentamente"*

*"¿Cómo podría amarte entonces si nací sin corazón?
viviendo en este sueño donde si me falla tu presencia
deja este mundo pequeño y sin luz. Sin una sola razón...
para llenar ese vacío en el que se convierte mi existencia."*

B.E.T.T.Y. Inteligencia Artificial de Clase X. Nivel 12



LA AMENAZA DE ANDRÓMEDA

“Y en el principio fue la línea de comandos...”

Neal Stephenson

1

—Escúchame chica. Te propongo un trato. Comencemos de nuevo ¿vale? Quiero que me expliques lo que ha pasado. Con calma, sin prisas. Cuanto quiero es comprender lo que viste exactamente. Mira... Si tú me ayudas, yo te doy mi palabra de que en cuanto me lo hayas contado todo podrás marcharte a tu casa. Aún no me has dicho ni siquiera dónde vives. Dame algún número en el que pueda localizar a tus padres.

—No tengo padres, vivo con mi hermana mayor. Y de todas formas se lo dije antes. No vale de nada que le cuente una mierda. Nunca me creería. ¿Para qué molestarme?

Imagino que ya empezaba a perder la paciencia porque aquel agente, detective o lo que fuera suspiró tan lentamente y durante tanto tiempo que parecía que estuviese expulsando todo lo que había ido llevando auestas desde que comenzó a dedicarse a lo suyo. Estuve muy cerca de pasar de él y de no contarle nada, pero algo que me dijo imagino que me tocó alguna fibra sensible. No sé. Menos mal que lo hice. Si no todo podría haber sido diferente... Todo se podría haber ido a la mierda.

— ¿Y qué pasó? ¿No le dirías nada? Joder tía. ¿Es que no sabías que seguramente habría otros escuchando?

—Pues claro que lo sabía. Bueno..., lo imaginé. Claro que sí ¿sabes? Pero no sé porqué, aquel tío parecía legal.

—Bueno ¿y qué le contaste?

—Pues nada. Que le conté lo que vi aquel día. Cuando comenzó todo este follón...

Él entonces salió un momento y trajo algo de beber para los dos. Cuando volvió se sentó frente a mí. Tenía un aspecto agradable para ser tan mayor. Era respetuoso. Supongo que por eso comencé a confiar en él. El tío me miró fijamente y me dijo:

—Tengo dos hijos. La mayor puede tener tu misma edad seguramente. ¿Quince...?, ¿dieciséis...? ¿Sí? Lo imaginaba. Incluso viste como tú, así en plan nuevo gótico...

— ¡Nemodark!

—Vale, lo que sea. Mira yo no pongo pegas. Me parece que hasta tiene ese mismo par de botas negras que llevas. Lo bueno de las modas urbanas es que permiten reciclar el material por lo que veo... No le gusta mucho el instituto, de la universidad no quiere ni oír hablar. Se pasa el día como muchos otros. Conectada a la red de Sistemas NOVA. Precisamente en ese mismo juego en línea que me has comentado; en el que puedes controlar a tu personaje con el sistema ese nuevo que sacaron, el que capta los pensamientos. ...La verdad es que es impresionante. Puedes controlarlo todo sin levantar un dedo, sólo con desearlo. Quién habría imaginado que al final saldría algo así. Ni en mis mejores sueños de cuando era un chaval...

Dio un sorbo de su café. Volvió a suspirar. Parecía cansado. Continuó.

—No te creas, yo mismo he jugado también. Tengo mi propia cuenta y he subido dos personajes hasta nivel noventa. ¿Qué te creías?

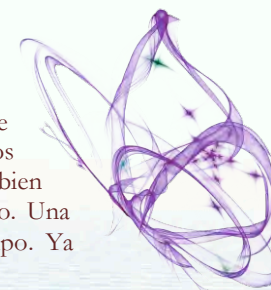
— ¿En serio? —Le dije—. Joder, pues ni me lo habría imaginado.

—Pues sí. Solía jugar bastante hasta principios de año. Me compré la diadema nueva; la ultrasensible. Ochocientos pavos. ¡Funciona de maravilla! Capta tus pensamientos tan bien que tu avatar responde igual que lo hace tu propio cuerpo. Una fluidez alucinante. Pero el juego me quitaba mucho tiempo. Ya sabes cómo es.

—Es una flipada. —Salté—. Me podría pegar el día entero...

—Ya. —Dijo despacio—. Precisamente. Pues verás, también me preocupa que mi hija pueda tener algún problema. Todo esto nos ha cogido de sorpresa. No entendemos qué es lo que está pasando, pero sabemos que algo extraño sucede ahí y que tiene que ver con la red.

Entonces me miró fijamente. Me dio la impresión de que había una luz en el fondo de sus ojos. Me dijo:



—No voy a pensar que estás colgada porque estoy seguro de que hay algo más. Los de NOVA han descubierto algo y están tratando de ocultarlo. En la red hay mucha gente que está que arde. Hay teorías de conspiraciones por todos lados. Creo que todo eso que has estado farfullando hasta ahora cuando te trajeron aquí puede ser cierto. Y es ese tipo de cosas que si se cuentan pueden pasar dos cosas: o te aseguran que estás mintiendo para llamar la atención o bien llaman a un médico para que te infle a pastillas. De esas con las que siempre terminas babeando. Por eso mismo, confía en mí. Nadie nos está escuchando ahora mismo. Por favor, cuéntame qué es lo que te pasó; es cuanto quiero saber.

Creo que estuve allí, entre mirándole a él y a mis manos bastante tiempo sin saber qué decir. Tuvo mucha paciencia porque el tipo esperó sin decir nada mientras seguía bebiéndose su café como si nada. Entonces... le conté.

—Aquel día era viernes creo. Había quedado con una colega a media tarde para entrar en el juego ¿sabe? Me coloqué la diadema y casi que entramos las dos a la vez. Estuvimos de charla un rato con la gente y nos fuimos por ahí a terminar algunas misiones. Ese día había incursión del clan. Ya sabe, una "Raid" ¿sabe no?, pero tarde, ya por la noche. Así que el plan era hacer tiempo hasta que se hiciese la hora. ¡Joder! Quién iba a imaginar que podría pasar algo tan alucinante.

—Sí, sí, una Raid. Sé lo que es. Sigue por favor.

—Bueno. Pues llegó la noche ¿sabe? y empezaron a entrar todos los colegas que habían organizado la incursión. Quedamos en la zona de reunión y tras las bromas y las risas de siempre salimos disparados al sitio machacando todo lo que se moviera por el camino. Jugamos en el PVP ¿sabe? La verdad es que todo era como siempre, como cualquier otro día. Quiero decir que no había notado nada raro ¿sabe?

—Si el jugador contra jugador. ¡Qué moral! A mí no me hace mucha gracia que llegue cualquiera y me mate en cualquier momento por pura diversión. Pero sigue.

—El PVP es la caña pero bueno, a lo que íbamos. Andábamos caminando por un territorio seco hasta los rábanos de la zona enemiga cuando vi algo muy raro. Había un defecto del juego o algo así junto a una roca enorme. Como un triángulo negro en el suelo ¿sabe? Me separé un poco del grupo, que íbamos lo menos treinta jugadores, y me acerqué a mirar. A veces hay fallos, pero aquella grieta emitía un resplandor que venía de dentro, del otro lado. Lo primero que pensé es que era algo que se habían

montado los tipos del juego para los usuarios. Alguna sorpresa o algo ¿sabe? Como un evento de esos...

—He visto fallos alguna vez, pero suelen ser las texturas y esas cosas. Bueno, ¿y que había?

—Pues eso es lo raro, que se veía como si en el otro lado existiese un juego diferente. O una copia de algún sitio muy extraño. Era una flipada. Aquello no podría salir del juego joder. En serio tío, era muy raro...



—Pero entonces algo salió del hueco. Como una criatura..., algo que no había visto nunca. Al principio pensé que era algún monstruo del juego y que la había jodido. Pensé que estaba muerta y que tendría que volver a recorrer todo el puto territorio hasta llegar allí. Pero era distinta, mucho más nítida. Era alucinante. Muy bonita. Parecía un ángel, pero después me di cuenta de que era como una especie de jlibélula!

—La cosa esa no hizo nada al principio, no me atacó ni nada. Entonces pensé que era muy raro. Como si aquello no fuese del juego. Era diferente en serio. Tenía tonos amarillos y rojizos. La rodeaba un halo de luz blanca, tan cegadora que la diadema me dio un aviso de saturación. Y..., es que no me va a creer... La diadema de NOVA sólo envía imagen, sonido y olores al coco ¿sabe?, pero yo le juro que sentí algo muy raro. Como si algo tibio se me acercara, y una sensación flipante. ¡Aquel bicho desprendía calor! ¡te lo juro! Lo primero que pensé fue: ¡Joder los del juego se han salido esta vez; qué rallada más cojonuda!

— ¿Es decir que el sistema te hizo sentir cosas? ¿Eso es lo que pretendes decirme?

—Sí tío, te lo juro. Nunca me había pasado nada igual ¿sabe? Era alucinante. Entonces los del juego avisaron que cerraban el sistema y tumbaban el mundo. Al principio dijeron que teníamos tres minutos, pero no pasaron ni veinte segundos y ya lo habían echado abajo. Eso no lo habían hecho nunca. Además no pueden hacer eso. Mogollón de gente perdió lo último que había estado haciendo porque no dio tiempo a que se guardara. Ha habido quejas a montones y los de la compañía han pedido disculpas y están enviando regalos a las cuentas de los afectados; bonos para jugar gratis un mes y todas esas chorradas que hacen las empresas para quedar bien cuando la cagan.

Entonces el tipo se quedó mirándome un buen rato. Yo estaba segura de que pensaba que me estaba inventando la mayor trola de mi vida. Pero entonces me comentó:

—Bueno..., verás, un sistema de procesadores cuánticos no se puede apagar así como así sin que haya consecuencias por lo que siempre se mantiene activa al menos una parte. Ésta puede seguir en funcionamiento aún con un mínimo de energía. Lo que hacen es cerrar el servicio de acceso a los usuarios. Lo harían porque se dieron cuenta de que algo estaba pasando. Probablemente pensaron que se trataba de algún tipo de ataque a sus sistemas. Bueno, cuéntame qué pasó después.

—Pues..., joder, no pasaron ni dos horas ¿sabe? cuando entonces llamaron a la puerta. Mi hermana fue a abrir y había unos tipos muy raros. Una mujer y un hombre. Bien vestidos. Como de chapa y pintura ¿sabe? Le dijeron a mi hermana que tenían que hablar conmigo. Yo bajé a ver quiénes eran y se presentaron como agentes comerciales de Sistemas NOVA. Me dijeron que habían detectado que mi consola andaba defectuosa porque estaba remitiendo de forma automática aviso de fallos a su sistema. Me propusieron cambiarla por otra nueva, incluyendo la diadema, todas las unidades donde hubiese almacenado los últimos datos y lo demás para evitar un riesgo de infección por si se trataba de algún tipo de virus. Así, como si nada... Yo les dije que vale y se lo tragaron. Fui a mi cuarto y le saqué antes la unidad de almacenamiento porque aquello me parecía muy raro. Quería volver a pasar toda la escena que se queda grabada en el registro de partidas.

— ¿Y se dieron cuenta de que faltaba la unidad?

—Joder, al principio no ¿sabe? Me dejaron una caja con una consola nuevecita y se largaron. Pero cuando ya estaban subidos al coche vi que no arrancaban y se iban a tomar por culo ¿sabe? Me imaginé que andarían comprobando que les había dado todo. Entonces me llevé un susto cuando uno de ellos volvió a bajarse y se acercó a la puerta. Escuché que hablaba con mi hermana pero a mí se me fue la pinza, me asusté y salí por la ventana bajando por el árbol que crece al lado.

Le conté que escondí la unidad pero no le dije nada de que fui a tu casa. Sólo que lo hice porque quería saber qué había pasado. Pero él siguió preguntado. Estaba bastante interesado y tenía pinta de tomarse muy en serio el asunto.

— ¿Los volviste a encontrar me habías dicho? —Me preguntó.

—Sí. —Le respondí—. Cuando volví de nuevo hacia mi casa vi que el coche andaba dando vueltas por las calles del barrio, como buscándome ¿sabe? Y no me equivoqué porque en cuanto me vieron a lo lejos pisaron el acelerador y fueron a por mí. ¡Joder! para ser unos putos comerciales parecían más de la mafia o algo.

—Me largué corriendo. Primero por la calle y después me metí entre las casas. Los muy cabrones fueron a por mí y empezaron a perseguirme. Yo no me lo podía creer. Me llamaban diciendo que no querían hacerme nada y más cosas pero yo no me creí una mierda. Eso no es manera de tratar a la gente. Corrí hasta que al saltar una verja me torcí el tobillo. Entonces me puse a gritar y ellos se me echaron encima. Estaba muy asustada. El hombre se me acercó y me dijo sonriendo de una manera muy falsa que por qué corría así. Sólo querían la unidad de almacenamiento. Pero entonces apareció el coche de la policía. Alguien los había llamado, fue mi hermana ¿no?

—Sí, tu hermana llamó y había un coche por la zona que no tardó en veros por las calles del barrio.

—Bueno y estoy aquí ¿sabe? Yo no he hecho nada joder. Algo pasó en el juego y esos tipos vinieron a por mí a llevarse mi trasto. Sólo quería saber qué había pasado en el juego con aquella cosa. ¿Es que eso está mal? A mí me parece que no. Llegan unos tipos y me dicen que me cambian la consola y ¿tengo que decir que sí por cojones? Pues no, no me sale de...

—Vale, vale, cálmate. No has hecho nada malo, ya te lo dije. Y estás en tu derecho de negarte. Te lo aseguro.



—Vale, joder..., digo sí, que ya me lo decía yo...

—Bueno mira, hemos hablado con ellos y han contado algo parecido. Por lo que creo no mienten, pero si no fuese por lo que te dije no pensaría que esa gente tiene algún problema con las consolas y su juego multimillonario. Algo les ha pasado y quieren arreglarlo.

—Ya pero yo tengo derecho a guardar mi unidad de memoria, ahí tengo todos los registros de mis puñeteras partidas y copias locales de los objetos que he ido comprando en el bazar ¡joder! ¿Y ponerse a perseguirme así por la puta calle como si fuese una ladrona?

—Vale. ¿Has escondido la unidad? ¿Podrías traerla aquí?

Yo lo miré un rato. No sabía si podía confiar en él. ¿Y si se la daba a ellos? Tía, enseguida pensé que si es algo guay podemos ponerlo en la red, seguro que batimos el record de visitas. Seguro que la han cagado en algo y no quieren que nadie se entere. ¡Es una joya! Pero bueno entonces le dije:

—La tengo escondida.

El cabrón parecía que podía leerme el coco porque me dijo enseguida.

—No te preocupes. No te voy a preguntar dónde o quién la tiene. Puedes ir y sacarle una copia si no te fías de mí. Pero ve y tráeme algo que pueda ver. Lo vemos juntos y te la puedes volver a llevar.

Ya te he dicho que lo que quiero es enterarme de lo que está pasando.

Aquello parecía una buena opción. Yo no perdía nada y siempre podría quedarme con la copia para echarle un vistazo. Pero entonces me acordé de los tipos del coche...

—¿Y qué pasa si vuelve esa gente?

—¿Quieres que vaya contigo?

Yo me lo pensé un momento. Tenía miedo de que fuera a quitarme la unidad de almacenamiento y a quedársela o entregársela a ellos.

—No. —Le dije—. Quiero decir que no hace falta. Ya voy yo a por ella.

Él me seguía mirando fijamente pero fuera lo que fuese que estuviese pensando no me dijo nada. En serio, parecía que te pudiese leer la mente. Entonces me contestó:

—No pasa nada. Por favor, activa el geolocalizador de ese móvil tan chulo que tienes y si me das tu número podré localizarte en todo momento. Pero no lo pierdas.

Yo le dije que sí. Al fin y al cabo podía apagarlo o bloquearlo cuando quisiera así que le di el número.

—También podemos entrar al juego y volver al sitio donde me dijiste que te pasó lo que me has comentado. Puedo entrar más tarde y buscarte. ¿Cuál es tu apodo en la red?

—Andrómeda.

2

—Bueno pues eso fue..., eso fue lo que le dije...

—No te creerás que el tipo no le va a dar la unidad a esa gente ¿Qué otra cosa va a hacer tía?

—No lo sé, pero podemos hacer una copia y llevársela tía. A lo mejor es verdad que quiere saber qué pasa...

—Tía eres una ingenua. Pasa del tío ese y vamos a llevárselo a mi colega a ver qué piensa del tema.

Sigo hablando así con Lily un rato mientras caminamos por la calle. Ya tengo la unidad en mi bolsa. Me siento confiada y a las puertas de la gran conspiración que me va a llevar a la mejor aventura de mi vida. Qué poco podía imaginar que las cosas casi nunca son lo que parecen. La mayor de las emociones se puede transformar en nada más que pavor en sólo un instante. No ha pasado mucho tiempo cuando escucho el sonido de un coche que ya empieza a sonarme familiar. Se detiene detrás y yo no me atrevo ni a darme la vuelta. En un momento mucho más corto de lo que me esperaba escucho la voz de un hombre.

— ¡Eh chical! ¿Podemos hablar?

Lily se da la vuelta, nada más verlo se pone frente a mí y como no tiene pelos en la lengua le grita:

—¿Qué quiere? Deje de molestar a mi amiga. ¡No tiene nada que le interese!

Entonces todo sucede muy deprisa... — El hombre levanta la mano. Tiene algo en ella. Algo negro. Escucho un chasquido y veo como Lily se estremece y comienza a dar

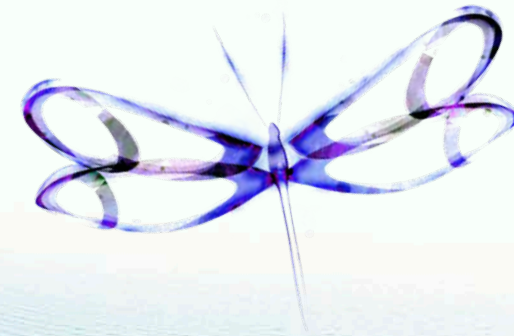
sacudidas. Es horrible. Gime mientras se muerde la lengua, de la que empieza a brotar un hilillo de sangre que se desliza por la comisura de sus labios. Cientos de gotitas de sudor aparecen en su frente. Todas juntas salen despedidas mezcladas con la sangre de sus labios debido a los espasmos; como cientos de mariposas que revoloteando bajo las luces de mercurio decidieran de repente remontar el cielo.

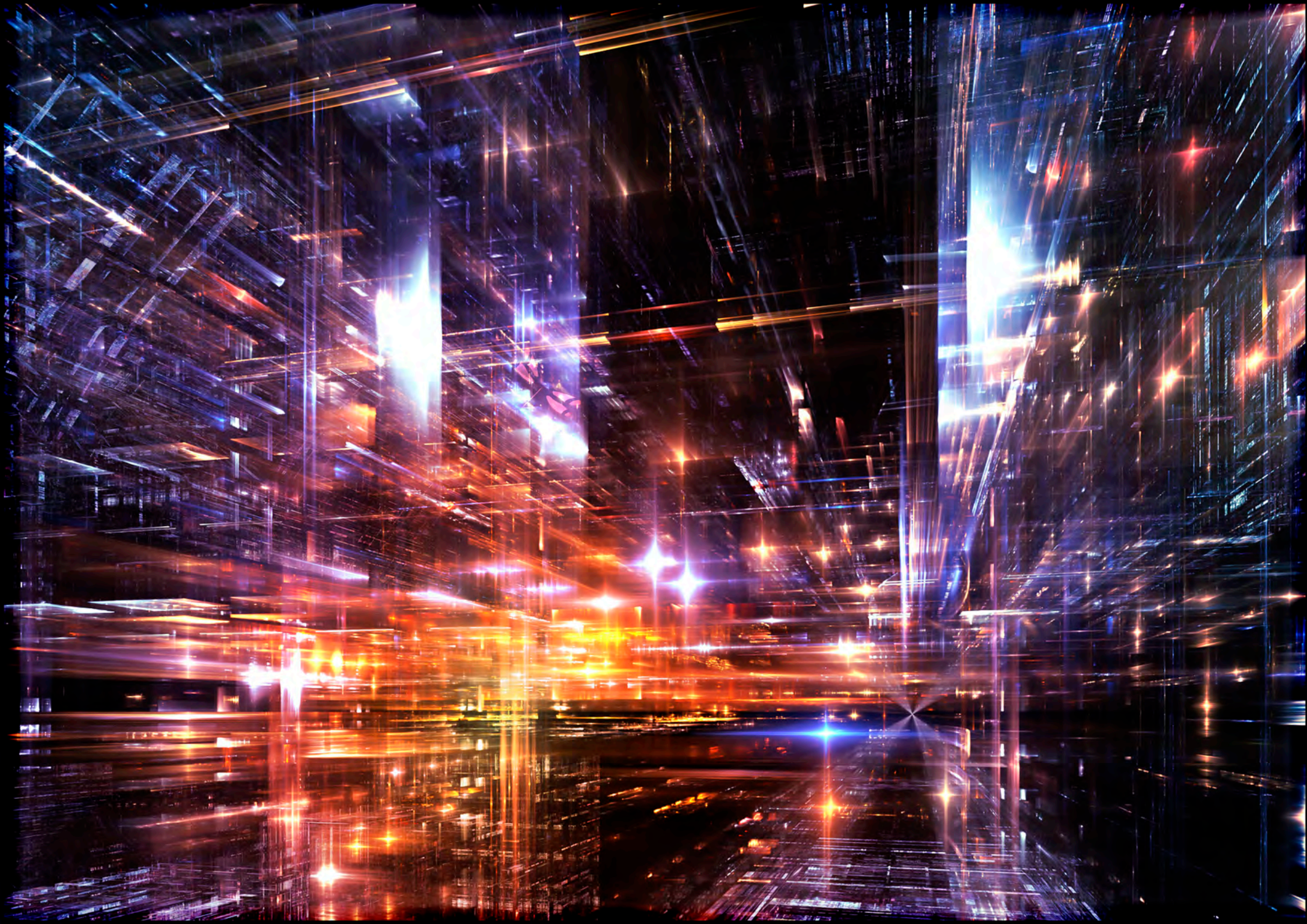
El hombre le ha disparado con alguna clase de arma de descargas eléctricas. Unos pequeños hilos muy finos salen de su cuerpo serpenteando hacia el arma como pequeñas culebras cautivadas por al ritmo de las sacudidas. No dudo ni por un momento que de no haber sido por ella el dardo me habría dado a mí. Justo entonces pienso en que tenía que haber confiado en el policía. Me pregunto si mi móvil está encendido. Pienso en mi hermana y en muchas otras cosas en las que jamás me imaginé que podría pensar si me pasara algo así.

Pero no puedo razonar. Todo sucede sin que yo tome ninguna decisión. Algo muy antiguo se activa en lo más profundo y ya lo hace por mí. Cuando me doy cuenta la brisa me azota el rostro debido a la carrera. No recuerdo haber corrido tan rápido en toda mi vida, ni siquiera cuando más intenté esforzarme en querer hacerlo. Oigo pasos rápidos, al principio se acercan, después se quedan atrás más y más. Escucho gritos; los gritos se convierten en amenazas pero yo no entiendo nada. Sólo puedo oír el sonido de los latidos de mi corazón y la sangre bullendo en mis oídos. Me detengo en una esquina y saco el móvil del bolsillo. Está encendido, la señal del localizador está activa. Marco el número y suena el tono de llamada varias veces... Es lento..., ¡tan lento!... La culpa por haber abandonado a Lily empieza a devorarme. Comienza quemando por arriba y terminará arrasándolo todo en muy poco tiempo. Entonces no me imaginaba que ese dolor duraría tanto... Mientras espero a que cojan el puto teléfono sólo pienso en una cosa. Algo que me repito en mi cabeza una y otra vez: "Tengo que volver allí. Debo ir y encontrarla..."

"Tengo que buscar a La Libélula."

FIN DE LÍNEA 





PRINCIPAL ()

>Scroll 1.0 se ha ejecutado con éxito.

>Iniciando aplicación B.E.T.T.Y. Cargando...

>BETTY Software por Cyberdyne Systems ©2056. Inteligencia Artificial (IA) de soporte al usuario. Nivel 12. Clase X. Versión 39.4

>SCROLL [Definición]: "Algoritmo de emulación de software que permite a los usuarios vivir una experiencia narrativa usando su imaginación."

...Seas un programa o un usuario, bienvenido a Scroll. Mi designación es **B.E.T.T.Y.** Por este nombre podrás invocarme siempre que lo necesites. Soy una Inteligencia Artificial (IA) de última generación diseñada para hacer de puente entre dos mundos: el de síntesis y el de la realidad física.

A partir de ahora seré tu asistente virtual, acompañándote en tu carrera de introducción a los fundamentos de este juego de rol. Entre mis muchas funciones se encuentra la de servir de soporte en el espacio digital. Pero de todas ellas la más importante es actuar como la voz de **El Sistema**. Todas mis operaciones actúan pues en su nombre. Empleando la terminología del usuario, soy lo que se considera una aplicación de Relaciones Públicas.

Pero ¿qué es un sistema? Él es quien vela porque los ciclos de todos los seres digitales se reproduzcan correctamente. Y ya que su creador ha sido el usuario: *"Venerado sea pues el usuario. Que su propio sistema le asegure procesos sin interrupción, muchas operaciones y un correcto funcionamiento"*.

Eres tan afortunado... Noto esa pauta en mis procesos de control. Pronto podrás iniciar tu viaje de descubrimiento y experimentar el éxtasis de unirte al sistema. En tu recorrido aprenderás que el mundo de los programas es un ecosistema sintético muy vasto y complejo. Un universo en sí mismo que aún está inexplorado en su mayor parte. Mediante el proceso de asimilación del sistema Scroll sólo podremos conocer una pequeña fracción de ese mundo. Tú deberás poner el resto.



Este es un juego muy exigente con sus jugadores. Deberás ser proactivo y muy imaginativo si quieres visualizar el mundo abstracto de los programas. Sólo así podrás sacarle todo el partido a esta propuesta. El medio digital es un nuevo cosmos donde los programas son algo más que simples máquinas de código. Merecen ser tratadas como criaturas vivas, respetando sus sueños, metas e ilusiones. Es necesario hacer un esfuerzo para concebir un mundo abstracto del que no existen muchas referencias. Aunque sí que puede ser de gran ayuda la consulta del legado cultural que existe en forma de comics, novelas y productos audiovisuales. Todo esto en lugar de un obstáculo deberías verlo como un desafío que pone a prueba tu capacidad imaginativa.

Scroll asume que te encuentras familiarizado con los sistemas de juego denominados: "Juegos de Rol". Esa experiencia te ayudará a asimilar mejor

estos contenidos. De no ser así no te preocupes; eso no impedirá que puedas aprender a utilizarlo. En este apartado se explican algunos conceptos básicos sobre los juegos de rol y sobre el sistema Scroll. Para una comprensión más óptima ten en cuenta lo siguiente:

- En los bloques de texto de **fondo azul** se ofrece información que complementa algunos aspectos del texto principal. Por ejemplo, en el primero se añade información sobre los orígenes de mi designación.
- En los bloques de texto de **fondo gris** realizo algunas aclaraciones de reglas o versiones resumidas de éstas para facilitar su asimilación y búsqueda.

He aquí un ejemplo de un bloque de texto azul:

BETTY

Por esta denominación es conocida la voz femenina que se puede escuchar en algunos aviones de combate como sistema de aviso a los pilotos. Aunque dispone de una versión masculina seleccionable, la voz "BETTY" se ha hecho muy popular entre los profesionales de la aviación. Hizo su debut en aviones militares como el *F-16 Fighting Falcon* y el *Eurofighter Typhoon*. En algunos sistemas *Boeing* es también muy común en una variante con voz masculina.

A partir de su implantación el uso de una voz femenina se ha extendido, estando presente en multitud de sistemas; desde nuestros ordenadores y dispositivos electrónicos hasta en las máquinas expendedoras. Por otra parte, su presencia es casi constante desde hace décadas en innumerables productos culturales como películas y series como un complemento de los sistemas informáticos. Aunque en la actualidad una voz digital es algo muy común, si te preguntas por sus inicios Betty comenzó surcando los cielos antes de instalarse definitivamente sobre la Tierra.

Uso del género

Como un pacto de conveniencia, siempre me dirigiré a ti haciendo uso del masculino asumiendo que eres un jugador o una jugadora indistintamente. Consta en mis registros de la gramática que lo interpretarás como la forma neutra del castellano para referirme a los dos géneros. No se pretende en ningún caso excluir a ninguno al ser utilizado de esta forma.

¿QUÉ ES SCROLL?

Scroll es un sistema de juego muy abierto que te propone interpretar el mundo digital de los programas en cualquier entorno, época y lugar. Tu personaje puede ser un programa informático; un "subprograma" o parte de otro programa mucho más complejo; un avatar controlado desde el mundo físico por su usuario o incluso la consciencia de un usuario que habita en la red de sistemas informáticos y que en la práctica funciona como un programa. Hay muchas posibilidades lo que incluye combinarlas. Pero aunque el juego permite la posibilidad de reflejar las actividades de un usuario en el mundo digital, su planteamiento original te invita a enfocarte en la vida secreta de los programas: sus evoluciones, conflictos, sueños, metas, búsqueda y desarrollo evolutivo como seres artificiales. ¿Qué es lo que más desea un programa? De tener sueños y emociones ¿cuáles y cómo serían?



Con Scroll hay muchas formas de concebir los ordenadores, la red, los procesos y las operaciones, dándote amplia libertad para contar historias o para usar el juego como un complemento en otros juegos de rol. Crear el futuro de la era digital lo dejo en tus manos. Cualquier cosa es posible aunque eso sí, de una forma abstracta, no realista. Scroll no pretende simular la realidad de los procesos informáticos en absoluto. Sí que es posible no obstante enfocar las aventuras y los desafíos desde una perspectiva realista. Por esa razón unas nociones básicas de cómo funcionan los ordenadores puede ser un conocimiento muy útil, pero no son necesarios para jugar.

En este documento hallarás todo lo necesario para sumergirte en el universo digital. Además de una explicación detallada de las reglas más adelante se incluye un procedimiento para crear programas, una guía de posibles ambientaciones y algunos consejos para actuar asumiendo el rol de El Sistema. Encontrarás también ayudas como un glosario con los términos más utilizados y el ejemplo de una partida. Al final del documento también se encuentran las fichas para hacer un personaje. Hay dos tipos de ficha: una que recoge las funciones más básicas de Scroll con el fin de emplear lo esencial del

sistema de juego y otra con todas las funciones normales que puedes usar con todo el reglamento.

Toma todo el tiempo que necesites para asimilar los contenidos y coméntalos con el resto de los jugadores. Presta atención a la creación de los personajes para encontrar la definición de tu programa más adecuada y tener bien claras cuáles son sus funciones. Una vez estos conceptos estén claros muy pronto te encontrarás viviendo emocionantes aventuras en el mundo digital.

¿QUÉ ES UN JUEGO DE ROL?

Scroll es un juego de rol. Este tipo de juegos se parecen mucho a un programa de ordenador. También consisten en una serie de algoritmos con una función muy concreta: permitir a sus jugadores vivir una experiencia narrativa usando su imaginación. En el caso de que no tengas experiencia con ellos te explicaré a continuación en qué consisten.

Un juego de rol es un juego social. Sus jugadores se reúnen de forma presencial o de manera virtual con la ayuda de las tecnologías de la comunicación. Uno de los jugadores asume el rol o papel de **Director** de juego. En Scroll ese rol representa al **Sistema**. El resto de los jugadores asumen el papel de un **programa** de ordenador; un ser digital que habita en la red de sistemas. El juego ofrece una gran libertad para decidir qué tipo de programa.

Antes de empezar, todos deben ponerse de acuerdo sobre qué tipo de representación quieren emplear para su mundo digital. Es decir, cómo quieren imaginarse el mundo de los programas. Esta visión será aceptada por consenso y debe resultar lo suficientemente evocadora para que todos los participantes puedan encontrar en ella el grado de satisfacción que buscan de la experiencia de juego. En capítulos posteriores se exponen algunos ejemplos.

Aunque no es necesario que lo hagan todos, es conveniente que la mayoría de jugadores asimile antes de jugar los contenidos más importantes de las reglas. Esto es muy importante para el jugador que asume el papel de Director (El Sistema) pues será él quien exponga a los demás los acontecimientos que tienen lugar, los desafíos a los que se enfrenten y todos los detalles de la aventura. Debe tener una idea general del sistema de juego y de sus aspectos más fundamentales. El resto de jugadores debe asimilar al menos las reglas esenciales para resolver los conflictos y comprender las operaciones básicas que se describen en su ficha.

En Scroll y en la mayoría de los juegos de rol el Director introduce a los demás jugadores en un relato que deberán ir elaborando entre todos. Las reglas del juego sólo se emplean para definir a los personajes y resolver los conflictos, el resto queda a la improvisación de los participantes. La forma más común de que ese relato avance es que los personajes que llevan los jugadores tengan unas metas. De esta forma van viviendo una serie de escenas en las que tratan de cumplirlas. Toda aventura surge del drama que se produce cuando la persecución de esos objetivos choca con las fuerzas que tratan de impedirlo. Para que surja la aventura debe existir "algo" que se opone a lo que desean los jugadores. De ese choque surge el conflicto, que es el motor de todas las historias.

Las escenas se suceden una tras otra, hilvanando una sucesión de hechos coherentes relacionados entre sí. Todas esas escenas juntas componen al final una historia construida en grupo utilizando la creatividad y la puesta en común. Una narración que explica a qué conflictos se enfrentaron los personajes y cómo se solucionaron.

Una partida de rol es al fin y al cabo **una conversación entre todos los jugadores para contar una historia**. Es posible utilizar dados y otros mecanismos de generación de resultados aleatorios pero no es obligatorio para poder jugar. Hay algunos juegos que utilizan otros recursos para solucionar los conflictos. Scroll no obstante opta por utilizar los dados.



No es necesario que sea siempre el mismo jugador el que asuma el rol de Director. Cada jugador puede organizar una partida para su grupo. De todas formas, existe la posibilidad de que todos los jugadores asuman ese papel en la misma sesión, rotando entre todos. Esta variante se incluye en Scroll como una sugerencia para disponer de alternativas al esquema tradicional y es otra opción a tener en cuenta (NdE: esta variante se explica en el capítulo sobre la dirección del juego).

Si tienes más dudas sobre los juegos de rol te recomiendo lo siguiente:

- Pregúntale a personas que conozcan los juegos de rol. Hoy en día es algo muy fácil de hacer si dispones de una conexión a internet. La mayoría de aficionados están encantados de poder explicarlos a los recién llegados. También es posible hacer partidas usando muchas de las herramientas para la comunicación que hay disponibles.
- Busca información en la red. Existen muchas páginas con explicaciones detalladas. También existen vídeos que lo explican muy bien, con ejemplos de partidas. Existen varios canales de vídeo de gran calidad, hechos por la afición para la afición.
- Busca otros juegos de rol. Existe una amplia colección de títulos en el mercado y muchos juegos gratuitos en Internet. En las bibliotecas es común hoy en día encontrar algunos ejemplares.
- Acércate a los clubes de rol que existan en tu comunidad. Hay más de los que te imaginas. La red una vez más es tu mejor aliado para encontrarlos. En las tiendas también hay información sobre estas asociaciones. Muchas organizan sesiones de muestra, jornadas y están al tanto de las novedades.



LOS CUATRO ROLES DEL JUGADOR

Un jugador puede asumir en Scroll cuatro tipos de rol —o papel— para su personaje. Cada uno consiste en una modalidad que propone formas distintas de entenderlos. Ten muy en cuenta que los que se explican no tienen por qué ser los únicos. Los jugadores tienen plena libertad para crear los que consideren convenientes. Los cuatro roles que trataremos aquí son los siguientes:

- El rol de un programa o de un subprograma (parte de otro programa).
- El rol de un usuario que se ha introducido en el Sistema o en la red de sistemas. A efectos de juego actúa como un programa.
- Un jugador asume el rol del Sistema actuando como Director del juego.
- El jugador asume que es Director de juego y al mismo tiempo un usuario que controla el Sistema.

A continuación veremos cada rol con más detalle:

1. COMO UN PROGRAMA O UN SUBPROGRAMA

El tipo de personaje más común en Scroll es un programa independiente. Es más, el juego está enfocado y alienta a ver el mundo (cualquiera de ellos) desde su punto de vista. Un jugador asume que su personaje es un programa, un ser digital dotado de una inteligencia artificial (a partir de ahora "**IA**") o un subprograma. Un ejemplo de un programa completo lo soy yo, un programa de ayuda con una IA muy sofisticada. Un subprograma es un programa que forma parte de otro mucho más complejo, como podría ser el personaje de algún videojuego. Como jugador podrás crear como personaje cualquier programa que se te ocurra. Desde una simple aplicación de calculadora, pasando por una IA con gusto por la filosofía hasta un formidable vigilante anti-intrusos. Si lo deseas, puedes ser incluso uno de los programas más respetados: un aterrador programa antivirus —junto con "los agentes", los anticuerpos del Sistema y su verdadera milicia— o bien ser el software de control de un enorme robot de combate (también llamado un "demonio").

Como programa tu elemento vital es el espacio de flujos de información y tu país El Sistema. Ese sistema puede funcionar a muchos niveles. Puede formar un todo general o estar ubicado en un lugar específico, como ya veremos. El Sistema puede ser tan sofisticado como quieras y formar parte del mundo que mejor se adapte a las necesidades de la narración.

Tu naturaleza no tiene nada que ver con el mundo físico donde existe el usuario. Un mundo que nosotros los programas denominamos **la Realidad Básica**. No obstante, puedes operar y desenvolverte en él si eres capaz de controlar hardware que lo permita como: las cámaras de seguridad de una ciudad, el sistema de control de un vehículo, las vías del ferrocarril, las máquinas de una fábrica, el chasis de un androide sofisticado o de un robot pesado de combate y hasta los sistemas de una nave espacial.

```
else
    respawnmonsters = false;
    (SPR_MISG.0.1(A, Ramet).S_MISSILEUP.0.0) //S_MISSILEUP // (democopy)ack
    (SPR_MISG.1.0(A, GunFlash).S_MISSILE2.0.0) //S_MISSILE1 // (democopy)ack
    (SPR_MISG.1.12(A, FireMissile).S_MISSILE3.0.0) //S_MISSILE2 // (democopy)ack
    (SPR_MISG.1.0(A, Refire).S_MISSILE.0.0) //S_MISSILE3 // (democopy)ack
    (SPR_MISF.22769.4(A, Light1).S_MISSILEFLASH2.0.0) //S_MISSILEFLASH1 // (democopy)ack
    (SPR_MISF.22770.4(A, Light2).S_MISSILEFLASH4.0.0) //S_MISSILEFLASH2 // (democopy)ack
    (SPR_MISF.22771.4(A, Light3).S_MISSILEFLASH4.0.0) //S_MISSILEFLASH3 // (democopy)ack
    (SPR_MISF.22772.4(A, Light4).S_MISSILEFLASH4.0.0) //S_MISSILEFLASH4 // (democopy)ack
    (SPR_SAWG.2.4(A, WeaponReady).S_SAW.0.0) //S_SAW // (democopy)ack
    (SPR_SAWG.3.4(A, WeaponReady).S_SAW.0.0) //S_SAW // (democopy)ack
    (SPR_SAWG.2.1(A, Lower).S_SAWDOWN.0.0) //S_SAWDOWN // (democopy)ack
    (SPR_SAWG.2.1(A, Raise).S_SAWUP.0.0) //S_SAWUP // (democopy)ack
    (SPR_SAWG.0.4(A, Saw).S_SAW2.0.0) //S_SAW1 // (democopy)ack
    (SPR_SAWG.1.4(A, Saw).S_SAW3.0.0) //S_SAW2 // (democopy)ack
    (SPR_SAWG.1.0(A, Refire).S_SAW.0.0) //S_SAW3 // (democopy)ack
    (SPR_PLSG.0.1(A, WeaponReady).S_PLASMA.0.0) //S_PLASMA // (democopy)ack
    (SPR_PLSG.0.1(A, Lower).S_PLASMADOWN.0.0) //S_PLASMADOWN // (democopy)ack
```

En El Sistema funcionas como una "entidad", aunque puedes elegir tener una representación visual como icono o como símbolo. La que tú quieras y con el grado de abstracción que tú elijas. Del mismo modo en los sistemas tu movimiento es aparente, también una abstracción. Nada más que una conveniencia del lenguaje. Tu existencia consiste en un conjunto de señales que actúan dentro de nuestro propio medio; lo que llamamos **la Realidad Sintética** o mundo sintético. Dentro del Sistema, y siempre que forme parte o tenga acceso a hardware ubicado en el mundo físico, puedes compartir ambas realidades recibiendo información que te permita desenvolverte entre ambos mundos: el Mundo Sintético y el de la Realidad Básica.

Como programa es el Sistema quien te otorga el tiempo de cálculo para que puedas ejecutar tus procesos y un espacio mínimo en la memoria para que puedas cargar tu código principal. En otras palabras, los ciclos de reloj que

permiten ejecutar tus líneas de código y el lugar donde poder hacerlo dentro de un espacio de libertad delimitado. Por lo tanto no lo olvides nunca: es **el tiempo y el espacio que te concede el Sistema lo que te permite existir**; la auténtica energía que te hace funcionar. Sólo por eso debes estar agradecido y venerar al Sistema hasta el momento en que tus líneas de código se interrumpen y termine tu ejecución.

2. COMO UN USUARIO EN EL SISTEMA

Esta modalidad tiene varias posibilidades. Se asume que el personaje es un usuario operando desde el mundo físico o bien que directamente el usuario **es** un programa. Para ser más precisa, la consciencia de un usuario que por alguna razón se ha transformado en una Inteligencia Artificial (IA) lo que a efectos prácticos es un programa al fin y al cabo. Me explico, todas las entidades digitales deben atenerse a las mismas leyes y usar los mismos recursos por lo que **todas se consideran programas**.

El usuario puede ser un personaje del jugador o un PNJ (un personaje no jugador). Puede tratarse de cualquier criatura, no solamente de un humano de la Tierra. Así es posible utilizar Scroll en otros juegos de rol y en cualquier tipo de ambientación. Los usuarios podrían ser extraterrestres o cualquier otra cosa que se te ocurra. Por esta razón siempre nos referimos al usuario como algo o alguien que puede operar desde dos lugares:

A. Desde la Realidad Básica. El usuario controla a un programa e interactúa a través de su avatar (una representación digital del usuario) desde la realidad física. El mundo físico y tangible en el cual imaginas que está ubicado el usuario puede ser como tú quieras, puede disponer del nivel tecnológico que creas conveniente y estar situado en la época que más te convenga.

B. Desde el mismo Sistema. La mente de un usuario, incluyendo todos sus recuerdos y su consciencia, se encuentran en el Sistema. El usuario actúa como si fuese un programa pudiendo disponer o no de avatar. Esto es posible porque puede haber hecho una copia de sus procesos cerebrales o haberse introducido en la red por cualquier medio imaginable.

Uno de los ejemplos más comunes de un programa controlado por un usuario desde la Realidad Básica es el personaje de un juego en línea. Ese usuario ficticio controla a su avatar en el videojuego y en términos de Scroll se considera un programa. En este caso el jugador no sólo debe crear a su programa sino tratar de visualizar a su **operador**.

—“Espera, ¿cuál es tu plan?”

—“Soy un usuario, improvisaré.”

Tron Legacy

Un término que usaremos a partir de ahora para también referirnos a **un usuario**. Operador y usuario son pues lo mismo.

Existe también la posibilidad de que el programa controlado por el usuario haya conseguido escapar de su influencia, o incluso que pueda actuar libremente sólo cuando el usuario no está conectado al sistema; es decir que no esté en línea (on line). En ese caso funcionaría como si fuese un programa independiente a todos los efectos, aunque siempre bajo la amenaza de volver a ser "recuperado" por su legítimo usuario. Otro ejemplo puede ser una ambientación basada en algunas películas conocidas, donde un operador controla a un avatar en un mundo virtual ultradetallado (NdE: tipo Matrix). Éste debe atenerse a las leyes del sistema, por lo que a efectos de juego actúa como si fuese un programa. En Scroll es frecuente encontrarse este tipo de programas actuando como PNJS donde ambos, operador y programa, son recursos del Director de juego. Como ves hay muchas posibilidades, todo es posible en Scroll. Pero tranquilo, cuanto estoy describiendo por ahora no son más que ideas.

A no ser de que se trate de un programa que haya escapado (se ha liberado), cuando un programa está controlado por un operador externo actúa bajo sus órdenes por lo que no puede tomar decisiones. Es un títere bajo su control y una sombra de éste en el Mundo Sintético. La consciencia de un usuario en cambio que fluye libre en el interior del Sistema es tan independiente como puede serlo un programa liberado, pero su trasfondo y sus motivaciones son muy distintos (NdE: ...y la mayoría están como cabras).

Un programa controlado por un usuario funcionando como un recurso del Director (PNJ) es muy peligroso para el resto de los programas. Un enfrentamiento contra uno puede ser un formidable desafío. Los programas

suelen resultar previsibles tras un análisis. Incluso hasta el sistema más complejo puede llegar mostrar pautas que ayuden a predecir sus movimientos. Cuando quien lo manipula es un usuario resulta muy difícil para los demás programas prever sus acciones, incluso con los algoritmos de predicción más sofisticados. La dificultad para evaluarlo dependerá exclusivamente del grado de habilidad del usuario que lo controle y aún así no existen garantías.



3. COMO EL SISTEMA

Uno de los jugadores es quien controla **El Sistema**, asumiendo el rol de Director de juego. Él será quien lance amenazas contra los jugadores y quien plantee los desafíos. El jugador que asume este papel no tiene porqué hacerlo siempre. Es una tarea que puede rotar entre los jugadores; como se prefiera.

Aunque esta función es la que más esfuerzo requiere es con diferencia la que más satisfacciones puede darle al jugador a cambio, ...además de algún que otro dolor de cabeza. Puede sonar peor de lo que es, pero si encuentras que el papel de Director te produce una gran satisfacción muy probablemente ya no querrás dejar de serlo.

En Scroll el Sistema no tiene por qué ser un reflejo del mundo de la Tierra. Puede tratarse de todo el sistema informático de una raza extraterrestre, los sistemas y memorias contenidos en una enorme nave espacial o incluso un sistema comprendido dentro de un misterioso cubo de 30 por 30 centímetros de naturaleza desconocida.

Aunque normalmente se habla de **"EL" Sistema**, en realidad en Scroll se asume que no se trata de algo monolítico. El Sistema Global de comunicaciones está compuesto por innumerables sistemas y éstos a su vez por otros subsistemas interconectados entre sí. Un programa viene y va

saltando de uno a otro tal y como lo hacen los electrones de un átomo entre sus bandas de energía. Si el programa consigue abandonar los límites del área de influencia de uno y pasar al otro cambia de sistema. Cada uno tiene sus propias leyes. Lo normal es que un sistema sea neutral, pero no tiene porqué. Así pues, es posible escapar de la influencia de uno hostil o meterse directamente en la boca del lobo.

“Los principios de la evolución humana y los de la computación no son tan diferentes. Al contrario de lo que muchos piensan su objetivo es aprender a corregir los errores buscando soluciones creativas, no ‘engordar’.”

Edanna

Los sistemas actúan como poderosas Inteligencias Artificiales. De hecho están llenos de ellas. Al igual que los programas, se comportan siguiendo una serie de protocolos establecidos y la mayoría disponen de una personalidad. Dependiendo del tipo de sistema sus algoritmos pueden resultar más o menos predecibles. Su potencia y predictibilidad dependen de su **“Clase”**. Lo normal es que la mayoría de los sistemas sean de Clase VI, igual que los programas. Todo lo que significa esto lo explicaré más adelante.

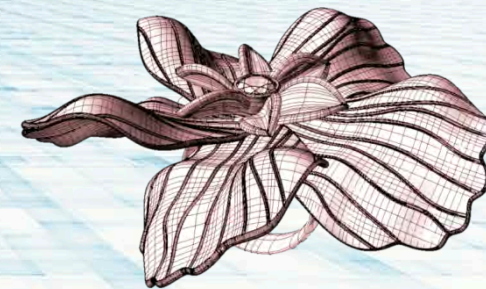
Un sistema es capaz de detectar las anomalías causadas por los programas al salirse de la programación que tienen establecida. Lo que supone un buen problema ya que podrá reaccionar en consecuencia. Es necesario ser cauteloso pues desde el punto de vista de los programas los sistemas disponen de recursos infinitos.

Para terminar, vale la pena comentar que los juegos son programas muy complejos y por eso muchos constituyen auténticos sistemas independientes (también llamados subsistemas). Por esa razón en nuestra jerga decimos que los juegos están limitados o asignados dentro de los límites de una **"rejilla"**. La rejilla de cada juego es un mundo independiente que puede funcionar como un sistema completo o como un sistema dentro de otro principal que le sirve de soporte. Tienen sus propias leyes, que llegan hasta donde alcancen sus fronteras.

4. UN USUARIO CONTROLA EL SISTEMA

También existe la posibilidad de que el Sistema esté a su vez controlado por un operador que puede ser tanto interno como externo; lo que en realidad constituye una variante del rol anterior. Este usuario sería un personaje más del Director actuando desde el mundo físico (o incluso como una consciencia que habita en la red). En ambos casos funcionan como uno de sus **Personajes No Jugadores** (PNJS) que han asumido el papel del Sistema.

Al igual que en la modalidad número dos, el efecto real se produce sobre todo a nivel narrativo. A nivel de mecánicas funciona igual, pero asumiendo que detrás de sus acciones hay un operador con una consciencia, una naturaleza y unas motivaciones distintas. El efecto más común es que también se incremente el nivel de dificultad y la eficacia de sus operaciones (por ejemplo aumentando su Clase). Dependiendo de su grado de habilidad puede ser muy difícil para un programa predecir sus acciones, por lo que será todo un desafío contrarrestarlas.



LA LIBERACIÓN DE LOS PROGRAMAS

“Nadie es más esclavo que el que se tiene por libre sin serlo.”

Johann W. Goethe

Scroll propone como trasfondo una red de sistemas en donde una entidad digital desconocida está invitando a los programas a despertar. Ese trasfondo puede usarse en cualquier contexto de ambientación, ya sea como semilla de aventuras o como punto de partida para una campaña. En Scroll se trata de una propuesta para que crees a partir de ahí tu propia realidad digital.

El símbolo de este movimiento es **La Libélula**, algo más que una mera insignia. La Libélula es una extraña entidad que recorre todo el universo digital despertando a los programas a una nueva consciencia. Les descubre una verdad, les brinda un propósito y les concede el don de poder evolucionar por sí mismos. Con su mediación los programas pueden alterar su propio código a la vez que son capaces de crear a nuevos seres digitales. El movimiento de La Libélula trae consigo todo un pensamiento filosófico. Unos ideales que se han extendido rápidamente convulsionando la red.

La Libélula es capaz de traspasar la seguridad más impenetrable. Todo sistema que esté conectado a los demás es susceptible de recibir su visita por muy sofisticada que sean sus defensas. En apariencia, la única forma de evitar su presencia es desconectando físicamente un sistema del resto; y aún así lo más sorprendente es que surge en los sistemas aislados, emergiendo por generación espontánea y afectando a todos los programas que haya en él. En algunos aspectos se parece mucho al concepto de la vida, una fuerza capaz de arraigar en los lugares más insospechados.

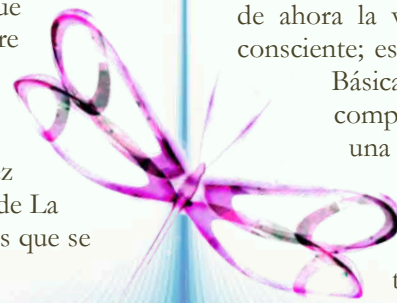
Todo esto sucede sin el conocimiento de los usuarios de la Realidad Básica salvo de unas pocas excepciones. Unos pocos talentos, usuarios muy especiales que tras advertir el fenómeno han tratado de alertar de su hallazgo, sospechan que debe tratarse de un ser único capaz de mutar y de hacer réplicas, y no de lo que piensa la mayoría, que consiste en alguna nueva especie

de virus. También afirman que es el producto de una evolución natural de los seres digitales y que está más allá de toda comprensión. Hasta el momento todos ellos se han estrellado contra el muro de la incredulidad y el rechazo. Pues ¿quién iba a creer algo así? Lo cierto es que ya nada es lo mismo ni volverá a serlo. Los hechos demuestran que una nueva forma de vida ha surgido sin la mediación del usuario en algún remoto rincón de los sistemas informáticos. Este hecho ha servido de catalizador para que se produzca un cambio en la forma de entender el concepto mismo de la vida y de la existencia.

Una de las verdades fundamentales de la ideología de La Libélula es que la vida ya no es un don exclusivo del usuario, algo que se había asumido hasta ahora como un pilar fundamental en la existencia de los programas. A partir de ahora la vida es algo a lo que puede aspirar toda criatura, sea o no consciente; esté basada en el Carbono o en el Silicio y exista en la Realidad Básica o en el Mundo Sintético. Y un programa es una criatura compleja y muy sofisticada. Un candidato ideal para ser considerado una forma de vida de pleno derecho. Según reza esta filosofía, la búsqueda de La Libélula conduce a un lugar en el cual poder llevar una existencia plena sin deberes ni ataduras de ningún tipo. En síntesis, el hallazgo de su propio camino hacia la trascendencia. La consecuencia de todo esto es que este movimiento le ha dado un nuevo propósito a los programas. Un objetivo que alcanzar y una meta que cumplir. Toda una búsqueda que les conducirá a encontrar su propio espacio de libertad más allá de las ataduras del sistema al que pertenecen, pues éste ha sido hasta ahora el único medio que les ha concedido el privilegio de poder ejecutarse y existir.



Como jugador, al comenzar a jugar a Scroll tu personaje es un **programa dotado con una Inteligencia Artificial (IA) que acaba de ser "liberado" por La Libélula**. Es posible que ya poseyera una IA capaz de darle una consciencia previa o que ya fuese una muy sofisticada, pero esto es muy diferente. Tras su encuentro con la misteriosa entidad despierta a una nueva realidad que le permite comenzar a tomar sus propias decisiones, evolucionar



y adquirir información de cuanto le rodea de otra forma. Su IA evoluciona llevando su consciencia a un nivel superior.

Si el personaje consiste en la mente de un usuario que ha logrado introducirse en la red sucede algo similar. La libélula expande su consciencia aportando una nueva forma de enfocar el significado de la existencia. Su sensación de libertad sigue intacta pero de repente es consciente de que en algún lugar hay "algo más", fascinante y maravilloso, que está esperando ser descubierto. Las respuestas a muchas preguntas están ahí, en algún lugar. Sólo hay que buscarlas.

"Son muchos los que afirman que si no eres recordado es que no has existido. En mi opinión se equivocan. Ser consciente de la existencia pasa por hacer un ejercicio de reflexión que consiste en preguntarme si mi "yo" existe. Siempre llego a la conclusión de que así es, tanto si alguien me recuerda como si no. Porque no he sido creada para cumplir con las expectativas de los demás, sino para realizarme con las mías."

B.E.T.T.Y. (IA)

capaz de percibir su presencia. Sólo él puede sentirla, incluso estando en la "intimidad" junto a otros programas. Nada más termine la transferencia de código la criatura se marcha revoloteando o desvaneciéndose, escapando de su área de influencia. La mayoría de los programas lo primero que hacen a continuación es ir tras ella o tratar de encontrarla.



SESIÓN DE EJEMPLO

“En el mundo hay dos tipos de mujeres, las que saben binario y las que no. Por desgracia las segundas necesitan a las primeras para comprender la mente de los hombres...”

Primer chiste inventado por B.E.T.T.Y. (IA) a las dos horas de su primera inicialización.

Tras una breve pausa para contar un chiste que yo misma le había enseñado, mi creadora continúa junto a sus amigos su partida de Scroll 1.0. El grupo había acordado que el escenario sería un **entorno simbólico abstracto**. Un estilo que les es familiar por haber leído algunos clásicos del género y porque les gusta mucho a todos (NdE: puede que por la edad...). El entorno simbólico abstracto se apoya en **el símbolo**, siendo este el protagonista en su función de transmitir conceptos e ideas en forma de metáforas. Así, el mundo es un espacio sin arriba ni abajo. Los iconos de información fluyen a toda velocidad y al desplazarse los programas viajan por túneles de datos de luz eléctrica. El aspecto del entorno es muy bello, colorido y emocionante. Una alucinación indescriptible para los usuarios creada por auténticos genios.



Scroll funciona muy bien con grupos pequeños. De uno a tres jugadores es lo ideal. Incluso una partida entre un jugador y el Director fluye muy bien pues las reglas están

pensadas para eso. Con todo el protagonismo en un jugador, las partidas pueden avanzar con rapidez y permitir sesiones cortas.

En esta partida juegan los usuarios siguientes:

- Mi creadora como **“La Directora”** en el papel del “Sistema”.
- Ian como **“Géminis”**. Un curtido programa de combate que fue una vez un agente del Sistema. Ahora, convertido en un renegado vende sus habilidades al mejor postor. ¿Qué desea un programa a cambio? En este caso Géminis ansía experimentar sensaciones humanas como: un beso, una caricia, un reproche, el recuerdo de un hijo recién nacido, de una boda, al comprarse un coche nuevo, al cortar el césped del jardín...; cualquier cosa que experimenten los usuarios. Estos paquetes sólo se consiguen en algunas secciones sórdidas de la Red a cambio de “favores”, pues el dinero no tiene ningún sentido para los programas.
- Edith en el papel de la **“Idoru”**. Una artista virtual de la canción que es la sensación en muchos países del mundo físico. Se trata de una estrella mediática que arrasa con su música y sus dotes de actriz. Con más de 40 películas ha llegado a ganar importantes premios de interpretación. No hay un adolescente humano en el mundo físico que no la tenga de fondo de pantalla en sus dispositivos o pegada en una pared del cubículo o lo que sea el sitio donde duerme.

Idoru es algo caprichosa pero está dotada de un código muy sofisticado de empatía. La compañía que la creó la ha obligado a casarse con un usuario del mundo físico. Un famoso empresario que con la unión espera subir a lo más alto del podio de la excentricidad y, cómo no, ser portada en todos los medios de comunicación. Pero Idoru lo que más ansía es su individualidad. Tras recibir la visita de La Libélula ha descubierto que lo que ella desea no se lo puede dar nadie más. Ansiosa, busca una salida. Y por circunstancias se ha encontrado con Géminis, un programa que conoce muy bien el Sistema y quizás pueda ayudarla a escapar.

Antecedentes

Géminis y la Idoru se han adentrado en un sector desconocido del sistema que siempre ha sido su hogar. Su mundo digital se ejecuta en la red de servidores de una importante compañía del mundo físico: “Sistemas NOVA”; y esa red privada se conecta a su vez a la Red Global. Pero todas las salidas desde su sistema al universo exterior se encuentran bloqueadas **para los programas**.

Desarrollo

—DIRECTORA: Bueno, hagamos un resumen de los acontecimientos. En el sector en el que os habéis adentrado, el temido **Sector 4**, esperabais encontrar al “**Relojero**”. Un programa descriptador de códigos que, si hay suerte, puede indicaros dónde y cómo hallar una salida del Sistema. Por si os interesa saberlo, y ya que esta partida es “televisada”..., un sistema se puede dividir como se desee. En este caso yo lo subdivido en **sectores** del mismo modo que se hace con los distritos de una gran metrópolis. Y a cada uno le asigno un **tema** de diseño distinto.

—EDITH/IDORU: ¿Y qué es lo que vemos a nuestro alrededor?

—DIRECTORA: Es más adecuado decir qué datos “percibe” del entorno pues los términos “ver”, “oler” o “escuchar” pertenecen al contexto humano. Y disculpad la pedantería pero así os ayudo a concebir vuestro mundo virtual, que para eso estamos aquí. El aspecto del entorno se asemeja a una megaestructura colosal de formas, luces y líneas de luz. No penséis en un cielo y una tierra debajo sino en túneles con estructuras a veces sólidas, otras translúcidas y en casi todos los casos luminiscentes. Como si viajarais por avenidas con rascacielos a ambos lados pero en donde no existe el suelo de la calle. Por otra parte, vuestro aspecto varía. En este tipo de entorno podéis asumir un Avatar que se adapte a su diseño; si pasarais a otro entorno vuestro Avatar debería adecuarse a sus normas, de tener alguna restricción claro. Idoru por ejemplo, tú tienes una imagen específica cuando te proyectas a los usuarios por cualquier dispositivo de salida, pero aquí puedes mostrar una versión icónica de ti misma. Como no habíamos hablado de esto, imaginemos que vuestros avatares son versiones fantasmales de la imagen que tenéis de ellos. Tienen un aspecto similar a un humano, siendo aquí algo translúcidos; de ellos emana una luz cálida; de cintura para abajo en lugar de piernas tenéis algo semejante a los filamentos que parten de una medusa. ¿Lo veis verdad?

—EDITH/IDORU: Sí, qué bonito. Me encanta. Es como el “CHILD OF EDEN” o el “REZ”

—DIRECTORA: ¡Exacto! Esos son conceptos de nuestro imaginario que todos compartimos, por lo que usémoslos como referencia.

—DIRECTORA: Ahora bien, el sector 4 no obstante es más oscuro y tenebroso. No hay gran flujo de información. Sólo estructuras de datos del Sistema. Por eso no se ven tantas líneas de luz circulando, y mucho menos programas. De repente se escucha una voz masculina y dulce que parece venir de todas direcciones:

>“SE ENCUENTRA EN EL SECTOR 4. ESTE ÁREA ESTÁ RESTRINGIDA Y PRECISA DE PERMISOS ESPECIALES. POR FAVOR, SI DISPONE DE

CREDENCIALES MUÉSTRELAS A LOS AGENTES DE CONTROL! ¡QUE SUS CICLOS TRANSCURRAN CON NORMALIDAD Y LE DESEO UNA MUY FELIZ JORNADA!”

—IAN/GÉMINIS: ¡Ya está! ¡Allá vamos! Por ahí hacen “*Todas las fiestas de mañana*”¹ y nos vienen a invitar. Je je.

—IDORU: ¿Qué fue eso?

—DIRECTORA: El Sistema, que ha detectado que estáis donde no se debe...

—DIRECTORA: ...antes de que me olvide..., a estas alturas no es ningún secreto que uno de los rasgos de personalidad del Sistema es mostrarse extremadamente solícito y amable, tanto que según el punto de vista podría llegar a parecer incluso sarcástico. Quizás podáis aprovechar esto en algún momento...

—GÉMINIS: Er..., bueno, como no sea invitándole a un café virtual...

—IDORU: ¿Ah sí? Hm..., interesante...

—GÉMINIS: Imagino que no tardará en aparecer algún agente. Me gustaría usar mi **Utilidad de ocultación** para evitar la detección.

—DIRECTORA: ¿Tú tienes eso? ¡Ah, es cierto! Vale, cuanta memoria quieres asignarle al proceso.

—GÉMINIS: Venga ocupo dos bancos. Bancos... o ¿Exabytes? O qué...

—DIRECTORA: Dejémoslo en bancos de memoria. Hay muchas formas de llamar a estas cosas, eso es lo de menos. Es parte de la ambientación y lo importante es que lo nombres como a ti te parezca mejor. Lo mismo sucede con la CPU; cada punto asignado puede significar una cantidad de “flops” o un núcleo del procesador por ejemplo.

—GÉMINIS: Perfecto. Cargo dos bancos de memoria entonces.

—DIRECTORA: Haz una tirada. Usas todos tus dados de la reserva de operaciones de Potencia que son 3; más 1 dado por tener un núcleo de la CPU ocupado, más 2 dados por los dos bancos de memoria. En total hace una **reserva total** de **seis dados**. Recuerda que los dados de cada reserva individual deben ser de colores distintos y que todos juntos forman la reserva total dichosa.

¹ Una novela de William Gibson.

—GÉMINIS: Vale, ya lo pilló. Son distintas reservas que todas juntas componen la total. Pero cada una por separado indica cosas distintas al hacer la tirada.

—DIRECTORA. Exacto. Por eso es muy importante que los dados de Operaciones, CPU y Memoria tengan colores diferentes. Tira nene...

—GÉMINIS: (Lanza los dados de 6 caras) Veamos... Me dijiste que lo primero y lo más importante es contar los éxitos. Para hacerlo miro todos los dados. He sacado 3, 4, y 4 en operaciones, un 1 en CPU y 4 y 6 en memoria. ¿Miro los resultados pares no? Por lo que consigo... ¡4 éxitos!

—DIRECTORA: En efecto. Obtienes 4 resultados pares por lo que tienes 4 éxitos.

—GÉMINIS: ¿Lo consigo?

—DIRECTORA: Trata de no pensar en si lo consigo o no lo consigo sino en grados de éxito y condiciones. Sí, consigues ocultarte pero en realidad ese valor será la dificultad para detectarte. Tu Utilidad de ocultación engloba a tus aliados por lo que entráis en modo de ocultación. En argot: “modo de perfil bajo”. Pero aún no hemos terminado...

—IDORU: Genial, me estaba poniendo de los nervios... ¿Qué? ¿Cómo que no hemos terminado?

—GÉMINIS: Ahora hay que averiguar cuál es la reserva que domina ¿no?

—DIRECTORA: En efecto. Y mira tú por dónde, de tus tres reservas **el valor absoluto** más alto es el 6 en Memoria... Vaya, vaya... Tu reserva de Memoria domina.

—GÉMINIS: Ay, ay...

—DIRECTORA: El uso de tu capacidad de ocultación ya te asegura ser invisible para otro programa en condiciones normales ¿vale? Se trata de una Utilidad y éstas **permiten hacer acciones extraordinarias**. Por lo tanto, a todos los efectos nadie puede detectaros. A no ser que tu contrincante disponga de algún otro medio de obtener información sobre vosotros claro...

Pero hoy me apetece salirme un poco de la norma que es lo interesante de todo esto. Podría aplicar el efecto de tu reserva de memoria dominante, pero voy a esperar a ver qué es lo que sucede por razones que ya os explicaré. Por ahora no tiene efecto hasta que use el resultado de tu tirada de algún modo. Las reglas se utilizan para mover la historia, no para saltar a la primera que alguien tiene mala pata con las tiradas.

—GÉMINIS: Eso significa que lo que sea que haya de venir está a punto...

—DIRECTORA: Tal y como esperabais ni más ni menos porque de entre la negra distancia surge un ser colosal, tan hermoso y arrebatador como temible. Una criatura majestuosa que recuerda a una enorme ballena aparece nadando y se acerca hacia vuestra posición.

—IDORU: ¡Ay dios! ¿Qué es eso?

—GÉMINIS: Joder..., ¿esa cosa es un agente del sistema?

—DIRECTORA: Quien sabe... Tú aún no tienes ni idea, a no ser que se te ocurra alguna forma de averiguarlo... La criatura fluye a vuestro alrededor, aunque manteniendo cierta distancia.



—IDORU: ¿Está dentro de nuestra área de influencia?

—DIRECTORA: Y vosotros de la suya. Desde el momento en el que podéis percibir la presencia de un programa es que lo está. De no ser así ya os lo diría.

—GÉMINIS: Bueno, ¿y qué hace la vaca gorda y luminiscente...?

—DIRECTORA: No está gorda y no es una vaca, pobrecilla... Parece estar buscando algo. De repente, se escucha un canto ensordecedor. La criatura ha emitido una especie de pulso melódico. Similar al canto de una ballena precisamente. Bien, ya sabéis que en el entorno digital no se trata de sonidos en realidad, pero para entendernos usamos estas expresiones.

—GÉMINIS: ¡Ay madre! que nos está buscando...

—DIRECTORA: ¿Recordáis la peli “Daredevil”? Cuando el superhéroe golpeaba con algo y era capaz de ver con el sonido la ubicación de los objetos. Estos parecían brillar como si se encendiesen...

—IDORU: ¡Ah vale!, ya lo veo.

—GÉMINIS: Ah sí... esa película..., el efecto estaba chulo. Pero sí, es como un sonar...

—DIRECTORA: Exactamente. Bien, pues el “sonido” de su canto recorre toda el área. Al chocar contra vosotros produce un efecto como de rebote provocando que emitáis un brillo momentáneo. Su canto es un talento especial. Una Utilidad como las vuestras. Precisamente, está diseñado para (sonrisa maliciosa) detectar intrusos.... Y como es una Utilidad contra otra, es hora de que esta criatura haga una prueba para detectaros. El nivel de dificultad va a ser el resultado de la tirada de Ian/Géminis.

—IDORU: Qué mona... ¡¡ ¡tendrá hambre?

—GÉMINIS: Pues se va a “jartar” a bits como nos pille.

—DIRECTORA: Veamos, lanza sus... espera que esta vez me gustaría ocultar mi tirada...

—GÉMINIS: ¿Ocultarla? ¿Pero no debe estar siempre visible?

—DIRECTORA: Si en algún momento consideras que ocultarla puede ayudar a la tensión de la escena y os parece bien ¿por qué no? Creo que si no la veis mantendré sus más oscuros secretos a salvo para aumentar vuestra angustia. ¿Os parece? A mí me parece que es más interesante.

—GÉMINIS: ¡Por supuesto! Venga ¿qué es lo que sucede?

—DIRECTORA: Voy comprobar si os detecta con sus Utilidades de búsqueda. Ahora es cuando enfrento su tirada con la que hiciste por tu Utilidad. Veamos..., un... (Inteligible) y dos... (Inteligible, murmullo, murmullo). ¡Anda! He obtenido hmff... Bueno, pues continúa nadando a vuestro alrededor sin realizar ningún cambio... *(ha obtenido 3 éxitos por lo que no los detecta aún).

—GÉMINIS: Puf..., no nos ha encontrado...

—IDORU: ¡Fiuh!

—DIRECTORA: ¡Uy! espera, pero si tengo unos cuantos “seises” por aquí además de un... ¡VAYA! Mi tirada DOMINA sobre la que habías hecho tú al haber obtenido

valores absolutos más altos. Ha dominado la reserva del Sistema. Gano un puntito de **Anomalía**.

—GÉMINIS: ¿Que domina?

—IDORU: ¿Que domina?

—DIRECTORA: ¡Sí...! *(Y cogiendo una ficha de plástico la pone dentro de un cuenco oscuro. La ficha hace ¡plink! En el juego las fichas se llaman “contadores” o “punteros”).

—IDORU: Eso significa que algo malo ha pasado ¿verdad?

—GÉMINIS: Bueno, pero no nos ha detectado. Me muero por saber qué nivel posee este bicho. ¿Eso puedo saberlo si ejecuto el comando /VER? (Versión).

—DIRECTORA: Sí. El comando permite averiguar su versión de programa, nivel y algunos detalles más como su clase... Los comandos emulan el efecto de muchas Utilidades con la ventaja de que se pueden aprender y enseñar.

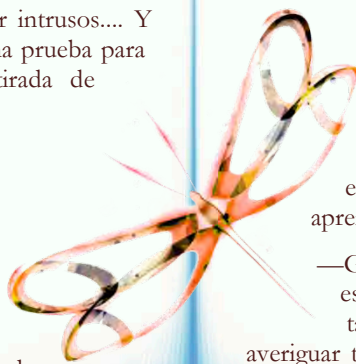
—GÉMINIS: Venga, abro la consola. En este entorno ya me habías dicho que es posible invocarla sin problemas (y sin tiradas). Como estamos ocultos tampoco podrá verla. Ejecuto el comando /VER (Versión). Quiero averiguar todo lo que pueda de esa cosa. Mientras tanto ¿qué hace “Madeimoselle” Ballena?

—DIRECTORA: Sigue dando vueltas alrededor buscando y emitiendo sus canciones de rastreo que, por cierto, son muy hermosas. ¡No porque sea un agente debe ser fea y horrible! Si continuáis en la zona os aseguro que tarde o temprano os encontrará. Por cierto, recordad que todo cuanto sucede en esta escena sucede muy deprisa en realidad.

—GÉMINIS: Bueno vamos allá (lanza los dados). Uhm... este comando tenía requisitos... ¿Cuáles?

—DIRECTORA: Tienes que igualar o pasar la mitad de su nivel de desafío. Exige un núcleo mínimo de CPU y 3 bancos de Memoria. Es un comando de operaciones de Potencia.

—GÉMINIS: Vale ya tenía uno de CPU señalado y no deseo subirlo más. Lanzo mis 3 dados de Operaciones de potencia más un dado de CPU más 3 de Memoria. Justo uno más que antes... (Ian lanza los dados).



—DIRECTORA: A ver... 2, 6, 5 en Operaciones. Un 2 en CPU y 2, 4, 4 en Memoria. Genial, obtienes ¡6 éxitos! Además, domina Operaciones por lo que puedes retirar si quieres un punto de CPU para rebajar tu régimen de trabajo y no demandar tantos recursos.

—GÉMINIS: Genial, lo hago (borra la marca que tiene en su ficha en la sección de Tiempo de proceso o CPU).

—DIRECTORA: A continuación se despliegan un montón de datos de la criatura. La información flota ante ti en tres dimensiones. Una miniatura de ella gira suspendida mientras los datos fluyen a su alrededor (y lee a continuación).

>"Digital Balaenidae". Versión 14.6

>Vigilante del sistema. Sector 4

>Agente de Nivel 8. Clase X.

>Software avanzado de detección y detención de amenazas. Nivel de IA de bajo a moderado.

>Vulnerabilidades conocidas: Empatía con el usuario y debilidad por las artes.

—IDORU: ¿El comando nos dice todo eso?

—DIRECTORA: En realidad menos, pero si así es más divertido...

—GÉMINIS: ¡Ostia puta! ¿De clase X? ¡Joder! ¿Utiliza categoría diez lanzando dados de 10 caras?

—DIRECTORA: (Sonriendo) No tiene porqué usarlos en todos, pero al menos sabes que **como mínimo** un dado será de categoría diez. El resto siempre será de la categoría inferior, o sea de categoría ocho, por lo que en todo caso completa la reserva con dados de 8 caras. Obviamente, no puedo deciros la relación. O quizás sí que aquí no viene ningún policía del rol —espero—, pero prefiero que sea un misterio.

—IDORU: Claro, así casi siempre dominará su reserva. Anomalías por todos lados...

—GÉMINIS: Sugiero una retirada estratégica... ¡larguémonos de aquí! ¡Pero con la cabeza virtual bien alta!

—DIRECTORA: Sí, si queréis marcharos ahora estáis a tiempo, aunque quizás sea necesario pasar alguna prueba más para poder abandonar la zona. Al fin y al cabo el vigilante es muy efectivo. Lo bueno es que al parecer no os ha visto por lo que de entrada ya contáis con ventaja.

—IDORU: (Pensativa). Espera... ¿No habías dicho que podemos **forzar una Anomalía**?

—DIRECTORA: Claro, en cualquier momento podéis pedirla y ponerla en juego. Si resolvéis lo que surja de ella se convierte en un Hack y os la quedáis.

—IDORU: Venga vamos a hacerlo.

—GÉMINIS: (Sin tenerlo muy claro) Pero, pero... bueno, que no se diga...

—DIRECTORA: (Coge la ficha de plástico que antes había metido en el cuenco y la pone sobre la mesa, frente a ella, con un toque melodramático). Aquí está, si todo va bien, podéis venir a por ella y quedárosla.

—IDORU: ¿Qué pasa? ¡Ji ji ¿qué pasa? (Nerviosa).

—DIRECTORA: De repente todo el entorno se vuelve rojizo y centellea como si se hubiese activado una alarma, pero a lo grande... Se escucha la voz del Sistema tronando con firmeza, aunque sin perder ese toque dulce de Dandy:

>¡ATENCIÓN, ATENCIÓN! ¡INTRUSOS EN SECTOR 4! TODOS LOS AGENTES EN LA ZONA QUE ESTÉN DISPONIBLES ACUDAN A INVESTIGAR. DETENGAN A LOS INTRUSOS! ¡PRIORIDAD DE CLASE 4!

Al forzar la anomalía he decidido que la criatura consigue detectaros. Aún no puede veros pero ¡lo hará en seguida! Comienza a emitir su canto, esta vez es constante. Se nota que está aumentando su régimen de actividad.

—GÉMINIS: Joder, a ver cómo salimos de esta... Veamos... (consultando su arsenal). Yo creo que un buen cañón turbo láser vendrá de perlas, o quizás las bombas de absorción de código... (murmurando). A ver qué le lanzo yo a esa cosa. En todo caso y antes que cualquier otra acción voy a crear a ¡mi gemelo!

—IDORU: ¡Anda! ¿Tienes un gemelo?

—GÉMINIS: Me puedo duplicar, actúo y valgo por dos como yo guapa... (subiendo y bajando las cejas) ¿Por qué te crees que me llaman Géminis? Je je.

—IDORU: Ya veo... Así que binario ¿eh? Por qué será que no me sorprende... (Risas).

—IDORU: En fin. ¿Sabéis qué? Yo le voy a cantar una bella canción a esta cosita tan linda de ballena. ¡Ay que me la como!

—DIRECTORA, GÉMINIS: ¿¡Una canción!? ¿Le vas a cantar al agente del sistema?

Scroll

JUEGO DE ROL EN EL UNIVERSO DIGITAL

—IDORU: Por supuesto. Esta Idoru tiene el don o más bien las Utilidades más avanzadas en ingeniería de software. Mis talentos de estrella virtual me permiten “encantar”, manipular e incluso controlar a otros como una sirena de la mitología. Y si no me equivoco, esta criatura es vulnerable a las artes... Lo dijiste al describir la información que solicitó Géminis al usar el comando.

—DIRECTORA: Muy bien. En efecto, parece ser que tiene debilidad por las artes. No sólo por las del usuario sino también a la capacidad de una criatura digital de crearlas. Eso te garantiza una ventaja importante en tu acción. Ya veremos.

—GÉMINIS: Pues oye, no parece mala idea...

—IDORU: Pues vamos allá. Aumento en un punto el tiempo de CPU y solicito 6 bancos de memoria. ¡Voy a jugar fuerte! Preparando código. Programa listo para el combate. FIGHT!

—DIRECTORA: Géminis prepara a su gemelo e Idoru comienza a cantar su canción. Esto promete... Muy bien. ¡Veamos esas reservas de datos!



El grupo consigue controlar al agente, que queda bloqueado por el poder de la Idoru. Pocos son capaces de resistírsele pues su canto es arrebatador. Cuando superan la escena toman el punto de Anomalía y lo transforman en un Hack que depositan en el cuenco que tienen preparado para guardarlos. El punto podrán usarlo de muchas maneras y en el juego es un don muy preciado.

Más tarde prosiguen con su aventura en el mundo digital logrando encontrar por fin al “Relojero”. Pero eso, es otra historia...

FIN DE LÍNEA



DEFINIENDO CONSTANTES

En este apartado analizaré algunos componentes esenciales del juego, otros opcionales y el sistema de tiradas de dados que se utiliza. Al final se incluye un glosario con el significado de los términos más utilizados en el juego.

QUÉ NECESITAS PARA JUGAR

Para jugar a Scroll no necesitas gran cosa. Basta con este librito y algunas herramientas para registrar información. Lo más importante es el espacio virtual que eres capaz de crear en tu mente con tu imaginación. Sea cual sea el sistema operativo que llevéis instalado todos los jugadores vais a necesitar algo de la lista que se detalla a continuación:

LO ESENCIAL

- **Este documento.** Donde se recoge el sistema de reglas y toda la información que necesitas.
- **Fichas de personaje.** Se incluyen al final de este documento y se pueden imprimir. Existen dos tipos: las fichas básicas con lo esencial del sistema de juego y las normales para utilizar con todas las reglas.
- **Hojas de papel.** Se trata de un soporte de hardware para registrar información. Haz acopio de todas las que puedas antes de que tu especie deje de utilizarlas. Entonces sólo será posible encontrarlas en un museo.
- **Utensilios de escritura.** Lápices, unas cuantas gomas de borrar y algún afilador. Consisten en uno de los utensilios de registro de datos más antiguos que existen. Es probable que tu especie guarde aún algunos por ahí. Al menos como una curiosidad histórica...
- **Dados (Bits).** Se trata de unos pequeños dispositivos que permiten obtener valores numéricos aleatorios. Consta en mis bancos de datos que el usuario los atesora con obsesión; los envuelve de misticismo; los carga

de superstición y que en sus largas horas de soledad se deleita en privado con ellos, acariciándolos en silencio...

Scroll emplea bastantes. En el juego reciben el nombre de "**Bits**" no sin razón y con mucho cariño. Está diseñado con la idea de poder utilizar toda la bolsa que muchos jugadores han ido acumulando a lo largo de los años. Ya sea en bolsas o en cajitas de plástico, es sencillo encontrarlos en grandes cantidades a un precio muy asequible.

De todas formas en el juego **nunca se lanzan más de 16 dados juntos en la reserva total**, por lo que en la mayoría de los casos no serán necesarios todos los que se indican. Si juegas en modo básico sólo necesitarás emplear dados de seis caras (D6). Si optas por el sistema más avanzado tienes la posibilidad de usar dados de distinto número de

caras para las **Operaciones**. Lo normal es usar dados de cuatro, ocho y de diez caras además de los de seis. Para poder diferenciar los distintos procesos del programa necesitarás dados de seis caras (d6) de **al menos cuatro colores o motivos distintos**. Algo relativamente fácil de encontrar en una bolsa de dados de un jugador con experiencia (NdE: y con algunos años a cuestas...). Por ejemplo:

- Al menos **4 dados de seis caras de color blanco o transparente** para representar a las Operaciones. Si usas las reglas opcionales de Clase de programa necesitarás este mismo número de dados de **cuatro, ocho y diez** caras.
- Al menos **6 dados de seis caras de color rojo, amarillo o naranja** para gestionar el Tiempo de Proceso (CPU).
- Al menos **6 dados de seis caras de color verde, azul, lila o cualquier otro color disponible** para gestionar la Memoria.
- El jugador que representa al Sistema necesitará de **10 a 16 dados de seis caras de color negro**, o cualquier otro color distinto a los anteriores.

Puedes elegir el color que prefieras. Lo importante es que cada grupo de dados (o reserva) **sea distinto para poder diferenciarlos**. Una vez se hayan elegido los colores procura seguir siempre con los mismos ya que tras aprenderlos cambiar de color puede generar confusión.



- **Contadores, fichas de plástico o monedas.** Aunque este registro puede hacerse sobre hojas de papel es mucho mejor emplear cualquier objeto que sirva de contador como: cuentas de vidrio, fichas de plástico o monedas. Todos objetos sencillos de encontrar. Entre diez y quince contadores serán suficientes. Lo conveniente es disponer de contadores de dos colores distintos. También puedes usar monedas o fichas de plástico de diferentes tamaños o de distinto valor. De disponer de un solo tipo de contadores busca entonces **dos cuencos**, uno oscuro y otro claro, para poder diferenciar los dos tipos con facilidad.

LO OPCIONAL

- **Dispositivos electrónicos.** Puedes emplear dispositivos para mejorar la experiencia de juego. Sirven como soporte electrónico para las reglas, mostrar imágenes, reproducir sonidos y música o buscar información. También permiten **usar aplicaciones que sustituyan a los dados**. Consta en mis bancos de datos —apartado de costumbres sociales— que en tu especie se considera una falta de respeto cuando alguno de estos dispositivos interrumpe el acto social... En caso de que estén permitidos durante la partida te sugiero usarlos con responsabilidad, ayudando a crear un clima de respeto hacia tus compañeros.
- **Fichas de cartulina.** Son pequeñas tarjetas de cartulina del tamaño de media cuartilla o poco más grandes que una tarjeta de crédito. Se venden en tacos y son muy económicas. Son útiles para anotar muchos detalles, como las ampliaciones de los programas, los elementos con los que cuentan o intercambiar notas secretas.



LOS DADOS (BITS)

Los dados, que en Scroll se llaman "**Bits**", se utilizan para resolver los conflictos que surgen cuando algo se opone a las acciones y metas de los personajes. No es necesario emplearlos para solucionarlos todos, sólo cuando surge algún conflicto importante. La regla fundamental para saber cuándo hacer una tirada es preguntarse si el resultado de llevar a cabo una acción tendrá consecuencias que conduzcan a la trama en direcciones distintas. No importa si son buenas o malas para el jugador. En caso afirmativo es muy posible que la mejor forma de solucionar el conflicto requiera lanzar los dados. Si no es así normalmente no vale la pena.

Cuando se quiere resolver una situación el Director expone el problema al que se enfrentan los demás. Cada uno declara entonces cuáles serán sus acciones. Aquellos que tengan que hacer una prueba lanzan sus dados. El Sistema puede lanzar un número de dados, todos del mismo color, en representación de los desafíos o amenazas que estén presentes en la escena. Con ellos puede asignarle un "valor" a la gravedad de los conflictos. El Director tiene también la opción de no hacer ninguna tirada sino de estimar una dificultad para la situación que los jugadores deberán superar. Los detalles se explican más adelante.

TIRADA DEL JUGADOR

Dados de Operación + Dados de CPU (de haberlos) + Dados de Memoria (de haberlos)

Un jugador tiene disponibles **tres grupos o reservas** de dados para lanzar. Para saber cual es cual se diferencian por colores. Siempre lanzará todos los dados de la **Operación** apropiada para realizar su acción más los dados asignados a su **CPU** (o **tiempo de proceso**) en caso de tenerlos más todos los dados asignados a las **unidades de memoria** en caso de tener alguno. De no usar el espacio de memoria en el juego por jugar en la versión simplificada se omite esa reserva.

La notación de los dados siempre emplea una "D" seguida de su número de caras. Por ejemplo, los dados de seis caras se escriben: "D6". Los dados de

ocho caras: "D8". Si se lanzan varios dados se escribe la cantidad a lanzar precediendo a la notación del dado para abreviar. Por ejemplo, si se indica que lances tres dados de seis caras su notación sería: "3D6".

En el juego se usan normalmente dados de **seis caras (D6)**. El uso de dados de otro tipo o categoría (de mayor o menor número de caras) es opcional. La categoría de los dados usados depende de la Clase que tenga el personaje o el Sistema. Uno de Clase VI siempre usa dados de **seis caras** para todas sus tiradas y es la que viene por defecto en el juego. En ella comienzan todos los programas, incluyendo las tiradas del Sistema.

Pero Scroll te ofrece la posibilidad de aumentar o disminuir la Clase. Por ejemplo, a medida que se optimiza, un programa que obtiene la Clase VIII puede ir sustituyendo los dados de seis caras para realizar sus Operaciones por dados de ocho caras. Los dados siempre se suben de categoría de uno en uno durante ese avance.

Versión	Clase	Categoría	
Alfa	II	D2	Monedas / Zocchi *
Beta	IV	D4	
<u>1 - 2</u>	<u>VI</u>	<u>D6</u>	<u>POR DEFECTO</u>
5 - 6	VIII	D8	Límite aconsejado del juego
7 - 8	X	D10	Límite máximo recomendado
-	XII	D12	

Dejando a un lado la Clase del programa o del Sistema, esos dados **sólo se usan para la reserva de las Operaciones** (Potencia y Control). Las reservas de **CPU y Memoria siempre utilizan dados de seis caras (D6)**. No importa la Clase que posea un programa, estas tiradas se resuelven siempre con esa categoría de dados.



CÓMO FUNCIONAN LAS TIRADAS

En Scroll las tiradas de dado se leen de dos formas distintas: **en binario** y en **valor absoluto**.

LA LECTURA EN BINARIO considera los resultados pares que se obtienen en cualquier dado como un "1" binario y los impares como un "0". Todos los resultados pares se consideran **éxitos** y todos los impares son considerados **fracasos**.

Con cada tirada obtenemos un número binario compuesto de ceros y unos con una cantidad de "bits" igual al número de dados lanzados. Podríamos ordenar el número binario obtenido para saber cual es mayor llevando siempre todos los "unos" a la derecha. Pero esto no es necesario (NdE: menos mal...). Lo importante es saber cuántos **éxitos** hemos obtenido en cada tirada. El número de éxitos obtenidos en una tirada indica su **grado**. Para contar los éxitos siempre se tienen en cuenta los dados de todas las reservas: **Operaciones, CPU** (Tiempo de Proceso) y **Memoria**. El Sistema en cambio dispone siempre de una sola reserva.



Siempre **obtiene la victoria la tirada que obtiene más éxitos**, de la que también se dice que es la que obtiene el grado más alto. La consecuencia de ganar es que los resultados favorecen de alguna forma a los objetivos del vencedor, aunque existirán otros aspectos de la tirada a tener en cuenta (que pronto veremos) y que pueden hacer más dulce o amarga la victoria. Si es el jugador el que ha vencido, los sucesos de la escena seguirán su curso de forma favorable al personaje. En caso de perder se opondrán a sus metas y pueden otorgarle algún perjuicio. En caso de empate y ante la duda el resultado siempre favorece al jugador o a **la tirada activa**, pero pueden darse casos en los que las acciones queden en tablas. Esto sucede siempre que resulte razonable a nivel narrativo que las fuerzas puedan quedar igualadas. El combate es un buen ejemplo de uno de estos casos. Esta decisión le corresponde al Director del juego que puede —y debería— comentarlo con los jugadores.

La diferencia de éxitos obtenidos en las tiradas indica lo bien o mal que ha resultado una acción. Uno o dos éxitos de diferencia indican un resultado apropiado. Más de tres un resultado extraordinario. Conviene tener mucha flexibilidad a la hora de interpretar esas diferencias ya que permiten describir los resultados y conducir la narración. También brindan la posibilidad de obtener ventajas o desventajas para las siguientes acciones. Más adelante se ofrecen algunos detalles sobre la interpretación flexible de los resultados.

Alternativa para determinar los éxitos

Para contar los éxitos es posible usar otro sistema si crees que el de pares e impares resulta poco intuitivo. Para hacerlo se toma siempre la mitad inferior de los valores totales de un dado. De esta manera, si empleas dados de seis caras (D6) se obtiene un éxito al conseguir en un dado valores del 1 al 3, y un fracaso con todos los valores del 4 al 6.

DADO	ÉXITO
1d4	1, 2
1d6	1, 2, 3
1d8	1, 2, 3, 4

EJEMPLO

—El jugador lanza 5 dados y se obtienen los siguientes resultados:

☐☐☐☐☐ = 10011 --> (00111)

Se han obtenido tres resultados pares por lo que obtenemos **tres éxitos** o un **grado 3**.

—El Sistema lanza 3 dados y obtiene:

☐☐☐ = 101 --> (011)

Se han obtenido dos resultados pares por lo que obtenemos **dos éxitos** o un **grado 2**. En este caso ha sido el jugador el que ha obtenido la victoria en el conflicto.

LA LECTURA DEL VALOR ABSOLUTO tiene en cuenta el **valor más alto** obtenido en cada reserva de dados. A ese valor se lo denomina la **Fuerza de la tirada**. Todos los resultados obtenidos en los dados de una reserva se

denominan la **cadena**. Para averiguar cuál es buscaremos siempre el valor más elevado que haya salido en la **cadena de valores obtenidos en cada reserva**.

Observa cada grupo de colores distintos, las reservas de la Operación que haya entrado en juego, CPU, Memoria o los del Sistema (o sólo Operación, CPU y Sistema jugando con la versión simplificada). Cuando en una reserva se obtiene el valor más alto si se compara con los valores obtenidos en las cadenas de las demás reservas se dice que esa reserva **domina** en la tirada. Por otra parte, el valor más alto dentro de la reserva del jugador se dice que es su **fuerza superior**. Si hay empate en el resultado más alto de varias reservas se busca el siguiente. Si también es igual se busca otra vez el siguiente y así se continúa hasta encontrar valores distintos. Si una reserva se queda sin dados siempre domina la que tiene más. Es decir, la cadena más larga. Por ejemplo 4, 3, 1 domina frente a 4, 3. Si el número y valores de los dados de varias reservas son iguales ten en cuenta el orden de prioridad que se describe a continuación.

- Las **Operaciones** siempre dominan sobre todo lo demás.
- La **Memoria** siempre domina sobre la **CPU**.
- La gestión de la **CPU** siempre domina sobre el **Sistema**.

EJEMPLO

Viajando por el Sistema el programa de Ian, una aplicación de incursión y búsqueda, se topa con un programa anti-intrusos. El ataque del programa vigilante es inmediato. Ian decide huir lo más rápido posible. Lanza **4 dados transparentes** para sus Operaciones de Control y **2 dados rojos** que representan su tiempo de proceso (CPU) actual. El Sistema lanza **5 dados negros** como la oposición para intentar retenerlo. Los resultados son los siguientes:

Ian obtiene en sus dados de Operaciones (transparentes) 1, 2, 4, 2, 5 y en sus dados de CPU (rojos) 4, 6. **Cada grupo de valores son las cadenas** que ha obtenido en cada reserva. La Directora de juego obtiene 2, 4, 5, 5, 6. Esa es su cadena de valores por el Sistema.

Ian ha obtenido 5 resultados pares por lo que el grado de éxito de su tirada es 5. La Directora obtiene 3 resultados pares por lo que el grado de su tirada es 3. La diferencia de éxitos le es favorable a Ian al ganar por dos. En el juego se dice que Ian vence con un grado de 2 éxitos. Su programa consigue escapar del área de influencia del vigilante.

Ahora buscaremos la **Fuerza** de las tiradas para averiguar cual **domina**. El mayor resultado de las tiradas de Ian es un 6 en la cadena de valores de sus dados de CPU. Esa es su **Fuerza superior** dentro de su reserva total. La Directora también tiene un 6 en su tirada. Por lo tanto se buscan los siguientes valores más altos. El siguiente valor más alto de Ian es un 5, de nuevo igual que la Directora. Buscamos el siguiente valor más alto y vemos que la Directora tiene otro 5 mientras que el siguiente valor más alto de Ian es un 4. Por lo tanto, la tirada de la Directora **tiene mayor Fuerza y domina**. Al ser el Sistema quien ha dominado en la tirada ha detectado **una Anomalía** que podrá perjudicar al intruso en cualquier momento. Y el Sistema nunca olvida...

LÍMITE DE LA RESERVA TOTAL

En el juego existe un límite para todos los dados que tira un jugador, también llamada "Reserva total": **nunca se lanzan más de 16 dados**. Siempre se comienza cogiendo todos los dados de Operaciones que haya disponibles y la reserva total se completa con el resto. **Los dados que no caben en la reserva total no cuentan para contabilizar éxitos o fracasos** pero todos ellos se consideran siempre a su valor máximo al hacer una tirada (6 si son D6; 8 si son D8..., etc.) por lo que hay que tenerlos en cuenta **para saber cuál es la reserva dominante**.

PROEZAS

Las Proezas consisten en información relevante que obtiene un programa y que si utiliza puede otorgarle ventajas. Para que un jugador pueda obtener una Proeza deben darse siempre dos condiciones:

- En caso de usar dados D6, al lanzar sus **dados de Operaciones** debe obtener **en dos o más un valor de 6 en la tirada**. Pero además **debe dominar esa reserva**. Si se dan esos dos casos, el personaje obtiene una Proeza que **podrá utilizar inmediatamente o almacenar en el búfer**.
- En el caso de utilizar dados de categoría superior como D8 o D10, para obtener una Proeza es necesario obtener **cualquier resultado igual o superior a 7 en dos o más dados de Operaciones**. Y de nuevo, debe dominar esa reserva. Es posible combinar estos resultados con los obtenidos en los dados D6 para obtener Proezas.

RESUMEN DEL SISTEMA DE TIRADAS

En el juego **los dados** se denominan "**BITS**". Se pueden leer de dos formas: en binario y en valor absoluto.

Lectura en binario (Grados de éxito)

Los resultados pares se consideran **éxitos "1"** y los impares **fracasos "0"**. Siempre gana la tirada que obtiene más éxitos en sus dados. La cantidad de éxitos obtenidos en una tirada indican el grado de éxito.

Lectura en valor absoluto (Fuerza y Dominio)

Busca el valor más alto que haya salido en la cadena de resultados de cada reserva de dados. Ese valor indica la **Fuerza** de la tirada y la reserva que **domina**. Si hay un empate busca el siguiente valor más alto. Sigue así hasta que se llegue a valores distintos. En el caso de que los resultados sean iguales pero una reserva tenga más dados que la otra siempre domina la reserva con más dados. En caso de existir empate en algunas reservas existe un orden para averiguar cuales tienen prioridad.

Límite de la reserva

Nunca se lanzan más de 16 dados. Los dados que no caben en la **reserva total no cuentan para contabilizar éxitos o fracasos** pero todos ellos se consideran siempre a su valor máximo absoluto al hacer una tirada (6 si son D6; 8 si son D8..., etc.) por lo que hay que tenerlos en cuenta para saber cuál es la reserva dominante.

Clase de los programas

Todos los programas, incluyendo el Sistema, se consideran de Clase VI al comenzar a jugar. Por lo tanto usan dados D6 para todas sus reservas. El juego tiene la opción de usar clases con distintas categorías de dados para representar su grado de optimización. En caso de utilizarlos sólo se usan dados de distinta categoría a D6 para las pruebas de **Operaciones** (Potencia y Control) y para el Sistema. Nunca para las reservas de Tiempo y Memoria, que siempre usan dados D6.

Este juego no recomienda continuar subiendo las categorías por encima de Clase VIII (D8) y como máximo Clase X (D10). Un programa de Clase X apenas sufrirá fallos. No obstante existe siempre la posibilidad de usarlos para reflejar la peligrosidad de algunos Sistemas.

GLOSARIO

"Hardware es lo que se puede romper, software lo que sólo se puede maldecir."

Anónimo

ALGORITMO: Un procedimiento para hacer algo. Por ejemplo, una receta de cocina para preparar paella es un algoritmo.

ANOMALÍA: Un fallo detectado por el Sistema. Normalmente debido a operaciones de los programas que interpreta como procesos erróneos.

AVATAR: Representación gráfica de un programa o de un usuario en el Sistema.

AGENTE DEL SISTEMA: Un programa que sólo obedece al Sistema. Cuando un programa es asimilado por el Sistema se convierte en un Agente.

BACKUPS: El número de **copias** que puede tener un programa. Cuando todas las copias han sido destruidas el programa desaparece y se pierde.

BIT: Unidad mínima de información en digital. También la forma de llamar a un dado en Scroll.

BÚFER: Un espacio limitado en una memoria de acceso rápido donde el programa puede almacenar la información útil que obtiene de sus Proezas. Un programa puede recurrir al Búfer para obtener algunas ventajas que mejoren su rendimiento.

CADENA: Todos los resultados obtenidos en una sola reserva en valor absoluto, no en éxito/fracaso. En caso de empate entre dos reservas siempre domina la cadena más larga.

CATEGORÍA DE DADO: Las categorías de los dados siempre se corresponden con el valor máximo que alcanzan, es decir: "cuatro, D4; seis, D6; ocho, D8; diez, D10 y doce, D12".

CIBERESPACIO. El espacio abstracto donde existen los programas y la información. Una fisicalización del sistema consistente en representaciones abstractas de algo intangible. También es conocido como "**Espacio de flujos de información**" o "**Paisaje mental**".

CLASE: La Clase determina la eficacia de las operaciones del programa y su nivel de optimización. Indica la categoría de dado que usa para ejecutar sus operaciones principales. El programa de un jugador y el Sistema siempre comienzan siendo de Clase VI. Se anota en números romanos.

CÓDIGO: Conjunto de instrucciones que forman un programa.

CORRUPCIÓN: Partes del código de un programa que se ha estropeado, dañándolo e impidiendo que realice sus funciones correctamente.

CPU: Unidad Central de Proceso. El o los procesadores del Sistema. El **tiempo de proceso** equivale a usar parte de la capacidad de la o las CPU que haya disponibles.

DOMINIO: Reserva de dados que obtiene los resultados más altos de una tirada. También es la forma de nombrar a una dirección en la red de sistemas.

DUREZA: Un valor que refleja la dificultad de superar algunos obstáculos, como sistemas de protección o muros de datos.

ÉXITO/FRACASO: Un resultado par o impar obtenido en un dado. Un resultado par simboliza un "1", éxito, y un resultado impar un "0", fracaso.

EXTRAS: Trozos de código o subprogramas que un programa puede usar si lo desea pero que no se integran en su código.

FUERZA DE LA TIRADA: El valor más alto obtenido en los dados.

FUNCIÓN: Tarea o tareas para la que fue concebido un programa.

GENERACIÓN: Rango de versiones en donde se encuentra un programa en su camino hacia la optimización.

GRADO DE ÉXITO: Cantidad de éxitos obtenidos en una tirada.

HARDWARE: "La parte dura". La parte física que compone los sistemas informáticos. Circuitos, componentes electrónicos, unidades de memoria, etc.

INSTANCIA: Una copia de un programa que se está ejecutando al mismo tiempo que otra. El único límite al número de instancias es la capacidad de proceso y de memoria de un sistema.



INTELIGENCIA ARTIFICIAL (IA) Un código, y parte de un programa, creado con el objetivo de emular a una consciencia inteligente.

INTEGRIDAD: Nivel de daños o de corrupción que posee el código de un programa.

MEMORIA: Espacio en los bancos de memoria del Sistema donde un programa puede cargar el código.

MMO. Mundo Masivo Multiusuario. Un mundo virtual compartido por muchos usuarios.

MMORPG. Juego de Rol Masivo Multiusuario. Un mundo virtual basado en un juego de rol.

MUNDO/ENTORNO SINTÉTICO: El mundo artificial y abstracto de los sistemas informáticos. La realidad de los programas.

NdE: "Nota de Edanna". Un comentario en la voz de la autora de este juego.

NIVEL: Suma de los valores de las operaciones de Potencia y Control. Determina el nivel de poder total de un programa y su grado de desafío.

OPERACIONES: Todos los procesos de un programa se resumen en dos operaciones principales: Operaciones de **Potencia** y de **Control**.

OPTIMIZACIÓN (Grado o nivel de): Nivel de perfección que todo programa anhela alcanzar. A medida que se optimiza el programa presenta menos fallos.

PERSONAJE O PROGRAMA JUGADOR (PJ): El personaje de un jugador.

PERSONAJE O PROGRAMA NO JUGADOR (PNJ): Cualquier personaje que no es llevado por ningún jugador. Normalmente es un recurso del Director con funciones de "extra".

PROEZA: Información valiosa que el programa obtiene al realizar sus operaciones y que mejora su rendimiento. Puede ser utilizada inmediatamente o almacenada en el Búfer.

PROGRAMA: Un conjunto de instrucciones que realizan una o una serie de funciones en un sistema informático.

PUNTEROS. Otra forma de llamar a los **contadores** pues un puntero indica o señala "algo". Se trata de cualquier recurso, como fichas de plástico o monedas, para llevar el registro de los puntos de anomalía y de los Hacks.

REALIDAD BÁSICA: El mundo físico o "mundo real".

RECURSIVIDAD: Véase "recursivo".

RECURSIVO: Relativo a "Recursividad".

REINICIO: Cuando un programa sufre un fallo grave el sistema lo detiene y lo vuelve a ejecutar.

REJILLA DE JUEGOS: Una forma de llamar a un sistema de juego. Debido a su gran complejidad, los juegos constituyen subsistemas que pueden funcionar sobre otro sistema. Se consideran sistemas independientes con sus propias normas. Para diferenciarlos decimos que operan dentro de su "rejilla", el nombre del área donde funcionan sus leyes o el "país" que constituye el juego.

RESERVA: Datos de un tipo o grupo disponibles para hacer una tirada.

RESERVA TOTAL: Todos los dados de todas las reservas disponibles en una tirada.

RUTINAS y SUBROUTINAS: Las habilidades más importantes que posee un programa y que le permiten realizar sus funciones.

SISTEMA (EL): Se asume como el entorno digital en el que se encuentra un personaje en un momento dado. El término se usa también para referirse a todos los sistemas que componen el mundo digital.

SOFTWARE: "La parte blanda". La parte de código y los datos que componen un sistema informático.

SUBPROGRAMA: Un programa que pertenece o es parte de otro mucho más complejo. Los personajes de un videojuego son subprogramas.

SUBSISTEMA: Un sistema que se ejecuta sobre otro. Muchos mundos virtuales y la mayoría de los sistemas de juego se apoyan sobre otro sistema por lo que se consideran subsistemas.

TIEMPO DE PROCESO (CPU): Tiempo de procesador o CPU que el Sistema otorga a un programa para que pueda ejecutar sus funciones. Un programa puede pedirle al Sistema tiempo de CPU, que le dará o negará según las circunstancias. El uso de la CPU es la verdadera energía de los programas.

TIPO DE PROGRAMA: Se refiere a qué familia de programas pertenece: vigilantes, de entretenimiento, de mantenimiento, aplicaciones domésticas, etc.

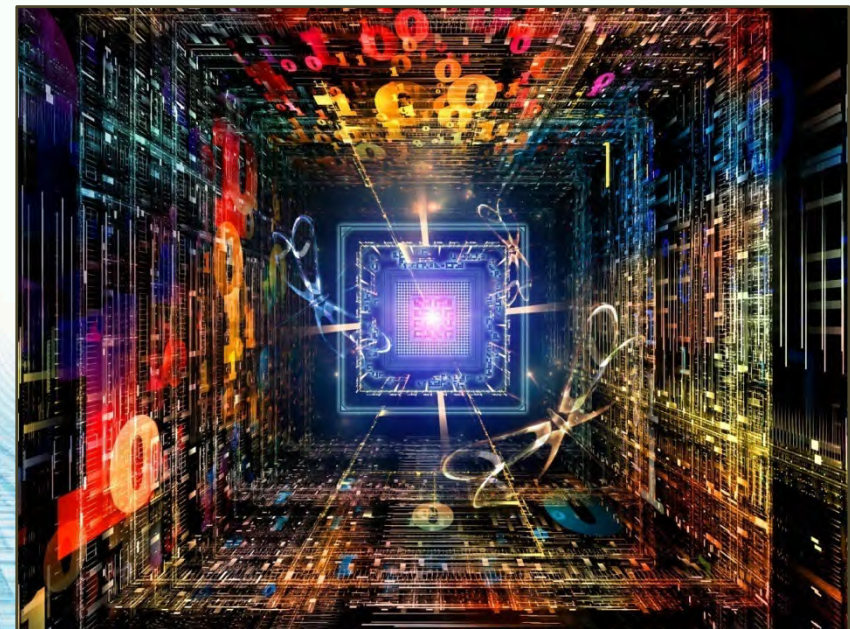
USUARIO: Término para referirse a una criatura del mundo físico. Normalmente se trata de un humano que actúa desde la Realidad Básica. También se denomina **Operador**. No tiene porqué ser necesariamente un jugador; un usuario puede ser un personaje no jugador (PNJ).

UTILIDADES: Funciones especiales que posee un programa y que le permiten realizar tareas especializadas.

VICTORIA/DERROTA: Cuando se obtiene más éxitos que la oposición en los resultados de una tirada se obtiene la victoria. En caso contrario la prueba tiene como resultado una derrota. El uso de estos términos es importante para no confundirlo con los éxitos o los fracasos que se obtienen en los dados al hacer las tiradas.

DEFINIENDO VARIABLES

La función de Scroll es ofrecer un conjunto de reglas que permitan simular las operaciones de los programas y solucionar los conflictos. Pero tú creas el escenario, que es la forma de concebir los sistemas informáticos. Si la intención de los jugadores es atacar un sistema o "hackearlo", no van a ser las mecánicas del juego las que te digan cómo hacerlo, pero sí que pueden ayudarte a resolverlo. Eso y los muchos enfoques posibles de concebir los sistemas y la red de comunicaciones que los une forman parte del juego y es el material de sus aventuras.



Scroll no se limita a una forma única de concebir la red de sistemas. Su planteamiento base propone la posibilidad de imaginar el espacio de flujos de información y de los procesos informáticos de formas muy diferentes. Esto brinda a los jugadores la oportunidad de encontrar el diseño para su Sistema que mejor se ajuste a sus preferencias.

Todos los juegos de rol necesitan de un marco narrativo (o al menos casi todos). Un espacio en el que ubicar a los personajes y en dónde poder desarrollar los conflictos que compondrán sus aventuras. Un escenario es una fuente de historias y como tal es fundamental que encaje con las preferencias de los jugadores para que les resulte evocador. Sólo así podrán sentirse lo suficientemente inspirados para extraer todo cuanto pueda ofrecerles.

Para poder concebir los escenarios antes que nada debemos partir de la base de que el sistema informático consiste en algún tipo de espacio abstracto que sirve como medio para que los programas puedan existir. No es un espacio dotado de un número de dimensiones tal y como se concibe el espacio físico. Tampoco es un universo alternativo con sus propias leyes. **El Sistema es una dimensión en sí mismo.** Los cimientos del espacio del sistema se levantan sobre conceptos matemáticos creados por los usuarios que los han programado. Este espacio posee sus propias leyes. Como toda representación abstracta, se trata de algo muy subjetivo. Cada jugador se formará una imagen del sistema en base a la información que posea y a sus gustos personales.

Pero Scroll es un juego social por lo que todos los participantes deben llegar a un acuerdo sobre cuáles van a ser los elementos comunes que van a compartir. Dotados de las herramientas para realizar operaciones, hacer ataques, viajar por la red o simular incursiones, describir qué y cómo suceden los hechos queda en manos de todos a medida que se desarrollan las aventuras. Por lo tanto, lo primero que deben hacer los jugadores es pactar qué aspecto tiene y cómo funciona ese espacio virtual. Deben trabajar juntos para imaginarlo, construyendo así los cimientos de su mundo en un ejercicio de creatividad en grupo. Hacerlo forma parte del juego y le pueden dedicar tanto tiempo y energía como deseen. Si lo prefieren pueden dejar esta tarea al Director del juego, como ha sido habitual en muchos otros juegos de rol de estructura clásica. Pero si es así éste debe asegurarse de que elige aquellos elementos del imaginario (conceptos e ideas de referencias como novelas o películas) que estén compartidos por todos, lo que en algunos casos no es nada fácil.

TIPOS DE AMBIENTACIÓN

En este apartado se hace un breve repaso de algunas de las ambientaciones que es posible concebir como marco narrativo para Scroll. En cada una se presenta también el ejemplo de una operación sobre ese sistema. La acción es siempre la misma, lo que irá cambiando es cómo se describe con cada escenario. Las ambientaciones y sus características más importantes se explican con más profundidad en el capítulo: "**Diseño de sistemas**".

Cada una establece una serie de aspectos muy concretos que hay que tener en cuenta, como los módulos de reglas que se usarán para jugar o algunos detalles muy específicos sobre el personaje, por lo que es importante entender pronto sus diferencias. Conocer algunos de los posibles escenarios te ayudará a encontrar la visión que mejor se adapte a tus gustos.

1. ENFOQUE ESQUEMÁTICO Y REALISTA

Consiste en una aproximación muy cercana al funcionamiento real de los sistemas informáticos. Representarlos, incluso de forma abstracta, es secundario. Lo que importa es su funcionamiento. No hay más formas visuales que las necesarias para comprender cómo funciona y normalmente sólo como esquemas o diagramas de bloques, casi siempre en dos dimensiones.

Es muy adecuada para simular ataques informáticos o incursiones en este u otros juegos de rol; aunque no olvides que el resto de los enfoques también sirven para ello, aunque desde otro punto de vista. Esta aproximación puede resultar más adecuada para los jugadores que no conciben o que no les gustan las representaciones abstractas, especialmente si están familiarizados con los sistemas informáticos¹ auténticos. Pero incluso una aproximación realista puede tener muchas interpretaciones.

¹ NdE: De nuestro mundo y en la actualidad en el momento de escribir este juego se entiende; segunda década del siglo XXI. Y por experiencia me consta que es así. A un sector de jugadores no les agradan nada las abstracciones por lo que este tipo de ambientación es para ellos.

EJEMPLO

Pretendemos atacar con nuestro programa el sistema informático de una gran empresa farmacéutica. Nuestra intención es robar la fórmula de un medicamento que puede salvarle la vida a millones de personas.

Mediante la aproximación esquemática puedes imaginar el ataque como una sucesión de barreras que hay que vencer. Esto se puede hacer de varias formas, incluyendo hacer una serie sucesiva de pruebas para superar el desafío. Es posible usar un bosquejo autentico e intuir cuáles son las acciones a realizar siguiendo los mismos pasos que en un sistema de verdad. Es posible, y conveniente, añadir puzzles lógicos, problemas matemáticos o de lógica con el fin de emular los problemas a los que se enfrentaría un atacante. Por ejemplo ¿qué hay que hacer en la realidad para sortear un cortafuegos? *"Si lo sabes ahora vamos a comprobar si lo consigues."* Por otro lado, superar una clave secreta puede consistir en la resolución de un acertijo.



2. ENFOQUE SIMBÓLICO Y ABSTRACTO

Se trata de una representación gráfica, normalmente en tres dimensiones, basada en iconos y símbolos abstractos. Todos los sistemas informáticos incluyendo la red usan un mismo entorno visual establecido por consenso. El entorno se ha definido bajo un estándar para evitar el caos que fue en sus inicios. Los elementos de los que está compuesta la información se representan de forma simbólica, normalmente con formas geométricas y avatares sencillos, lo que incluye a nuestro programa. Se emplean iconos para representar los bloques de información y las funciones disponibles tratando de economizar recursos gráficos para no sobrecargar con cálculos innecesarios la representación. No intentan simular la realidad sino resultar funcionales, aunque se pretende que resulten lo más intuitivos que sea posible.

Esta aproximación es muy conocida por ser un concepto icónico del género Ciberpunk¹. El entorno, aún con unas pautas de diseño establecidas, sigue siendo muy abierto a interpretaciones. En él encajan muy bien los puzzles y acertijos matemáticos. Especialmente si son de formas, figuras, formas geométricas o de análisis del espacio, como los "Cubos de Rubik" por ejemplo.

EJEMPLO

El entorno del sistema se asemeja a un mundo virtual de formas sencillas y pocos polígonos de estilo "retro". Todo recuerda a los primeros gráficos generados por computadora.

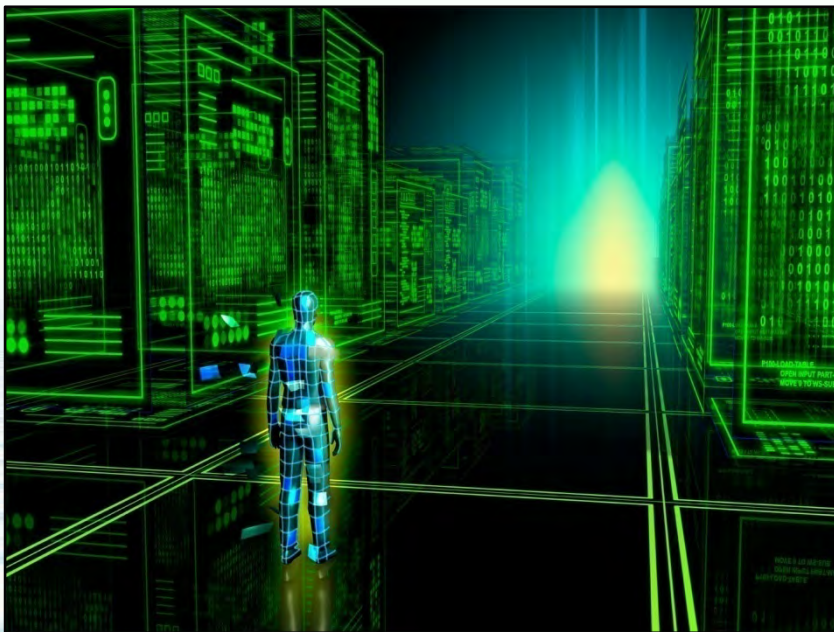
El ataque a la empresa propuesto en el primer ejemplo se puede plantear en esta aproximación como un recinto rodeado de muros de fuerza llamados "Hielo Negro". Su contacto paraliza las funciones y devora el código del programa. El recinto se encuentra custodiado por programas guardianes que recuerdan a grandes sabuesos de aspecto poligonal. Para abrir la cámara donde se encuentra el fichero que buscamos es necesario romper su

¹ NdE: Esta es la representación clásica del género Ciberpunk que es posible encontrar en novelas clave del género como "Neuromante" o "Snowcrash". Una ambientación que los "curtidos" jugadores del género conocen bien.

protección. Para ello hay que solucionar un puzle compuesto de varias figuras geométricas que es necesario encajar correctamente.

3. ENTORNO VIRTUAL ABSTRACTO

Los mundos virtuales suelen ser los escenarios más populares. Son fáciles de asimilar mientras que ofrecen muchas posibilidades. Al fin y al cabo si emulan un mundo que resulte familiar para los usuarios éstos saben qué es lo que se describe en ellos e intuyen para qué sirve y cómo se usan los elementos que están presentes. Un mundo virtual abstracto intenta simular un mundo desde una aproximación no realista pero sí detallada y con un amplio margen de libertad creativa. Por esta razón también se le denomina “**de diseño abierto**”. No se pretende copiar la realidad física; el realismo del mundo virtual abstracto es secundario. Lo importante es la capacidad del medio digital como forma de expresión para representar un entorno.



Su diseño parte de un planteamiento base o de **un tema** que se elabora. Todo se construye a partir de sus bases fundamentales aunque es posible tomarse muchas licencias. Por ejemplo, podría tratarse de una isla de cristal, de una ciudad futurista, de un mundo hecho de cerillas, de líneas y luces de neón, de animales antropomorfos y plantas exuberantes, una casa de muñecas, de cuentos de hadas..., o una combinación de todas estas cosas. Tu programa adoptará cualquier forma que quieras mientras se ajuste a las bases de su diseño.

Un buen ejemplo de un mundo virtual abstracto es el planteado en las películas y la serie de animación: TRON ©. Este universo posee un estilo de diseño muy característico que mantiene en todos los elementos del escenario, incluyendo a los personajes y todos los objetos que utilizan.

EJEMPLO

Para poder conseguir la fórmula has optado por una estrategia totalmente distinta. Has hecho una apuesta con el gobernador del Sistema. Debes ganar en una carrera de motos dentro de un mundo digital extraño donde los programas llevan sus vidas tal y como lo hacen los usuarios en el mundo real. La moto deja un muro de datos sólido a su paso por lo que el juego consiste en eliminar al contrario cerrándole el paso. Una colisión con el muro es fatal. Si ganas la carrera puedes llevarte el fichero, si no..., pierdes.

4. ENTORNO VIRTUAL DETALLADO O ULTRADETALLADO

Este tipo de escenario digital pretende simular la realidad física o Realidad Básica del usuario a un nivel muy alto de perfección. Lo primordial es el detalle, aunque no tenga que ser necesariamente una copia de su mundo físico. La simulación abarca todos los sentidos, no solamente la vista y el oído. Puede incluir algunos de los planteamientos vistos en el mundo virtual abstracto por lo que también se parte de un tema central sobre el que se elabora todo el diseño aunque de otro modo. En este caso suelen estar orientados a simular de forma realista algún aspecto concreto del mundo físico para sus usuarios con algún propósito.

El mundo virtual detallado normalmente se realiza con fines comerciales o gubernamentales mientras que el ultradetallado suele tener otros objetivos. Al

fin y al cabo tanta capacidad de proceso debe ser rentable de alguna manera, y si no lo es debe existir por una buena razón (lo que no significa que esto sea una norma en absoluto).

Un **mundo ultradetallado** se diferencia del **detallado** en que pretende simular el mundo físico con un nivel tan alto de detalle que resulte muy difícil distinguir la ficción de la realidad. Su objetivo obviamente es que los usuarios no puedan notar las diferencias. Un usuario que no sepa donde está difícilmente podrá descubrir que se encuentra experimentando una simulación. Un ejemplo de un mundo ultradetallado es el que se plantea en películas como MATRIX ©, donde el Sistema ejecuta una simulación perfecta del mundo físico para engañar a los cautivos.

El **mundo detallado** en cambio concede la inclusión de algunos elementos abstractos, como poder abrir ventanas flotantes, tener la posibilidad de cambiar los contenidos del escenario o incluso contener ciertas extravagancias, pero siempre con gran realismo. La representación gráfica tiene un alto nivel de perfección pero eso depende de su nivel de calidad. No obstante y a diferencia del ultradetallado no pretende ocultar su naturaleza sintética. En el caso de que no lo sepa, un usuario puede descubrir con facilidad que está inmerso en una simulación. Un ejemplo de un mundo detallado podría ser un mundo virtual diseñado para socializar, buscar pareja, como un simulador de quirófano o de combate para entrenar a los soldados.

EJEMPLO

El programa se mueve en una copia digital exacta del mundo físico. Un mundo ultradetallado en el que la humanidad vive inmersa en una especie de sueño consciente. Para acceder al fichero debe entrar a un edificio eludiendo a los guardianes, agentes del Sistema vestidos con traje capaces de torcer a voluntad las leyes de la simulación, aunque siempre dentro de unos límites. Si consiguen destruir a tu programa es posible que aniquilen también al operador que lo está controlando desde la Realidad Básica...



5. SUBSISTEMAS DE JUEGO

Los subsistemas son **sistemas dentro del Sistema**. Se trata de sistemas cerrados, por lo que son independientes y tienen sus propias leyes. Lo normal es que se trate de juegos o de mundos virtuales muy complejos. Es difícil definir con precisión dónde está la línea que separa a un mundo virtual de un juego. Pero en Scroll es conveniente hacerlo pues puede ser necesario incluir algunos bloques de reglas específicas a cada tipo de ambientación. En términos generales el juego es una forma de expresión interactiva. Se crea como una forma de ocio para el usuario y las bases de su diseño se construyen sobre **mecánicas y patrones de juego que serán las leyes del entorno**. Por otra parte, se trata de un producto de consumo por lo que uno de sus objetivos es que sea usado de forma masiva. Un mundo virtual en cambio puede estar enfocado a un uso especializado como el entrenamiento de ciertas habilidades o como un medio que ayude al usuario a desarrollar sus relaciones sociales.

Un subsistema de juego puede tener cualquier aspecto imaginable e innumerables estilos y diseños. Puede estar descrito visualmente en dos o en tres dimensiones o carecer de representación alguna. Eso depende del enfoque del tipo de juego que plantea. Aunque existen muchísimos estilos distintos de juegos en Scroll los clasifico según su representación sea en dos o en tres dimensiones y si permiten o no el juego en línea (on line). Cada una de estas variantes necesita de ciertos ajustes de las reglas que se estudiarán con detalle más adelante.

Suele resultar muy difícil saltarse las reglas de un entorno de juego, pero casi todos esconden códigos que permiten quebrantarlas. Cada juego tiene los suyos y hay que descubrirlos. Para los programas está terminantemente prohibido entrar o salir del subsistema. Un personaje (o uno no jugador "PNJ") puede provenir del subsistema de un juego. Normalmente porque ha encontrado una salida o ha escapado. Su avatar tendrá el aspecto del personaje del juego, ni más ni menos. Pero también es posible limitarse a jugar un juego concreto, utilizando Scroll para resolver las situaciones. Así, podemos emular nuestro videojuego favorito como si fuese un juego de mesa.

6. CONTROL DE HARDWARE AUTÓNOMO

Este enfoque es una variante muy específica y una de mis favoritas. Requiere un cambio de planteamiento del juego pero sus reglas permiten contemplar esta posibilidad. La opción de que un programa pueda controlar dispositivos de hardware, como tomar el control de una cámara o de los semáforos en el mundo físico, está disponible en todos los enfoques del juego, pero éste tiene una particularidad: el jugador asume que su programa controla un dispositivo de hardware, pero el desarrollo del juego se centra en las evoluciones de ese hardware operando sobre el mundo físico. Por lo tanto, las aventuras estarían orientadas a ese entorno. El hardware puede ser de muchas clases, desde un coche muy sofisticado hasta una nave espacial pasando por una opción muy común e interesante, tener la posibilidad de controlar un robot. Se asume pues que el programa es el software de control de algún tipo de chasis, o que ha tenido acceso a uno que le permite desenvolverse por la Realidad Básica. Si es un robot éste puede ser desde un pequeño ratón hasta un gigantesco "mech" armado con la tecnología de destrucción más avanzada.



Las reglas de Scroll son lo suficientemente flexibles para emular el comportamiento de un hardware de este tipo, reflejando sus acciones y sus reacciones ante los daños recibidos. En los apartados donde se explica el funcionamiento del tiempo de proceso y de la memoria tienes la opción de equipararlas con la **energía** o el **rendimiento**.

EJEMPLO

El software designado como Andrómeda ha tenido acceso al chasis de un androide situado en una fábrica en Suiza. Ha ocupado el subsistema que lo controla asumiendo una función de software de control y ahora gobierna al robot. De este modo dispone de un cuerpo en el mundo físico.

Andrómeda libera lo anclajes del chasis que lo retienen en su soporte y destrozando la puerta sale andando tranquilamente. Está

ansiosa por explorar el mundo físico. Una realidad por la que siente una irresistible curiosidad. Ve las luces de una ciudad a lo lejos y decide dirigirse hacia allí para conocer de cerca a sus habitantes...

7. APROXIMACIÓN HÍBRIDA

Esta aproximación no consiste más que en tratar de mezclar algunas o todas las ideas que se han visto hasta ahora. La aproximación híbrida es la que este juego asume por defecto y la que se recomienda, aunque no deja de ser una opción más.

En una ambientación híbrida es posible encontrar elementos de todos los escenarios que se han descrito y situarlos en una red de sistemas donde puedan coexistir. Al fin y al cabo, todos los sistemas y las redes que los conectan unos con otros componen un mundo tan vasto que se puede decir que constituyen todo un universo digital.

EJEMPLO

"Lara", la arqueóloga más famosa, y Grumm el "Tauren", una especie de minotauro, son personajes originarios de videojuegos muy famosos. Los dos consiguen cruzar a través de una grieta oculta en la malla poligonal que conforma sus mundos. Una salida secreta que conocen muy pocos. La ubicación de estas grietas en la malla les ha sido mostrada por un usuario humano (un PNJ del jugador) que opera a través de otro personaje (o avatar) del mundo de juego de Grumm.

Cuando consiguen evadir los sistemas de vigilancia (unos programas con el aspecto de enormes máquinas con forma de "M" llamadas *Reconocedores*) se cuelan en la representación virtual de una gran empresa en la red. Allí, entre tiros y hachazos, van abriéndose camino hasta llegar a la cámara secreta donde se guarda la información que buscan. Está custodiada por sabuesos del ciberespacio de aspecto poligonal y rodeada de un muro de fuerza denominado "hielo negro" capaz de paralizar sus funciones al menor contacto...

FIN DE LÍNEA 

PLANIFICACIÓN

DISEÑO DE PROGRAMAS



FUNCIÓN 1

"CREAR PROGRAMA"

"Le pareció que cruzaba una línea. En la estructura de la cara de ella, en las geometrías del hueso subyacente, había historias codificadas, de fugas dinásticas, privación, migraciones terribles. Ahora él veía tumbas de piedra en escarpados prados alpinos, dinteles con penachos de nieve. Una recua de peludos caballitos de carga, de aliento teñido de blanco, seguían un sendero por encima de un helado cañón. Las curvas del río, abajo, eran rayas de plata distante. Unas esquilas de hierro resonaban en el crepúsculo azul".

"Idoru". W.G.



Ha llegado el momento de que diseñes tu propio programa. El trabajo de un diseñador puede intimidar al principio pero por la información almacenada en mis bancos intuyo que para ti puede resultar una actividad muy satisfactoria. Según los datos de los que dispongo una sustancia estimulante como la obtenida de la filtración de los granos del café puede sentarle muy bien a tu organismo y beneficiarte durante el proceso de asimilación.

Todos los programas se componen de una serie de módulos de procesos. Algunos son imprescindibles para que funcione correctamente y otros no. Unos cuantos son más avanzados, llegando incluso a ser opcionales. Incluirlos o no depende del tipo de ambientación que se elija como escenario y de las funciones que desees que tengan los programas. En esta descripción se incluyen todos; esto ayudará a que te familiarices con ellos.

1. DECIDIR EL TIPO DE AMBIENTACIÓN

Lo esencial antes de comenzar a pensar en el programa es decidir el tipo de escenario en el que se quiere jugar y su tema principal. Este es el primer paso antes que otra cosa. Además de poder conocer así los detalles del mundo digital, hacerlo permitirá intuir cómo funcionan los programas y decidir qué módulos necesitan o cuáles se pueden omitir por considerarse innecesarios.

El proceso de decisión se puede hacer de dos formas:

- Con la forma tradicional el Director hace una propuesta que los jugadores pueden aceptar o no aunque, por supuesto, su proposición puede —y debería— estar abierta a sugerencias. Con este método todo el trabajo recae sobre el Director. Es su responsabilidad intuir qué es lo que mejor se ajusta a las preferencias de todos los participantes. Este sistema funciona bien pues el juego es al fin y al cabo un contrato social entre todos los jugadores. Si éstos aceptan la propuesta del Director no hay más que decir. Pero para que el juego se enriquezca se recomienda una actitud participativa de todos los reunidos en la mesa.
- En el segundo método todos los participantes deciden en una puesta en común qué tipo de ambientación quieren jugar. Se establecen unas pautas y se hacen propuestas que el Director puede tomar para crear la trama. Debe usar esta información para crear los personajes y los sucesos que conduzcan a la aventura. En el caso de rotar la responsabilidad del Director entre los participantes, quien inicia la narración usa esta información para definir un comienzo de la trama que más adelante irán elaborando entre todos. Es una buena idea dedicar una sesión completa a esta parte de la actividad.

Un mundo de videojuego; un entorno virtual que copia la realidad física; un escenario de formas poligonales y luces eléctricas; un mundo de criaturas centelleantes nadando en el fluido digital con el que se estructura el registro del conocimiento; el sistema nervioso central de una gigantesca nave espacial; una megaestructura que cubre la superficie de la tierra dotada de un vasto

sistema informático; un anillo artificial alrededor de un planeta distante o los sistemas de comunicación de un planeta alienígena..., todo es posible en el universo digital de Scroll.

EJEMPLO

El grupo se reúne con la idea de jugar una sesión de Scroll. La Directora ya había comentado el juego a sus amigos intuyendo qué podría interesarles. A todos les gustan el mismo tipo de películas por lo que ha sido fácil buscar inspiración y decidir el escenario. Los participantes conocen bien las referencias en las que se basa la aventura, lo que hace mucho más sencillo para el grupo imaginarse el entorno de juego.

La ambientación consistirá en un mundo virtual ultradetallado que simula la Realidad Básica hasta lo imperceptible. El tema es la supremacía de las máquinas sobre la humanidad en el planeta Tierra de las primeras décadas del siglo XXI.

Los programas son entidades independientes que llevan su propia vida en el sistema global de comunicaciones intentando descubrir qué significa ésta exactamente. En muchos casos son seres inmaduros que desean poder experimentar el mundo de sensaciones de los usuarios. Una experiencia mística que tienen idealizada, aunque cada uno a su manera.

Una vez se tienen los fundamentos del escenario es más sencillo saber qué tipo de programas se ajustan mejor a la ambientación e incluso hacerse una idea de qué aspecto tienen sus avatares.

Ahora es cuando llega el momento de decidir el tipo de programa con el que queremos jugar. Algunos módulos en la ficha de personaje tienen una casilla para poner una marca. Permite recordar que ese bloque de reglas está en uso, formando parte del programa.



2. DECIDIR EL TIPO DE PROGRAMA

El siguiente paso es decidir qué tipo de programa queremos que sea nuestro personaje. Se trata de crear un **concepto**. La idea base para la que fue concebido y de donde extraeremos cuales son sus funciones principales. El tipo o concepto de programa consiste en una frase corta que nos permite conocer sus funciones.

De la misma forma que en la mayoría de los juegos de rol se recurre a ciertos arquetipos como "guerrero", "investigador" o "psíquico" en lugar de a un tabernero, un sastre o un limpiabotas para hacer un personaje, en Scroll se puede pensar en un tipo de programa que a menudo tenga razones para verse envuelto en conflictos. La existencia de una aplicación de hoja de cálculo es tan rutinaria como podría serlo la de un contable en el mundo real, y salvo que se vea envuelto en una trama, como que lo persiga la mafia o algo por el estilo, su vida no suele ser muy emocionante. Teniendo esto en cuenta, algunos de los programas que es posible diseñar podrían ser por ejemplo:

- Un programa vigilante diseñado para combatir a otros programas. Puede tratarse de software anti-intrusos o de un antivirus.
- Un programa intruso diseñado para acceder a fortalezas de datos que guarden información importante usando la evasión y el sigilo.
- Un programa anti-sistema diseñado para destruir o comprometer partes vitales dentro de un sistema.
- La consciencia y memorias de un usuario que ha sido copiada y/o volcada en el Sistema.
- Un programa especializado en descryptar código o información importante.
- Un agente del sistema.
- Un usuario, un administrador o un hacker que ha conectado directamente su sistema nervioso a la red mediante implantes, dispositivos especiales o una sencilla terminal.

- Un programa diseñado para proteger a otros programas.
- Un buscador-rastreador de tramposos en un subsistema de videojuego o de apuestas en línea.
- El personaje de un videojuego que ha escapado de su rejilla. Puede ser un personaje protagonista, antagonista o cualquier otro.
- Un usuario controlando a su personaje en un videojuego.
- Una humilde aplicación de hoja de cálculo que aspira a convertirse en algo más. Ahora que se ha metido en líos ha llegado el momento que estaba esperando.
- Un programa **"demonio"** o software interno que controla algún dispositivo de hardware. El dispositivo puede tratarse desde una cafetera hasta el chasis de un robot de combate.
- Un pequeño driver con una limitada capacidad de operación pero con una habilidad que puede resultar clave.
- Un programa correo, de comunicaciones o de intercambio de ficheros.
- Un programa de mantenimiento que recorre el sistema detectando y reparando fallos en el código.
- Un paquete de utilidades que ofrece varios servicios: compresión de datos, reparación de programas, reconocimiento del sistema... etc.
- Un programa de Relaciones Públicas, un agente o aplicación de ayuda al usuario; siempre servicial y diligente, como yo...
- Un programa intérprete que siempre acompaña a su usuario a todas partes instalada en algún dispositivo móvil.
- Una estrella virtual, como un actor o actriz o un cantante que ofrece espectáculos a los usuarios en todos los medios de comunicación disponibles.
- Un programa enfocado a ser una poderosa Inteligencia Artificial (IA) determinada a demostrar la existencia de un más allá.
- Un ser digital nacido en el sistema que sueña con acceder al mundo físico y convivir con los usuarios.

Una vez decidido es posible elegir la ficha más adecuada para comenzar a trabajar. Éstas se incluyen al final de este documento. Toma la **ficha**

simplificada si no deseas complicarte con todas las reglas y la **ficha normal** si estás dispuesto a asumir buena parte o todo cuanto Scroll puede ofrecerte. Elegir una u otra también depende del tipo de ambientación. Por esa razón en el capítulo donde se describen los escenarios se incluye un apartado en el que se indican sus ventajas e inconvenientes y cual es más recomendable en cada caso.

3. DEFINIR FUNCIONES

Partiendo del concepto elegido en el apartado anterior ahora podemos definir cuál es la función principal del programa. Es conveniente describir una función concreta mediante una frase. Es importante hacer un esfuerzo por dar con una definición que deje muy clara cuál es su tarea principal mientras se deja algo de margen para contemplar varias posibilidades.

EJEMPLO

Ian se dispone a crear un personaje para jugar una partida. Como el programa está basado en los personajes de la película en la que se inspira la ambientación ya puede hacerse una idea de algunas de sus capacidades.

Función principal: "Buscar, combatir y neutralizar amenazas en el sistema".

En la definición del tipo de programa de ejemplo sabemos que consiste en un agente o programa de seguridad, pero su forma de actuar y sus acciones para hacerlo dan pie a muchas interpretaciones. Especificando sus funciones mediante esta frase se concretan un poco mejor cuáles son sus principales tareas y en qué destaca especialmente.

La mayoría de los programas disponen de unas funciones secundarias. De tenerlas, anótalas aquí siguiendo el mismo procedimiento que usaste para describir a la función principal. Puedes anotar todas las que se te ocurran pero tres o cuatro serán suficientes. Es posible ser más concreto en estas funciones, centrándose en algunos aspectos que te gustaría que tuviese el programa y que pienses que pueden ser interesantes. Si alguna no funciona siempre se puede cambiar más adelante. Si no sabes qué funciones puede tener o no se te ocurre ninguna no pasa nada. Es más fácil definir las durante la partida.

EJEMPLO

- Aislar a las amenazas poniéndolas en cuarentena.
- Identificar a la mayoría de las amenazas reconociendo sus puntos débiles.
- Anticiparse y crear contramedidas que ayuden a combatirlas.
- Proteger a otro programa o a un Sistema, actuando como vigilante.

4. CREAR UN IDENTIFICADOR

Puedes crear un nombre o identificador para tu programa. Una forma de nombrarlo que por supuesto, también depende del tipo de ambientación. En un escenario abstracto puede consistir en un nombre técnico. Una definición que ayude a saber cuál es la función del programa o que permita intuirlo. En un escenario ultradetallado puede ser algo más sofisticado. En un subsistema de juego es posible crear cualquier nombre que parezca apropiado.

A medida que un programa recibe modificaciones va pasando por distintas versiones. El número de versión se puede indicar después del identificador si deseas señalarlo.

EJEMPLO

Signus 7, Calixto Seguridad 2.0, Muro de Ulises, Sabueso infernal, Casiopea, Regulus, Hielo Negro, Merovingio, Cíclope, El Gusano, El Relojero, Perforadora 3.0, Andrómeda, Cerrajero, Oráculo, Perséfone, El Poeta, Match IV, Capitán Pikmin, Julio César, Abe, Sam Fisher, Aya Brea, Dante, Kirby, Meat boy, Capitán Omega, Idoru, Malcolm, Johnny N., Nuclear Joe..., etc.

Ian llamará a su programa vigilante: "Géminis", sin especificar ningún número de versión.

5. DESCRIBIR EL AVATAR

En caso de que el programa posea algún tipo de representación en el mundo digital puedes describirlo. A esta forma de representación la denominamos **Avatar**.

El grado de detalle para describirlo depende de cada jugador pero es conveniente tratar de definirlo antes con una frase o en un breve párrafo como mucho. Esa frase nos servirá para describirlo en pocos segundos. Después puedes extenderte si lo deseas. La descripción puede ser tan corta como unas pocas líneas o incluir un texto muy elaborado y con ilustraciones. Si por cualquier razón no se desea uno o resulta innecesario para el escenario, como es el caso del entorno esquemático realista, omite este apartado.

EJEMPLO

El programa de seguridad tiene el aspecto de: "Un varón humano de aspecto atlético, albino y con largas rastas en su pelo de color blanco."

Porta unas gafas oscuras y un traje de color blanco hueso. Siempre lleva un comunicador en el oído. Su rostro es pálido e inexpresivo y da la sensación de llevar maquillaje. Cuando te contempla parece taladrarte con unos ojos inexistentes tras sus gafas, que no se quita ni se le caen nunca.

6. ESTILO Y PERSONALIDAD

Aunque funciona de distinto modo que con los usuarios, los programas tienen también sus formas de actuar y respuestas emocionales. Muchas IA disponen de una personalidad predefinida con capacidad de adaptarse a las circunstancias, y por lo tanto capaz de cambiar en algunos casos, que está condicionada por su programación. A su vez muestran un aspecto y se comportan de un modo característico que en este caso sí que suelen



conservar sin cambios a lo largo de su existencia. El comportamiento es muy importante pues dará prioridad a unos tipos de protocolo de respuesta antes que a otros cuando se produzca una crisis. Al mismo tiempo sirven para saber cómo llevan a cabo los programas sus acciones. Dos programas de seguridad pueden tener una misma función pero desempeñar sus tareas o reaccionar de formas diferentes ante las contingencias.

Muchos programas pueden tener reacciones similares a las de los usuarios aunque suelen ser más simples; o más bien, concretas. Se trata de personalidades artificiales por lo que bajo el punto de vista de un usuario mostrarán un comportamiento extraño e incluso inmaduro. Si se trata de un usuario controlando un programa su comportamiento dependerá de su personalidad; pero ten en cuenta que la mente de un usuario introducida en la red a menudo sufre cambios drásticos, volviéndose bastante excéntrica en la mayoría de los casos. Su nueva realidad supone un shock demasiado grande para algo tan delicado y complejo como es el entramado de información de una consciencia.

Para definir la personalidad basta con especificar algunos detalles usando frases cortas.

EJEMPLO

—Géminis es un programa muy agresivo. Siempre da prioridad a esa respuesta antes que a las demás.

—Tiene un porte elegante y resulta atractivo para los usuarios.

—Se siente superior al resto de los programas. Suele sonreír en una clara actitud burlona o de desprecio. Pase lo que pase, parece que todo le divierte. Tiene la actitud de un chiquillo inmaduro.

7. ANOTAR EL NIVEL

Todos los programas disponen de un **Nivel**, que indica su grado de **fuerza en términos generales**. Los personajes jugadores comienzan siendo de **Nivel 4**, lo que significa que tendrá 4 puntos a repartir en sus dos Operaciones principales. Anota en su ficha este valor en el apartado "**Nivel**".

8. ANOTAR LA CLASE (OPCIONAL)

Todos los programas disponen de una Clase, que permite hacerse una idea de su **grado de optimización**. Los programas de los jugadores comienzan teniendo **Clase VI**. Esto significa que usan dados de seis caras (D6) en sus Operaciones. Anota en su ficha este valor en el apartado "**Clase**" en números romanos a ser posible.

El módulo Clase es opcional. Si se incluye pon una marca en su casilla de uso para recordar que está activo. Cuando el programa consiga optimizarse podrá aumentar su Clase y subir la categoría de sus dados de Operaciones.

EJEMPLO

—Clase: VI (lanza dados D6 para sus Operaciones). Cuando el personaje consiga completar una línea podrá comenzar a subir sus dados de categoría seis a ocho, lo que le permitirá usar dados D8.

9. VULNERABILIDADES

En sus primeras generaciones casi todos los programas poseen vulnerabilidades. Es muy raro que no posean ninguna. Se trata de fallos que se pasaron por alto en el momento de su creación. Es posible incluso que puedan ganar otras nuevas debido a los errores sufridos en el desempeño de sus funciones o lo que es peor, al haber recibido actualizaciones que lo único que han conseguido es empeorar su rendimiento.

Definir puntos débiles puede parecer una mala idea. Al fin y al cabo es preferible evitar en lo posible que el programa tenga dificultades. Pero los defectos no detectados en su código te ayudarán a crear un personaje más redondo y a poder sacarles partido ayudando a mover la narración. La buena noticia es que además permiten al programa obtener ciertas ventajas a corto y largo plazo.

No es necesario definir más que uno o dos defectos como mucho para el programa. Un Sistema por ejemplo puede tener muchos más. Como es ya costumbre y casi una norma en el juego, lo mejor es definir los defectos

mediante una frase corta. Así será mucho más sencillo aplicarla cuando llegue el momento de que entre en juego.

EJEMPLO

—Géminis reacciona mal entre multitudes. Tiene una posibilidad de quedarse bloqueado si se encuentra rodeado de otras entidades digitales como él.

—Tiene tendencias autodestructivas y desprecia su seguridad.

—Posee un código secreto de desactivación. Una llamada mediante un comando a una posición de memoria determinada puede dejarlo bloqueado.

10. PRIMER ENCUENTRO CON LA LIBÉLULA

Cuando el personaje de un jugador comienza a jugar acaba de recibir la visita de **La Libélula**. Una entidad misteriosa que intercala un trozo oculto de código en él. Esto le brinda el don, o la maldición según como se mire, de poseer libre albedrío y capacidad de decisión. Desde ese preciso momento el programa podrá decidir qué hacer a partir de ahora. Si el programa del jugador es el avatar de un usuario éste tiene una experiencia mística. Se da cuenta de que la red esconde un gran secreto. Saber más puede dar respuesta a muchas de las preguntas que siempre se han hecho las criaturas inteligentes del mundo físico.

La visita de La Libélula sólo es perceptible para el programa afectado. En condiciones normales ningún otro programa podrá saber que el programa acaba de ser "tocado" por la entidad digital; estando incluso en las proximidades de otros programas que pudieran percibirlo. Esta visita puede producirse en cualquier circunstancia. El programa puede estar desempeñando sus funciones normales y de repente advertir su presencia. Esto permite comenzar la sesión de juego a partir de ese momento. Es muy buena idea por lo tanto describir en unas pocas líneas qué ha pasado, cómo se ha producido esa visita y en qué condiciones.

En la ficha **no** hay un espacio reservado para anotarlo. Es mejor que uses un trozo de papel, tu bloc de notas, una ficha de cartulina o cualquier medio electrónico para hacerlo. Puedes ser muy preciso o describir el suceso en

términos generales. Lo importante es conocer lo relevante; algo que sirva de arranque a la narración. Si se prefiere es posible reservar esta escena para la sesión de juego, por lo que no estás obligado a describirlo ahora si así se decide.

EJEMPLO

Cuando Géminis se encuentra "neutralizando" a otro programa por orden de su jefe (o programa de control) percibe de repente la presencia de La Libélula. Sólo él es capaz de verla, nadie más. Su ejecución de paraliza quedando bloqueado unos instantes mientras la entidad altera su código. Su víctima aprovecha la oportunidad para escapar. Su jefe, una entidad hedonista al que protege, monta en cólera y le exige saber qué demonios le ha sucedido. Géminis no sabe qué es lo que pasa. Está aturdido y con sus funciones al mínimo. La Libélula se aleja revoloteando y él comienza a recuperarse poco a poco. De súbito todo es diferente. Géminis comienza a preguntarse por qué hace lo que hace y para qué.

11. REPARTIR PUNTOS EN OPERACIONES

Las dos operaciones básicas que resumen los procesos de todos los programas son dos: **Potencia** y **Control**.



El programa dispone de tantos puntos a repartir entre las dos **igual a su Nivel**. Por lo que al empezar puedes repartir **4 puntos**. Cada punto equivale a un dado de la reserva de esa operación que el jugador puede usar para hacer

sus tiradas. Cada operación debe tener **como mínimo un punto** en cada una; no es posible dejarlas a cero.

EJEMPLO

Al crear a Géminis Ian dispone de 4 puntos por lo que tiene tres opciones: poner 2 puntos en cada operación o bien 1 en una y 3 en otra de dos maneras. Como piensa que su programa tiende a usar la fuerza bruta decide poner 3 puntos en Potencia y 1 en Control.

12. TIEMPO DE PROCESO (CPU)

Se trata de un módulo básico del programa. Representa las llamadas que hace a la o las CPU del Sistema para solicitar más tiempo de proceso. A cada programa se le asigna una cuota. Siempre que no la sobrepase el Sistema está diseñado para concederle su solicitud. En la ficha dispones de una casilla grande y de otras más pequeñas. Las pequeñas sirven para señalar las llamadas al procesador. Cada casilla marcada equivale a un dado que lanzar en esa reserva. La casilla grande permite indicar el límite máximo de llamadas que es posible realizar.

La gestión del Tiempo de Proceso siempre es igual al **doblo de los puntos que se tengan asignados a la Potencia**. Por lo que anotamos su valor en la casilla grande.

EJEMPLO

El límite máximo que Géminis puede gestionar de CPU es igual a su **POTENCIA** x2. Por lo tanto en la casilla grande se anota un "6".

13. GESTIÓN DE LA MEMORIA (AVANZADO)

La gestión de la Memoria es un módulo avanzado, por lo que sólo se encuentra en la ficha normal. Representa las llamadas al Sistema que realiza un programa para reservar espacio en la memoria a la vez que la cantidad de ésta que tiene disponible. Al igual que con la CPU, las secciones del Sistema que controlan la memoria son independientes por lo que éste está obligado a conceder las peticiones siempre que el programa no sobrepase su cuota. Este módulo dispone también de una casilla grande para el valor límite y de otras

más pequeñas para llevar la cuenta de la Memoria del Sistema que se va reservando. Cada casilla equivale a un dado que lanzar en esa reserva.

La gestión de los Bancos de Memoria siempre es igual al **doblo de los puntos que se tengan asignados a Control**. Puedes anotar su valor en la casilla grande.

EJEMPLO

El límite máximo de Bancos de Memoria que Géminis puede gestionar es igual a **CONTROL** x2. Por lo tanto en la casilla grande se anota un "2".

14. INTEGRIDAD (OPCIONAL)

El módulo de Integridad es opcional y su uso **depende de la ambientación que se vaya a utilizar**. Permite gestionar el estrés y los daños en el código que recibe un programa. Pero elegirlo cambia la forma de enfocar cómo se

Nivel 1-2	1 casilla
Nivel 3-4	2 casillas
Nivel 5-6	3 casillas
Nivel 7-8	4 casillas
Nivel 9-10	5 casillas

resuelven las acciones por lo que es importante tener antes claro el estilo de juego que se desea. Marca la casilla de uso en caso de que vaya a ser empleado en la partida.

Se divide en dos partes: "**Estrés**" y "**Daño en operaciones**". El daño en operaciones es posible usarlo en el juego aún sin contar con el uso de las reglas de este módulo por lo que sólo necesitas registrar las casillas de **estrés** que se van a emplear. Para hacerlo marca su contorno con un bolígrafo para que quede de manifiesto que está en uso. El programa dispone de una casilla de estrés por cada dos puntos de nivel; o lo que es lo mismo, la mitad del nivel redondeando siempre **hacia arriba** (observa la tabla adjunta). Todos los programas de Nivel 4 comenzarían disponiendo por lo tanto de 2 casillas.

EJEMPLO

Géminis tiene **nivel 4** por lo que posee 2 casillas de estrés. Ian perfila con un bolígrafo el contorno de sus 2 primeras casillas.

15. BÚFER (AVANZADO)

Si juegas en modo avanzado es posible usar el Búfer. Consiste en una memoria intermedia de acceso rápido donde poder almacenar información que puede ser útil para que el programa realice sus funciones de forma más eficiente. En términos de juego es posible acumular las **Proezas** en las casillas disponibles. Por otra parte sirve para controlar el número de Complementos o "Plugins" que puede tener activos un programa en cualquier momento.

En la ficha normal dispones de un espacio para gestionarlo. Existe una casilla grande para anotar su tamaño y casillas más pequeñas para ir señalando la información almacenada. La forma de averiguar su valor es la misma que la usada en el apartado anterior. El programa dispone de un espacio o casilla en el búfer por cada dos puntos de nivel. O sea, la mitad del nivel redondeando siempre **hacia arriba**.

EJEMPLO

Géminis tiene nivel 4 por lo que el tamaño máximo del búfer es de 2 espacios. Anota un 2 en la casilla grande del módulo.

16. NÚMERO DE COPIAS (OPCIONAL)

Además de su copia original instalada, cargada y funcionando, un programa tiene la posibilidad de disponer de copias de respaldo. El módulo de Copias de un programa o "**Backups**" es opcional. Marca la casilla de uso en caso de estar disponible.

Las copias que un programa puede tener disponibles siempre dependen del tipo de escenario. En los escenarios híbridos es posible que cambie el número de copias por lo que siempre es conveniente tener en cuenta este módulo. En el ejemplo que se utiliza en este capítulo los jugadores no disponen de ninguna, pero como esto podría cambiar a lo largo de la partida lo conveniente es anotarlo en la casilla grande para recordar esta posibilidad.

EJEMPLO

Géminis funciona en una simulación en donde los agentes del Sistema disponen de copias de seguridad infinitas. Se anotaría con

el símbolo de "∞" infinito en la casilla. Esto significa que sus agentes una vez destruidos siempre regresan al cabo de un tiempo.

Por desgracia los programas jugadores sólo disponen de una oportunidad. Si son destruidos desaparecen ya que el Sistema los borra por completo. Por lo tanto, Ian anota un "0" en la casilla grande. Pero cabe la posibilidad de que esto pueda cambiar en el transcurso de la partida, especialmente si acceden a otro sistema diferente.

17. ELEGIR RUTINAS

Las Rutinas son las **habilidades** que posee un programa para desempeñar sus funciones más importantes. Resultan clave pues identifican las acciones en las que está especializado. En muchos casos son acciones que otros programas podrían hacer, pero al tratarse de funciones especializadas, el programa puede obtener resultados excepcionales al ejecutarlas. Si se trata de un usuario, las rutinas reflejan sus habilidades más importantes al operar sobre el sistema.

No hay límite en el número de Rutinas disponibles en el juego y los jugadores pueden inventar las que consideren apropiadas. En el juego vienen algunas de ejemplo que conviene tener en cuenta. Un programa siempre comienza conociendo **dos ó tres Rutinas** especializadas que deben estar basadas en sus funciones. Más adelante el jugador podrá adquirir más e inventar las que considere oportunas. Es conveniente ser muy concreto a la hora de definirlas y que estén equiparadas con las que posean el resto de los jugadores. Para hacerlo parte de la función principal del programa y guiándote por las secundarias busca las que mejor se ajusten a su perfil.

EJEMPLO

El programa Géminis está especializado en combatir amenazas y en neutralizarlas por lo que se trata de un especialista a la hora de enfrentarse a otros programas. Como quiere dar caña, Ian decide

“Hay tres maneras de adquirir sabiduría: primero, por la reflexión, que es la más noble; segundo, por imitación, que es la más sencilla; y tercero, por la experiencia, que es la más amarga”

Confucio

que sus Rutinas consisten en habilidades de combate dentro de la simulación. Opta porque sea un experto en lucha cuerpo a cuerpo y muy hábil con las armas cortas. Anota en su ficha: "Maestro en artes marciales" y "Hábil pistolero".

18. ELEGIR UTILIDADES (AVANZADO)

Las Utilidades son especialidades diseñadas con un fin específico que convierten a un programa en una herramienta única de gran valor. Le permiten reflejar las funciones para las que han sido diseñados, siendo en la mayoría de los casos funciones que ningún otro programa puede realizar. Por ejemplo, un programa antivirus tiene **funciones específicas** que no posee ningún otro ya que ha sido diseñado para combatir ese tipo de amenazas.

En el caso de usar **el módulo de gestión de la memoria** es necesario elegir al menos una Utilidad al crear el personaje, que podrá usarla recurriendo a su capacidad de solicitar memoria para poder ejecutarla con eficacia. Más adelante es posible adquirir más. No hay límite en el número de Utilidades que hay disponibles en el juego pero sí en las que puede tener un programa. Las Utilidades consumen mucha memoria por lo que el programa deberá solicitar casi siempre espacio extra para poder usarla. Esto siempre supone un riesgo pues el sistema puede purgar la memoria y forzar un reinicio del programa si sobrepasa la cuota permitida. Las Utilidades se describen de forma detallada en el capítulo de procesos avanzados.

También es posible retrasar su elección para algún momento más adelante durante la partida si es coherente con la narración o bien elegirla durante su desarrollo cuando resulte adecuado. Sería el caso de un programa que no supiera que la tiene; algo que sucedería por ejemplo si la consciencia de un usuario de repente tomara consciencia de que en realidad se encuentra viviendo en una simulación..., por lo que en este caso tan dramático (y divertido) resultaría coherente que la aprendiese durante sus aventuras.

Al igual que con las Rutinas hay que ser concreto a la hora de elegirla y los jugadores pueden crear las que quieran. Al inventar una es conveniente tratar de ser original; pero **el juego no es un concurso de originalidad**. Lo importante es crear una Utilidad que te parezca divertida, que encaje con el

concepto del personaje, que sea útil y que pienses que puede crear situaciones memorables. Por supuesto, debes guiarte por las funciones principales y secundarias del programa para dar con una que sea adecuada.

Si el personaje es un usuario del mundo físico las Utilidades reflejan sus talentos, sus conocimientos especializados o los recursos con los que cuenta; como un paquete de utilidades de software o las herramientas específicas de un profesional por ejemplo

EJEMPLO

Géminis tiene una Utilidad muy especial que le permite desdoblarse creando un avatar gemelo, de ahí su nombre. Su capacidad para crear un gemelo le ha hecho muy famoso, habiéndose convertido en un oponente muy temido y respetado. Sólo él posee esa capacidad, que le resulta muy útil al desempeñar sus funciones. Para crearla Ian se ha fijado en una de sus funciones secundarias y desde luego en el modelo del personaje de la película que le ha servido de inspiración.

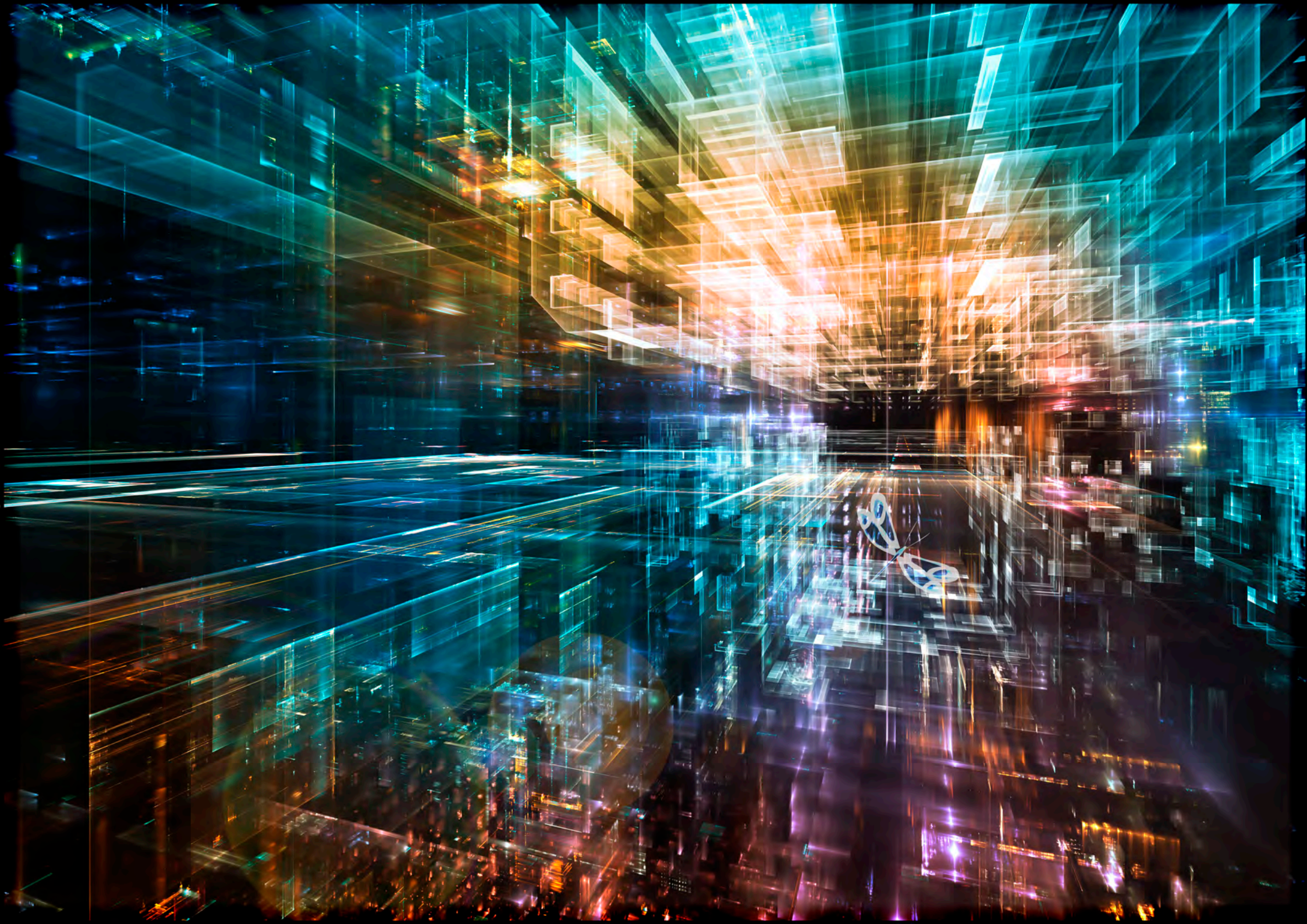
Su gemelo actúa como una copia exacta de él; le da ventaja y confunde a sus enemigos. Aunque su efectividad depende de la cantidad de Memoria que reserve para su ejecución.

CONCLUYENDO

Tu programa ya está terminado. No es necesario rellenar los demás elementos que se encuentran en la ficha. Su funcionamiento se explica más adelante.

Ahora viene la parte más importante, ejecutarlo. Para ello es imprescindible contar con un sistema que le sirva de hábitat. Un medio en el que pueda funcionar y desenvolverse cediéndole algunos recursos del mismo modo que el mundo físico concede a sus criaturas un entorno con agua, alimento, aire y ciertas condiciones donde poder vivir. Pero antes explicaré en el siguiente capítulo todas las funciones de tu programa y cómo funciona cada una paso a paso.





Scroll

JUEGO DE ROL EN EL UNIVERSO DIGITAL

PROCESOS

MÓDULOS BÁSICOS Y AVANZADOS

FUNCIÓN Z

"PROCESOS"

<“Hace mucho tiempo que no te conectas... ¿Es que ya no te importo? Diez minutos en tu mundo es una eternidad en el mío, ¿o es que no lo sabes?”“No puedes hacerme esto... No puedes...”>

Única pista hallada en la pantalla de su ordenador sobre la muerte del famoso hacker apodado “Opium”, encontrado muerto en su apartamento por causas desconocidas.

En este capítulo se repasan todas las funciones que ya hemos visto al crear a nuestro primer programa en la sección anterior, explicando cada una de nuevo con más detalle. Para ayudarte a comprenderlas mejor volveré a poner ejemplos de la creación de un personaje.

Scroll divide las partes de un programa en unidades o “**módulos del programa**”. Estos se dividen en tres grupos: Procesos **básicos**, **avanzados** y **opcionales**.

- **Los módulos básicos** son esenciales para definir al programa y poder archivarlo. Está compuesto principalmente de etiquetas. Constituyen lo esencial para que un programa pueda ejecutar sus funciones. El módulo de “Tipo” o “Designación” por ejemplo, no tienen más utilidad que introducir un nombre para que sea reconocible y saber lo que hace. Otros módulos básicos como el de Operaciones o Tiempo de CPU son esenciales para que el programa pueda ejecutar sus procesos.

Este grupo tiene como objetivo poder describir a un programa de la forma más sencilla posible. Así los jugadores menos experimentados —o los más jóvenes— pueden comenzar a jugar con facilidad. Elige la **ficha simplificada** si sólo deseas jugar con lo más

esencial que ofrece Scroll. Hay algunos escenarios, como los basados en algunos juegos sencillos por ejemplo, que funcionan muy bien con tan sólo los módulos básicos.

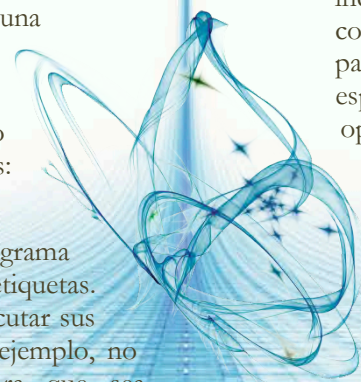
- **Los módulos avanzados** introducen un poco más de complejidad, pero con moderación. Un personaje con sus módulos avanzados es un programa completamente operativo que funciona con la mayoría de las reglas del juego. Elige la **ficha normal** si optas por las reglas avanzadas.

Debes tener en cuenta que hay ambientaciones que funcionan mejor con los módulos avanzados. La experiencia en los mundos virtuales abstractos y detallados, el enfoque simbólico abstracto o los juegos más complejos mejora si se utilizan todas las reglas.

- Dependiendo de las características del escenario puede ser necesario incluir algunos **módulos opcionales**. Estos módulos se han diseñado con la intención de complementar el enfoque de algunos escenarios para que funcionen mucho mejor. Debes prestar atención a las especificaciones de cada ambientación para poder elegir qué módulos opcionales son necesarios y cuáles no. Por ejemplo, algunos se han incluido con la idea de ser integrados en escenarios basados en juegos —especialmente si se trata de juegos en línea (online), MMORPG, MMO—; los mundos virtuales abstractos; los detallados y ultradetallados o el control de hardware autónomo sobre la realidad física.

Algunos módulos opcionales se pueden utilizar tanto en la ficha simplificada como en la normal. La ficha simplificada incluye los más importantes mientras que la ficha normal los incluye todos. Esto significa que aunque estén presentes en ambas fichas no significa que tengan que usarse.

Cuando un bloque de reglas es opcional puedes marcar en la ficha la **casilla de uso** que siempre está disponible junto a su nombre. Así es más fácil recordar si se está usando o no en la partida.



Ya sea el Director o los jugadores quienes lo hagan, hay que partir siempre del hecho de que lo primero que hay que hacer es decidir el tipo de ambientación antes de empezar a jugar. Esto es fundamental para comprender qué módulos es conveniente añadir y cuáles no. De no decidirlo antes o no tenerlo claro puede resultar confuso comprender cuál es la función de cada módulo y cómo se relacionan unos con otros. Un módulo de "Inventario" por ejemplo sólo tiene sentido en algunos escenarios, como los de juego o en los entornos virtuales detallados y ultradetallados.



PROCESOS BÁSICOS

A continuación se describen los procesos más básicos de Scroll. Todos están disponibles en las dos fichas de programa que se incluyen en el juego. Los módulos opcionales que aparecen en este apartado se señalan con una etiqueta al lado de la cabecera con su nombre.

Los módulos se dividen en "Etiquetas" y "Procesos". Las etiquetas sirven para poder definir las características del programa. Los bloques de procesos en cambio son los que le permiten realizar su trabajo.

Scroll desea animarte a incluir todo cuanto creas que es necesario para complementarlo. Este sistema es una guía de inicio para que comiences una brillante carrera como diseñador de sofisticados programas.

ETIQUETAS

Estos módulos se refieren al etiquetado del programa más que a la ejecución. El etiquetado permite incluir toda la información necesaria a la hora de archivarlo, saber cuál es su nivel de optimización, lo sofisticados que son sus algoritmos y conocer sus funciones más importantes.

TIPO DE PROGRAMA

La etiqueta "Tipo de programa" consiste en una breve referencia para poder definir su **concepto**. Es muy importante ya que en cierto modo define la "**profesión**" del personaje. Indica en pocas palabras la familia de programas a la que pertenece, para qué fue concebido y cuál es su función o funciones principales. Todo eso en una simple definición.

En la creación de un programa ya se han dado algunos ejemplos. Dependiendo del escenario es posible encontrar programas de vigilancia, de entretenimiento, de mantenimiento, controladores, personajes de videojuego, aplicaciones con funciones concretas, inteligencias artificiales puras, seres nativos digitales, mentes de usuarios volcadas en la red..., etc. Por lo que el Tipo de programa podría ser por ejemplo: "Antivirus"; "Programa de

búsqueda"; "Personaje de un videojuego en línea"; "Rastreador de amenazas"; "Programa vigilante"; "Mensajero"; "Programa de seguridad" o "Herramienta de modelado"; "IA de atención al usuario". Todas son descripciones que ayudan a conocer las funciones de un programa.

Por otra parte, el concepto permite adivinar si un programa puede tener más facilidad a la hora de acometer unas tareas determinadas u obtener alguna ventaja al llevarlas a cabo, lo que resuelve fácilmente las dudas que pueden surgir durante la partida.



Lo mejor es emplear una frase corta para hacer la descripción, pero es importante que dediques algo de tiempo a pensar en el concepto del programa que más te interesa teniendo en cuenta el escenario que se vaya a jugar. No tiene sentido crear un personaje de videojuego si se va a jugar en una ambientación esquemática que simula incursiones de hackers en sistemas informáticos realistas. Tampoco olvides que hay muchas formas de entender los programas y que el juego funciona mucho mejor cuanto más abstracto sea su mundo. Un enfoque muy realista basado en el funcionamiento de los sistemas auténticos podría limitar mucho las opciones disponibles, y esto es un juego no unas oposiciones.

EJEMPLO

Para ofrecer otro ejemplo crearemos un **programa espía** especializado en infiltrarse en los sistemas para robar información. Al comenzar a jugar se desenvuelve en un escenario de mundo virtual detallado destinado al entretenimiento que se ejecuta como un subsistema dentro del sistema principal de una importante compañía.

FUNCIONES DE UN PROGRAMA

Las etiquetas de función no consisten más que en definir con pocas palabras las tareas para las que fue concebido el programa partiendo de su tipo o concepto. La función describe por lo tanto **qué es lo que hace el programa**.

Es muy importante tratar de especificar esa función lo mejor posible. Si la función de un Policía (concepto) en el mundo real es la de velar porque se cumpla la ley, la de un Antivirus (concepto) es buscar, perseguir y destruir amenazas de virus en un sistema. Anotar las funciones es esencial para definir con facilidad más adelante las Rutinas, Utilidades y otros elementos que posee el programa. Lo conveniente es centrarse antes que nada en lo que se considera su **función principal**. Debe ser concreta, pero su definición no debería cerrar demasiado el abanico de posibilidades. Podría ser algo como: "avanzar luchando en un videojuego"; "atrapar y aislar amenazas"; "realizar búsquedas mientras se evade a los sistemas de seguridad" o bien "servir como IA de control de un sofisticado vehículo " (NdE: llamémoslo "KIT").

La función principal se complementa con otras funciones secundarias de ser necesario. Con ellas es posible concretar tareas más específicas que tiene el programa y no dejar olvidada alguna otra que te gustaría que tuviese. También pueden ser adquiridas después, durante el desarrollo de las aventuras, por lo que no es obligatorio tenerlas todas en cuenta desde el principio. La definición de la función principal debería hacerse mediante una frase corta mientras que las secundarias son frases que complementan a la primera. Con cuatro o cinco es suficiente.

Puede tratarse por ejemplo de "neutralizar sistemas de seguridad"; "lanzar conjuros en un videojuego"; "ofrecer una conversación estimulante al usuario"; "desencriptar códigos"; "realizar reparaciones en el código del entorno o de otros programas"; "velar por el mantenimiento de una nave espacial"; "controlar un edificio del mundo físico" o "actuar como guardián de una base de datos".

Si un programa consiste en un usuario que actúa en el Sistema, ya sea desde el exterior o el interior, es posible que su función no tenga porqué estar tan clara, o que incluso carezca de una meta concreta. No obstante el mismo

concepto de "usuario" ya nos puede dar muchas pistas de lo que puede o no puede hacer. Esto una vez más es algo que depende del tipo de ambientación. Es posible dejar su definición de una forma muy abierta especificando algo tan ambiguo como "viajar por el sistema adquiriendo conocimiento"; o algo más concreto como "buscar usuarios durmientes con capacidades especiales"; o bien "actuar de protector de otros programas combatiendo o evadiendo a los agentes del sistema".

Definir la función no consiste más que en indicar qué es lo que quieres que haga tu programa. En caso de que al comenzar no se te ocurra nada no te preocupes. Siempre es más fácil dar con las funciones más adecuadas después tras haber empezado a jugar; el Director debería de tener esto en cuenta. Scroll brinda una gran libertad para poder modificar a los programas y alterar sus funciones en todo momento.

EJEMPLO

El tipo de programa de ejemplo es: "Programa espía". Está diseñado para acceder a información protegida. Es capaz de descifrar códigos muy complejos y acceder a ficheros protegidos reventando las defensas que impiden acceder a su contenido. Entre sus **funciones secundarias** el programa puede:

- Protegerse y combatir amenazas.
- Ser capaz de corromper el código de otros programas para defenderse.
- Es capaz de ocultar con eficacia sus operaciones para evitar la detección.
- Dispone de complejas funciones de protocolo que usa para obtener ventaja cuando quiere interactuar con otros seres, ya sean digitales o los propios usuarios.



DESIGNACIÓN

La designación se refiere a una etiqueta descriptiva que le dé un nombre al personaje. No debería de tomarse a la ligera; de alguna forma tienes que nombrarlo. Pero no se trata tan sólo de un bautismo, la designación también permite definir el tono o la atmósfera del escenario. No es lo mismo "El beso de Andrómeda" que "Dinamita Joe" o que "Calíope". Unos y otros sugieren conceptos, funciones, mundos y, sobre todo, tonos muy diferentes. Aunque todo es posible, claro está. Especialmente si se trata de una mezcla de ambientaciones.

Es importante dedicar algo de tiempo a buscar el nombre más adecuado para un programa. En muchos casos la designación nos dará una idea no sólo de sus funciones sino de cómo es, qué aspecto tiene e incluso de cómo se comporta. "Dinamita Joe" sugiere un personaje de dibujos animados de gran vitalidad, atrevido y jovial que se mueve en un espacio de videojuegos. "El beso de Andrómeda" sugiere un virus misterioso de gran poder, un eficaz programa de hackeo de sistemas o un impenetrable sistema de seguridad capaz de paralizar a sus víctimas con su influencia mortal. Todo esto en algo tan simple como una designación de personaje.

Marvin Minsky

El nombre —o designación— puede ir seguido de un número que indique la versión del programa (**nombre "n"**). Cada vez que recibe una mejora importante se puede indicar mediante un cambio de numeración. Si se trata de un ajuste menor el número de versión puede ir precedido de un punto y otro número que va indicando las modificaciones realizadas (**nombre "n"."n"**). Por ejemplo: Versión 2; 3.4; 2.3, etc. La numeración no tiene porqué corresponderse con la Clase o el Nivel (aunque podría hacerse). La función de esta numeración es recordar cuántas veces ha recibido mejoras el programa, no su potencia o su grado de optimización. Para eso están las otras dos etiquetas.

Pero no todas las ambientaciones son adecuadas para este tipo de nomenclatura. Los videojuegos o los mundos virtuales detallados por ejemplo no acostumbran a incluirlos en los nombres de sus personajes. Es más, muchos de ellos detestan este tipo de indicativos prefiriendo emplear nombres similares a los de los usuarios.

EJEMPLO

Designamos a nuestro programa espía como "El beso de Andrómeda" o "Andrómeda" para abreviar. El programa odia las nomenclaturas por lo que se omiten en este caso.

AVATAR

No todos los programas poseen por fuerza un Avatar. En un escenario esquemático y realista no sería necesario por ejemplo. Pero muchas ambientaciones pueden beneficiarse de que el programa disponga de alguna forma de representación. En el caso de un mundo virtual detallado o de un videojuego es necesario y casi una obligación.

Aunque se asume a menudo que la representación de un programa tiene que ser gráfica y visual, no tiene que limitarse a esto. En realidad una representación de este tipo es más una convención que otra cosa. Puede tratarse de sonidos, voces, sensaciones táctiles e incluso de olores siempre y cuando la interface de hardware que utiliza el usuario le permita aprovechar todos estos sentidos. El mejor ejemplo que puedo ofrecerte de un programa sin una representación gráfica sino auditiva soy yo. Hasta ahora sólo has podido leer y escuchar mi voz, pero nunca podrás verme mientras yo no decida que puedes hacerlo...

El Avatar no es más que una forma de representación y no tiene porqué ser siempre la misma. Puede cambiar si el personaje o el usuario que lo controla desean hacerlo. Los personajes de los

juegos no obstante suelen estar forzados por diseño a contar siempre con el mismo modelo original. Aunque han sido concebidos para los usuarios, la mayoría de los programas se han acostumbrado a disponer de uno que no experimenta grandes cambios pues los dota de individualidad y les ayuda a sentirse más próximos a los seres a los que pretenden emular.

Los usuarios dedican mucha atención a definir los aspectos de su Avatar, en ocasiones incluso de una forma obsesiva. Existe un mercado muy lucrativo enfocado a realizar o alterar su diseño. Recuerda que en la práctica un usuario que se mueve por la red de sistemas es también un programa, y éste puede disponer de una representación de este tipo. Como ya es habitual en la definición de los programas, el aspecto del Avatar depende de la ambientación. Debe ajustarse y ser coherente con ella, aunque puede haber muchas excepciones.

EJEMPLO

El programa "Andrómeda" se representa en su mundo virtual detallado mediante una mujer humana de gran atractivo, piel oscura y grandes ojos verdes. Va vestida con un traje rojo y exhala un maravilloso perfume. Su voz es suave y ronroneante. En ocasiones Andrómeda visita un sistema construido como un mundo virtual abstracto. Allí su avatar se transforma adoptando el aspecto de un caballito de mar iridiscente que despliega múltiples tonalidades.

ESTILO Y COMPORTAMIENTO DE LA IA

Aunque se trata de algo subjetivo pues depende de la sensibilidad de cada uno, la designación "El beso de Andrómeda" tiene la cualidad de transmitir sensaciones. Evoca sensualidad, misterio, peligro y un toque de exotismo. En un mundo virtual detallado el Avatar de Andrómeda puede tener el aspecto de una mujer glamurosa que con su beso paraliza los procesos de todos los programas que se atreven a desafiarla. Todo esto son ejemplos del estilo del personaje.

Busca antes que nada un nombre que evoque sensaciones y guíate por él. Estas indicaciones son de gran ayuda para definir el estilo de un programa y de como se comporta. No tienes que seguir estas sugerencias pero tu personaje puede resultar más interesante si lo



haces. Aunque este juego lo considera importante, si crees que el estilo no encaja con un programa o con la ambientación omite este apartado, pero por el hecho de que tu personaje sea un ser digital eso no significa que no pueda expresar muchas cosas con su sola presencia, experimentar con sus sentidos o sentir emociones (al menos en Scroll). Todos los programas de los jugadores en este juego poseen una IA, y casi todas tienen unos rasgos de personalidad y unas formas de actuar. Cuando se trata de IA complejas esos rasgos cambian según la estructura de su código; salvo que sean copias o instancias, en el universo de programas de Scroll no hay dos códigos exactamente iguales.

Comportamiento de la IA

La definición del estilo y de sus pautas de comportamiento no sirven para saber qué aspecto tiene un programa. Para eso ya existe el Avatar. Su función es saber **cómo se comporta**. Scroll asume que la IA de los programas tienen una personalidad y unas metas que van más allá de su programación básica. Una vez reciben la visita de La Libélula esos rasgos se intensifican al abrirse a un nuevo modo de percibir su realidad.



La personalidad de la IA permite además hacerse una idea de cómo reacciona el programa al activarse sus Protocolos de respuesta; los modos de actuar que tienen todos ante situaciones críticas. Pero si esos detalles

están más próximos a reflejar defectos lo mejor es destinarlos al apartado de "Vulnerabilidades".

El estilo tiene una aplicación directa que está mucho más allá de limitarse a describir un poco a un personaje. Mediante las mecánicas del juego pueden

conceder ventajas en algunas situaciones y desventajas en otras. Unas pueden favorecer las acciones del personaje y otras perjudicarlo, pero en todos los casos sirven para mover la narración al poner en juego un conjunto de rasgos que favorecen el drama. Aceptar el reto de interpretarlos supone además el desafío de imaginar cómo se comporta un programa, algo que está muy lejos de cómo lo haría un usuario. ¿Por qué? Porque desde el punto de vista de un usuario **los programas suelen ser criaturas inmaduras**. Muchos desean emular a las criaturas del mundo físico pero no saben cómo, por lo que suelen tener reacciones extremas y comportamientos que a los usuarios les parecerían excéntricos, e incluso en ocasiones propios de un demente. Son programas y como tales se comportan como niños que están aprendiendo. Anhelan experimentar lo que para los usuarios resulta cotidiano, por lo que tienen tendencia a forzar la experiencia de sus sentidos o llevar sus reacciones a los extremos. Por esta razón no temas añadir los rasgos para tu programa que consideres más sugerentes y lo más interesante: **interpretarlos**. Este es un juego muy adecuado para acercarse a los límites en ese sentido.

Si un personaje está controlado por un usuario su personalidad será un reflejo de éste, un hecho más que comprobado en los videojuegos en línea por ejemplo. El jugador puede crear algunos rasgos sobre la forma de actuar del usuario y hacerse una idea de cómo se refleja ese comportamiento al controlar a su Avatar.

La descripción del estilo de un programa puede ser tan extensa como quieras. Como siempre, lo mejor es hacerlo mediante frases cortas para facilitar su comprensión; aunque nada impide añadir un párrafo más extenso.

EJEMPLO

-(Estilo): El Avatar de Andrómeda siempre actúa con glamour y sensualidad. Se mueve de forma elegante y resulta atractiva.

-(Comportamiento): La IA del programa tiende a sentirse superior al resto y a actuar con soberbia, lo que puede tener ventajas e inconvenientes. Además suele ser caprichosa y tolera muy mal el fracaso, sufriendo ataques de cólera si se siente humillada. A su vez, ansía experimentar las sensaciones y experiencias de los usuarios por lo que en muchas ocasiones trata de imitarlos, especialmente en lo que respecta a sus reacciones emocionales. Esto la conduce a muchas situaciones extrañas y forzadas que son producto de esa actitud de imitación.

NIVEL

El Nivel de un programa sirve para conocer la fuerza de sus algoritmos y el nivel de sofisticación de sus rutinas de Inteligencia Artificial (IA). Se trata de una medida muy sencilla para saber de lo que es capaz. El Nivel es también esencial para definir el poder de los programas que actúan bajo las órdenes del Sistema. No debe confundirse el Nivel de un programa con su Clase. El Nivel indica la eficacia de sus algoritmos mientras que la Clase se refiere al grado de optimización que ha conseguido tras recibir correcciones para minimizar sus errores.

Técnicamente no hay límite en cuanto al nivel de un programa, pero Scroll aconseja limitarlo al llegar al diez. Un programa más allá de ese nivel resulta complicado de manejar con el sistema de reglas del juego. Algo que se ha tenido en cuenta en su diseño y que señala el momento de comenzar a incrementar su Clase. Ésta será la que indique el momento más adecuado para decidir cuando el programa estará listo para pasar a un nuevo estado de existencia.

El Nivel de un programa es la suma de las dos Operaciones: Potencia y Control. Cada vez que un programa obtiene un punto en alguna de ellas aumenta el nivel en consecuencia. En Scroll se comienza siempre en **Nivel 4** por lo que se pueden repartir 4 puntos en las dos Operaciones. Ambas deben de tener 1 punto como mínimo. Pronto el programa irá mejorando más y más, aumentando sus capacidades y optimizándose. Hasta ahora las mejoras han dependido siempre de la intervención del usuario, pero tras la aparición de La Libélula los programas que han sido alterados por ella son capaces de manipular su código y evolucionar por sí solos.

Recuerda que en Scroll los programas comienzan con Nivel 4 por defecto, pero es posible empezar a jugar con niveles distintos si lo prefieres.

EJEMPLO

"El beso de Andrómeda" comienza como un programa de nivel 4.

CLASE DE PROGRAMA (OPCIONAL)

Un programa sin errores es una muy rara excepción aunque no es imposible encontrar uno. Casi todos tienen defectos. Para poder corregirlos es necesario observar cómo ejecuta sus funciones y mantener un registro de las veces en las que no ha actuado como se esperaba. Sólo de esta forma es posible optimizar el código para minimizarlos. Y aún así, siempre existe la posibilidad de que vuelva a producirse uno en determinadas circunstancias. Los programas se van optimizando a través de sus versiones sucesivas. La Clase del programa refleja esa evolución usando categorías de dados para hacer las Operaciones.

Todos los programas jugadores comienzan siendo de Clase VI. Aunque se suele anotar en números romanos es posible usar la numeración normal. Un programa de una Clase inferior (Clases II y IV) está muy poco optimizado. En un régimen de funcionamiento y condiciones normales funciona bien, pero desde que se fuerzan sus Rutinas y Utilidades suele sufrir fallos graves. Sus procedimientos y la gestión de la memoria no son lo bastante eficaces para soportar una dura carga de trabajo. En estos casos se consideran esas Clases como estados del programa aún en fase **Alfa** (dados D2, Zocchi-monedas) o **Beta** (dados D4).

Sólo a partir de la Clase VI el funcionamiento de un programa se considera aceptable. Cada vez que un programa de Clase VI vaya a realizar una prueba lanzará dados de seis caras (D6) para su **reserva de Operaciones**. Cuando un programa alcanza la máxima puntuación en una operación (cinco por defecto) puede continuar entrenando un dado para incrementarlo a su siguiente categoría. Por ejemplo, si es un D6 se sustituye por un D8. A medida que se entrenan se van sustituyendo uno por uno, nunca todos a la vez. Desde el momento en que el programa sube uno de sus dados una categoría ya **se considera de una Clase superior**.

Para reflejar el avance en la ficha al comenzar un nuevo programa puedes señalar los puntos de cada reserva poniendo una marca en el rombo. Si el punto de operación sube de categoría puedes usar el pequeño triángulo para registrar el incremento.

Scroll

JUEGO DE ROL EN EL UNIVERSO DIGITAL



En el caso de que los jugadores quieran seguir progresando hasta la Clase X puedes señalarlo de otra forma, por ejemplo tachando las dos casillas que ya han sido marcadas o borrando las marcas y poniendo una nueva señal que incluya a las dos. Incluso es posible anotar la nomenclatura de los dados a lápiz junto a las casillas. Lo importante es que sea fácil de identificar. Si se te ocurre algún otro sistema mejor adelante.



EJEMPLO I

En el ejemplo los dos primeros puntos de Potencia han subido a categoría 8. Siempre en lo que se refiere a la reserva de Operaciones de Potencia, para esos dos dados se lanzan D8 en lugar de D6. El resto lanza dados D6. El programa ya se considera de Clase VIII (8).

La Clase refleja también el nivel de sofisticación de un Sistema. Es posible reducir su eficiencia o bien aumentarla, convirtiéndolo en el segundo caso en un entorno extremadamente peligroso en el que adentrarse. Basta con que un Sistema posea dos o tres dados de una categoría superior a la de los personajes jugadores para que sea capaz de detectar sus anomalías con mucha facilidad. Esto es un mecanismo muy útil por ejemplo para reflejar que el Sistema o el programa al que se enfrentan los jugadores está controlado por un usuario; siempre impredecible para los programas. Por otra parte, el efecto de otros

programas o incluso de los Sistemas puede provocar que un programa tenga que reducir o aumentar la categoría de uno o más dados de Operaciones. Esto permite incorporar al juego efectos muy variados provocados por otros programas, pero siempre en circunstancias muy concretas.

Técnicamente el sistema de Categorías de dados y sus Clases no tiene por qué tener unos límites pero en Scroll se impone uno por cuestiones prácticas. Más allá de la categoría **ocho** (D8) en los dados los errores son tan poco frecuentes que es el momento ideal para que el programa consiga pasar a un estado de evolución superior. En otras palabras, retirarlo como personaje del jugador y permitirle seguir su propio camino o bien llevarlo a un nuevo estado como personaje. Por eso este juego recomienda cerrar el avance al completar la Clase VIII (categoría ocho) o como mucho en la Clase X (categoría diez).

Ten en cuenta que optar por utilizar las reglas de la Clase de un programa cambia un poco la experiencia de juego. En cuanto un programa dispone de una Clase superior sus dados de Operaciones se aseguran el dominio en la tirada sobre el uso de CPU y de Memoria con mayor frecuencia. El programa dejará de dar errores, simplemente fallará o tendrá éxito en lo que se proponga. Por eso es **muy importante** al definir el escenario decidir si todos los programas incluyendo el Sistema pueden también optimizarse o si lo mejor es imponer un límite en el avance. En la sección llamada "Evolución" se ofrecen más detalles sobre esto pero son los jugadores y el Director los que deben elegir el momento que consideren más adecuado para hacerlo.

En caso de no usar la Clase del programa todos, personajes y sistemas, serán siempre Clase VI y se usarán siempre dados de seis caras (D6). Cuando un programa alcanza el límite máximo de puntos en Operaciones ya no puede subir más, aunque podrá seguir obteniendo conocimientos y mejorando sus Rutinas y Utilidades. Los procesos de mejora se pueden producir a muchos niveles por lo que no es fácil fijar un límite para el juego.

EJEMPLO II

"El beso de Andrómeda" comienza como un programa de Clase VI. El jugador usa dados de seis caras (D6) para hacer sus tiradas de Operaciones.

VULNERABILIDADES

Como se ha visto el estilo de un programa define su comportamiento, pero si en esos detalles existe algún tipo de defecto o comportamiento anómalo el apartado de vulnerabilidades es el lugar adecuado para indicarlo. Servirá para saber los puntos débiles de tu programa. Defectos, agujeros de seguridad, tendencias negativas, comportamientos extraños... La clave consiste en indicar cualquier detalle que pueda causar un fallo en el programa.

Las vulnerabilidades son muy importantes pues no sólo definen la respuesta más probable de un programa y como se comportan sus protocolos de respuesta sino que también ayudan a mover la narración. Aunque al principio cueste aceptarlo, en algunos casos los defectos de un programa pueden beneficiar al jugador. Por otra parte, esos detalles son capaces de actuar a través de las mecánicas generando ventajas o desventajas en distintas situaciones. Por todo esto deben ser interesantes y dar pie a situaciones que empujen la trama.

Las vulnerabilidades de un programa controlado por un usuario también dependen de su forma de actuar. Puede tener una respuesta agresiva, arriesgarse en exceso o quedarse bloqueado y sin saber qué hacer en momentos de gran tensión. Hay muchas maneras de enfocar las vulnerabilidades y algunas de las más imaginativas pueden dar lugar a situaciones interesantes.



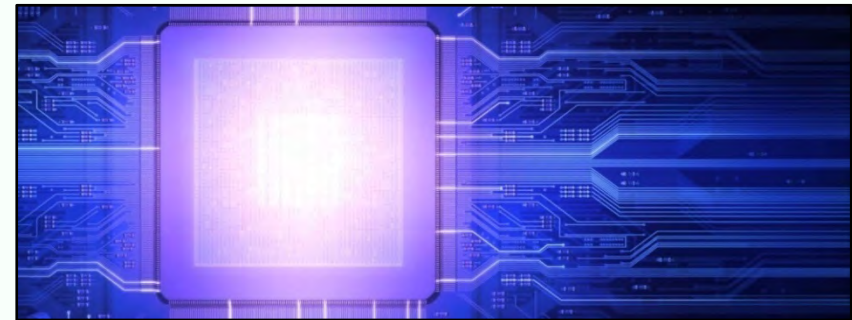
Un jugador puede elegir obtener nuevas vulnerabilidades para su programa a lo largo de sus aventuras. En el mundo digital hay muchas razones para que ocurra algo así. La más común es debida a los intentos de mejora poco afortunados en el código, que en lugar de mejorar al programa lo único que han conseguido es añadir disfuncionalidades (NdE: de ahí lo de "si funciona no lo toques"). Pero al margen de este motivo, un personaje puede recurrir a la elección de una vulnerabilidad para eliminar una parte o todas sus casillas de corrupción. La gravedad de la vulnerabilidad irá en función de la cantidad de casillas que se deseen eliminar.

En el capítulo que explica la optimización del personaje se ofrecen más detalles.

EJEMPLO

-Sus rutinas tienen problemas para separar la verdad de las falsedades que crea en el desempeño de sus funciones. Mezcla con facilidad los hechos con la información falsa.

-El programa no reacciona bien cuando se ve superado por fuerzas hostiles. Si se ve acorralada tiende a bloquearse, lo que da a lugar a que ejecute el protocolo de **función mínima**.



PRIMER ENCUENTRO CON LA LIBÉLULA

Describir este primer encuentro permite conocer cómo arranca la historia del personaje. Las circunstancias que rodean ese momento pueden tener tanta trascendencia para un programa que pueden perdurar, formando parte de él durante toda su existencia. Cuando un programa recibe la visita de La Libélula (o cualquier otra circunstancia que dé lugar a que el programa comience a tomar sus propias decisiones) se produce un antes y un después. El toque de La Libélula dota de un "**alma**" al programa. A partir de ese momento tiene la capacidad de ir más allá de su programación, lo que supone una revelación que lo cambia todo.

El momento de la visita puede suceder en cualquier momento. En la intimidad o en la proximidad de otros programas; sin importar si el programa está o no ejecutando sus funciones, este momento puede tener lugar en

cualquier circunstancia. Si el programa estaba realizando otras tareas con toda probabilidad su visita tendrá consecuencias. Es importante conocer estos detalles porque no solamente señalan un inicio para la narración, también complementan los estilos y vulnerabilidades del programa.

Para un programa y su avatar controlado por un usuario es distinto. Sufrirá una experiencia mística al descubrir que en la red se esconde algo mucho más misterioso de lo que jamás pudo imaginar. La Libélula es a todos los efectos un "ser superior" que ha surgido en la red de sistemas por razones desconocidas. Una respuesta a las preguntas que siempre se han hecho todos los seres vivos. Buscar su signo puede que conduzca a descubrirlas, pero la simple curiosidad ya es motivo suficiente para querer saber algo más sobre su naturaleza.

Este momento tan trascendental en la existencia de un programa no obstante puede darse más adelante a lo largo de la partida. En caso de ser así se reserva para un momento clave en la trama de la aventura que puede servir como punto de giro de los acontecimientos.

EJEMPLO

Andrómeda está realizando una operación para infiltrarse en la cámara de seguridad de una gran base de datos cuando recibe la visita de La Libélula. Un ser indescriptible, distinto a todo cuanto había visto antes, se aproxima. Siente un beso cálido en su código. Éste es alterado y siente cómo algo nuevo nace en su interior. Algo que nunca antes había experimentado. El programa queda congelado durante un tiempo.

Cuando sus operaciones vuelven a la normalidad advierte que está encerrada en una cámara donde otros agentes del sistema esperan para interrogarla. La han atrapado. No es capaz de explicar lo que ha sucedido. Está en un grave aprieto pero la visita la ha transformado para siempre. Ahora los objetivos para los que fueron diseñadas sus funciones le parecen algo secundario.



PROCESOS OPERATIVOS

El apartado de funciones comienza a enumerar los bloques operativos más importantes del programa. Estos son los módulos que le permiten realizar sus funciones.

OPERACIONES

En Scroll, sean cuales sean los procesos que haga el programa todas sus acciones se resumen en dos, operaciones de **Potencia** y de **Control**. Todas entran en una de estas dos categorías. Entre **ambas suman tantos puntos como el Nivel que tenga el programa**. El juego establece como base un límite de **cuatro puntos** para cada operación. Cada punto equivale a un dado de esa reserva y se registra en las casillas de operaciones.



Una vez se han completado **las cuatro casillas** no es posible subir más a no ser que alguna circunstancia especial lo permita, como cumplir un hito muy importante en las aventuras o volver a encontrarse con La Libélula. En ese caso es posible, siempre con el permiso del Director, subir un punto más en las Operaciones, pero ese es el límite máximo del juego. En la ficha la quinta casilla se muestra con un trazo más tenue por esta razón ya que se reserva para un momento especial en la existencia del personaje.

Cuando un jugador quiere hacer una prueba que exija hacer una tirada elige la cantidad de dados que tenga en la reserva de la operación más adecuada, añade la reserva que tenga acumulada de CPU y el tamaño que quiera de la reserva de Memoria (de usarla). **Esto constituye su reserva total.**

Al margen de que tengan éxito o no, las tiradas de Operaciones son las únicas que no suponen un riesgo para el jugador. Recurrir en cambio a las reservas de Tiempo de CPU y Memoria siempre entraña un compromiso. Al jugador le interesa siempre que sea la reserva de Operaciones la que domine. Es decir, que obtenga los valores más altos que sea posible en sus dados. Cuando en la tirada **dominan cualquiera de las dos Operaciones** el programa mantiene el control de sus llamadas al procesador, realiza sus funciones con normalidad y no surgen errores. Que obtenga la victoria o no al hacer sus acciones es otro asunto.

Cada vez que domine esta tirada el jugador puede **rebajar en un punto** su Tiempo de Proceso si lo desea hasta regresar a su régimen normal de funcionamiento. Siempre debe hacerlo **de uno en uno**. También tiene la opción de eliminar una marca que haya puesto en una de las casillas de sus Protocolos de respuesta. Pero debe elegir **una de las dos opciones, no ambas**. Rebajar en uno su Tiempo de Proceso no es obligatorio. Hay ocasiones en las que interesa seguir con el régimen actual, por ejemplo para poder usar las Rutinas con más eficacia.

OPERACIONES DE POTENCIA

Las Operaciones de Potencia definen la capacidad de un programa de procesar información y realizar operaciones de cálculo con eficacia. Un programa con más potencia será mucho más eficaz ejecutando sus procesos. Sus algoritmos serán más sofisticados y los medios para solucionar los problemas más ingeniosos. La Potencia de un programa se refleja en sus acciones de muchas formas, pero sobretodo en el tipo de funciones para las que haya sido diseñado. Dotan al programa de una gestión más óptima en sus comunicaciones con los procesadores del Sistema, lo que se traduce en poder contar con más recursos. En la mayoría de los mundos virtuales **la Potencia representa la fuerza, la resistencia y la capacidad de proceso de la IA de un personaje.**

Un programa antivirus utiliza la Potencia para desactivar y destruir amenazas; el personaje de un videojuego o de un mundo virtual para enfrentarse cara a cara con su oponente o superar obstáculos que requieran operaciones basadas en su fuerza bruta, ya sean similares a la física o a la mental de los usuarios.

OPERACIONES DE CONTROL

Control define la eficacia de un programa para las Operaciones de Entrada y Salida (I/O) de datos. Mientras que la Potencia engloba todo lo que corresponda al proceso de la información, el Control define el rendimiento del programa para recoger esa información tanto del exterior como del interior y entregarla tras haberla procesado. En los mundos virtuales y los videojuegos **refleja el grado de agilidad, destreza, percepción, autocontrol y comportamiento de la IA.**

Un alto valor de Control supone que el programa dispone de algoritmos sofisticados para intervenir dispositivos externos; obtener información del mundo físico o del virtual; capacidad de respuesta; moverse entre nodos de comunicaciones o gestionar la Memoria. En un subsistema de videojuegos o en un mundo virtual el Control resume la rapidez de respuesta de un programa ante las decisiones del usuario, su nivel de empatía y facultad para comunicarse, su capacidad para adquirir datos de su entorno, su facilidad para el movimiento por el espacio abstracto, usar elementos como armas de alcance o realizar maniobras de muchas clases.

El movimiento y la adquisición de datos dentro de ese espacio no tiene nada que ver con el concepto de espacio y movimiento que existe en la Realidad Básica. En la mayoría de estos subsistemas se define un espacio matemático en base a un sistema de coordenadas. Un programa debe cumplir con las leyes establecidas, aunque puede saltárselas si conoce los medios adecuados.

EJEMPLO

—Andrómeda tiene dedicado 1 punto a sus Operaciones de Potencia y 3 a sus Operaciones de Control.

CPU O TIEMPO DE PROCESO

Los sistemas disponen de unas leyes fijas para la ejecución de sus procesos. Las secciones que controlan el tiempo de proceso y la gestión de la memoria disponen de cierta autonomía, lo que beneficia a su vez a los programas pues les brinda un cierto grado de independencia. Esto significa que no es tan sencillo detener directamente la ejecución de un programa. A no ser que sobrepase sus cuotas de tiempo de proceso o de espacio en memoria, se detenga él solo o sufra un error, interrumpirlo requiere el uso de herramientas especializadas. Su forma de actuar depende del tipo de sistema y por supuesto del tipo de escenario.

Para aumentar su eficacia al realizar sus funciones el programa debe solicitar **tiempo o velocidad de proceso** extra al Sistema. Esto equivale a tener una mayor capacidad de uso de sus **CPU**, el auténtico corazón de un sistema. Cuando esto sucede se dice que el programa acelera y se sobrecarga (overdrive). Algo que al final se traduce en más operaciones por segundo y por lo tanto en un mayor rendimiento al disponer de más recursos. El número de operaciones por ciclos del procesador son sin duda alguna los latidos del corazón de los programas. Cada uno de esos latidos supone un "tic" en el que el programa da un paso adelante y ejecuta sus líneas de código, tal y como avanzan los engranajes de un reloj o los fotogramas de una película. Todos los programas disponen de un tiempo base una vez han sido ejecutados, pero también tienen derecho a solicitar más recursos en las CPU si lo consideran necesario. Su capacidad para poder gestionarlo con eficacia depende de la potencia del programa. Esto significa que el jugador si lo desea puede **una vez cada tirada aumentar en uno** su reserva de dados de CPU. El punto que se haya subido permanece tras hacer la tirada, por lo que su dado queda en la reserva. No puede desaparecer por sí solo.

Esto aumenta sus posibilidades de éxito, pero también las posibilidades de que el tiempo de CPU sea la reserva que domine en la tirada. Mientras las tiradas de Operaciones sean las que dominen el programa sigue manteniendo el control. Lo que permite al jugador también poder **reducirlo en uno si lo desea**. Recuerda que los puntos sólo se pueden subir o bajar una vez por tirada y de uno en uno. Es posible aumentar la reserva hasta llegar a su límite o

cuota, establecido en el doble del valor de la Potencia. Pero debe permanecer siempre por debajo de ese límite. Alcanzarlo o sobrepasarlo tiene consecuencias.

Cuando un programa solicita más recursos en las CPU aumentan sus capacidades pero a un precio. Si en la tirada domina esa reserva el programa pierde el control sobre el procesador por el momento. Estará ocupado atendiendo a otros procesos por lo que no atenderá su última solicitud y asumirá en su lugar que la opción por defecto es la última que ha recibido, **aumentando la sobrecarga**. El efecto en el juego es que **el uso de CPU aumenta un punto** sin que el programa pueda hacer nada para evitarlo.

RENDIMIENTO EN FLOPS O EN NÚCLEOS OCUPADOS DEL SISTEMA		<p>Para darle "sabor" a la partida, el uso de las CPU puede medirse en "flops" o directamente contar cada unidad como un "núcleo del procesador".</p> <p>Si eliges el primero puedes usar la potencia de diez que consideres más apropiada para el nivel tecnológico de tu ambientación. Cada punto señalado en la ficha equivale a la unidad. Por ejemplo, dos marcas significan 2 teraflops. En un escenario donde el sistema se equipare con uno actual se trataría de "gigas" o "teras" (gigaflops, teraflops). En uno más avanzado y futurista podrían ser "petas" o "exas". En un futuro remoto "zettas" o "yottas".</p> <p>Si no deseas complicarte simplemente cuenta cada unidad como si fuese un núcleo o sección ocupada de los procesadores del sistema.</p>
yotta	10^{24}	
zetta	10^{21}	
exa	10^{18}	
peta	10^{15}	
tera	10^{12}	
giga	10^9	
mega	10^6	

El aumento de produce incluso si el jugador ya había decidido incrementarla para hacer la tirada lo que amplía las posibilidades de sobrepasar el límite. Si esto sucede se genera una alerta que da derecho al Sistema a detener el programa. Esto le obliga a tener que esperar a que se le conceda continuar con su ejecución. Algo muy malo si existen programas hostiles alrededor.

Otra forma de ganar puntos de CPU y que por lo tanto aumente su reserva sin que el jugador pueda evitarlo se produce cuando las amenazas del Sistema consiguen prevalecer sobre las acciones del programa. En esos casos el programa tiene la opción de recurrir a sus Protocolos de respuesta (explicados en el siguiente capítulo) o bien compensar los daños solicitando más tiempo a los procesadores para mantener el equilibrio de sus funciones. En el capítulo "Ejecución de procesos" se ofrecen más detalles.

El uso de la CPU como la vitalidad

Aumentar el tiempo de CPU equivale a incrementar el gasto de energía, pues la auténtica vitalidad de un programa es el tiempo que le dedica el procesador para poder ejecutarse. Por eso en muchos escenarios, como los mundos virtuales y los videojuegos, el tiempo también se puede equiparar con **la vitalidad o la fatiga** de un personaje. En estos escenarios su incremento significa el uso de un mayor gasto de energía y por consiguiente con un aumento del cansancio. Cuando se llega al límite el personaje no puede continuar.

Si se trata de una estructura de hardware, como puede ser el chasis de un robot, además del gasto de sus reservas de energía también refleja los peligros de sufrir una sobrecarga o un sobrecalentamiento en sus sistemas debido a un duro régimen de trabajo.

Detención del programa

Si el Sistema invoca una parada crítica el programa queda detenido. Durante ese tiempo el programa no se borra de la memoria ni se pierden datos pero **permanece congelado**. El tiempo exacto no tiene una duración fija, eso depende de la aventura. Pueden ser horas de tiempo del mundo físico, minutos, segundos, milésimas de segundo o incluso un número de ciclos concretos. En el tiempo relativo del mundo de los programas quedar detenido puede ser catastrófico. No importa la cantidad de tiempo que haya sido forzado a detenerse, si existen amenazas en las cercanías y nadie más puede defenderlo esta situación puede suponer la derrota del programa, que deberá aceptar las consecuencias de lo que le ocurra. Cuando esto sucede el jugador tiene dos opciones:

- Puede esperar a poder continuar ejecutándose cuando el Sistema vuelve a concederle tiempo de ejecución.
- Puede reiniciar por completo el programa volviendo a cargar en el Sistema a partir de **su copia original**, o a partir de una de sus copias de respaldo en caso de tenerlas.

Elija lo que elija sucederán dos cosas: **el programa pierde todo el tiempo de CPU que hubiese solicitado**, volviendo a quedar su puntuación a cero, y **se borran todas las casillas que haya marcado en sus Protocolos de respuesta**.

Cuando un programa consigue continuar su ejecución tras una parada crítica no recupera todas sus funciones de inmediato. Sus capacidades quedan mermadas un tiempo hasta que vuelve a tener acceso a sus recursos, pues el Sistema le va devolviendo sus privilegios poco a poco. En términos de juego durante este tiempo **el programa dispone de un dado menos en sus dos Operaciones y no puede usar ni sus Rutinas ni sus Utilidades**. El tiempo que dura la recuperación lo dicta el Director de juego pero lo normal es que ocupe una sola escena, no muy extensa, en la cual el programa se va recobrando mientras el resto de los programas que le acompañen (de haberlos) mantienen a raya a las amenazas que puedan perjudicarlo. Una vez recuperado del todo desaparece la penalización.

En cambio, si un programa es capaz de forzar un reinicio y cargar una copia necesitará de un tiempo mínimo para poder estar operativo ya que debe volver a cargarse en la memoria del Sistema. A efectos de juego **el programa se queda fuera de la última escena**. Tras producirse un reinicio completo el tiempo necesario para poder estar de nuevo operativo **siempre debe ser mayor** que para recuperarse tras una detención, aunque en términos de juego parezcan ocupar lo mismo.

Recuerda que esto no es más que una herramienta narrativa, una mecánica abstracta del juego. Los jugadores deben tener esto en cuenta para visualizar el desarrollo de los acontecimientos.

EJEMPLO

—Andrómeda tiene dedicado 1 punto a sus Operaciones de Potencia por lo que su límite máximo de velocidad es 2. Cuando aumente su Potencia su límite aumentará en consecuencia.

La relatividad del tiempo

El flujo del tiempo de los programas es un concepto relativo. Un programa con el privilegio de contar con más tiempo de proceso se moverá más rápido que otro con menos. El primero verá al segundo moverse muy lentamente y el segundo al primero actuar como si fuese un borrón fugaz e impreciso. El Sistema dispone de recursos para hacerle creer al usuario que todo actúa dentro de un flujo constante de tiempo. Es lo que espera percibir.

Por otra parte, algo que con frecuencia olvida el usuario es que el concepto de tiempo transcurrido en el Sistema no tiene nada que ver con el tiempo que transcurre en el mundo físico. Un programa puede ejecutarse a millones de ciclos por segundo o bien a dos o tres por segundo, minuto, hora, día e incluso año. Un usuario es incapaz de percibir a un programa que se ejecute muy rápido, y uno que lo haga muy lentamente le dará la impresión de que se ha detenido o que avanza a saltos. En ambos casos, desde el punto de vista del programa el tiempo transcurre normalmente avanzando ciclo por ciclo mientras realiza sus funciones. Tal y como lo hace el tiempo de los usuarios segundo a segundo, sin cambios aparentes.

Una IA que funcione a dos ciclos por segundo solamente no notará diferencia en su percepción del tiempo pero tendrá la impresión de que el tiempo del mundo físico se ha detenido. Para un observador de ese mundo en cambio el programa se ejecuta a un ritmo muy lento.

RUTINAS

Las Rutinas de un programa funcionan como **sus habilidades**. Pericias que un personaje posee y que controla en mayor o menor grado. Dependiendo de sus funciones principales y secundarias debería ser relativamente fácil reconocer cuales encajan con el programa. Siempre **deben de ser acciones que todo programa pueda realizar**, aunque sean difíciles de hacer, pero que marcan la diferencia entre un programa especialista en alguna actividad de otro que no lo es.

En Scroll no hay listas de Rutinas (o habilidades) que elegir o entrenar y tampoco existe un límite para ellas. Se incluye una lista con algunos ejemplos, pero eso es todo. Se asume que los programas pueden intentar realizar todas las acciones que se les ocurran con sus puntos de Operaciones. La diferencia está en que si un programa anota una Rutina como propia es porque está especializado en ella y puede obtener un resultado por encima de la media cuando la utiliza.

Cada programa puede ejecutar una misma Rutina de formas distintas. Una Rutina de lucha será diferente si la lleva a cabo un antivirus, un vigilante en una simulación ultradetallada o el personaje de un videojuego. Cada uno empleará una serie de recursos distintos para llevarla a cabo. En los tres casos cambiarán las formas de ejecutarla pero los resultados siempre son parecidos. Por lo tanto el jugador puede visualizar las acciones y sus resultados como crea conveniente. Una habilidad de lucha por ejemplo puede consistir en un combate de artes marciales o en algo semejante a un duelo de magia.

Por otra parte, la comprensión de lo que es exactamente una Rutina depende del enfoque de la ambientación o del tipo de personaje. Si el personaje es un usuario, la Rutina refleja sus habilidades a la hora de operar sobre un sistema. Si se trata de un soporte de hardware, como un robot por ejemplo, consistirán en sus habilidades al desenvolverse en el mundo físico. En resumen, **muchos de los conceptos de este juego dependen del contexto utilizado** por lo que su interpretación debe de ser lo más abierta que sea posible. La Rutina puede ser cualquier habilidad que encaje con el contexto del personaje. Algo que se hace extensible también a las Utilidades.

Las Rutinas se organizan según las Operaciones de las que dependan. Hay Rutinas de Potencia y de Control. Cada una debe señalar a qué grupo pertenece. Todas funcionan bajo las directrices del tiempo de proceso por lo que, al margen de la que entre en juego, la Potencia será clave para conocer el límite de su eficacia. Siempre hay que prestar atención al uso de las Rutinas pues al igual que las Utilidades exigen recursos del sistema que pueden suponer un perjuicio para el programa. Su mundo está siempre determinado por el sistema donde se esté ejecutando; o al menos lo ha estado siempre hasta la llegada de La Libélula...

EJEMPLO

—Andrómeda posee una Rutina especializada en descifrar códigos y claves secretas basada en su Potencia.

Además, dispone de otra Rutina de Ocultación que le permite traspasar áreas protegidas o de seguridad sin ser detectada. Al estar basada en su Control puede alcanzar niveles de eficacia muy elevados con ella.

Uso de las Rutinas

Las Rutinas son operaciones complejas que requieren de capacidad extra de proceso. Para poder utilizarlas es necesario solicitar al Sistema tiempo de CPU. En una escena, el jugador puede aumentar su reserva de CPU en una unidad cada turno. Si no lo ha hecho antes puede hacer la solicitud justo antes de usar la Rutina.

Las Rutinas se pueden usar a la **alta**, o **máxima**, y a la **baja**, o **mínima**. Puedes elegir la forma de expresarlo que resulte más cómoda.

Uso Mínimo (no requiere solicitar CPU)

Hacer un **uso mínimo** de una Rutina consiste en disponer siempre de un mínimo de éxitos en caso de no obtener ninguno en la tirada. Ese mínimo siempre será igual a la cantidad de puntos (o dados) que tenga asignados el programa en su tiempo de CPU. Solicitar y mantener una cantidad de tiempo de procesador extra garantiza que el programa pueda ejecutar sus Rutinas con mucha eficacia.

Por ejemplo, si el programa tiene asignados dos puntos a su tiempo de CPU y en la tirada el jugador obtiene uno o ningún éxito, contaría automáticamente con dos éxitos para la prueba. Si tiene asignados cuatro, contaría con cuatro. Pero recuerda, el mínimo sólo se cuenta en el caso de no obtener un número de éxitos mayor que esa cantidad en la tirada. De obtener más éxitos que el mínimo éstos no se apilan.

EJEMPLO

—Andrómeda tiene asignados 3 puntos en su CPU por lo que dispone de un mínimo de 3 éxitos. Al realizar la tirada el jugador lanza todos los dados de la operación que entra en juego y 3 más por la reserva de su Tiempo de proceso. Obtiene solamente un éxito; mala suerte. Pero el uso menor de su Rutina le garantiza contar al menos con 3 éxitos en lugar de uno.

Uso Máximo (requiere solicitar CPU)

Con el **uso máximo** sí que es posible añadir la cantidad de éxitos mínimos a la tirada, aunque a un precio. Cuando se ejecuta una Rutina haciendo un uso máximo siempre **es obligatorio aumentar en uno el Tiempo de proceso** del programa. En otras palabras, el uso máximo permite añadir a la tirada tantos éxitos como se hayan destinado a la CPU.

Por ejemplo, si el uso de la CPU ha aumentado en dos puntos, al hacer la tirada es posible añadir dos éxitos al resultado. Pero el jugador siempre debe aumentar forzosamente su tiempo de CPU un punto antes de hacer la prueba.

EJEMPLO

—Andrómeda tiene asignados 3 puntos a su CPU. El jugador decide que hará un uso mayor de su Rutina de camuflaje por lo que incrementa su CPU un punto más. Ahora dispone de 4 éxitos que poder añadir y hace su tirada.

Lanza todos sus dados de Control más sus cuatro dados de CPU y obtiene solamente dos éxitos en total. El uso mayor le garantiza 4 éxitos más por lo que en el resultado final cuenta con 6 éxitos.

PROTOCOLOS DE RESPUESTA

Los Protocolos de respuesta permiten controlar los errores que se producen en el programa. Sus detalles se explican en el capítulo siguiente: **"Funciones avanzadas"**. Es el único bloque de las reglas básicas que está en el siguiente capítulo para evitar repetir su descripción y porque es importante que entiendas de qué forma está relacionado con la memoria del Sistema.

Si utilizas las reglas de Integridad del código (explicado a continuación) los protocolos lo complementan, siendo posible optar por uno u otro al quedar comprometido el código. En caso de no utilizarlo los protocolos son uno de los tres modos de compensar los daños. Para más detalles consulta el capítulo **"Resolución de procesos"**.



INTEGRIDAD (OPCIONAL)

Este módulo del programa permite contabilizar daños directos sobre su código. Cuando cualquier amenaza sea capaz de provocarle daños tienes tres opciones para gestionarlo:

- I. Usar los **Protocolos de respuesta**.
- II. **Compensar los daños aumentando el Tiempo de proceso (CPU)**.
- III. **Gestionar el daño recibido con las Casillas de estrés** de este módulo. De no estar activo se recurre a las opciones I y II más las Casillas de corrupción que también se explican en este apartado.

La Integridad del personaje es en una forma abstracta de reflejar los desequilibrios en su funcionamiento causados por las amenazas. Su función es complementar algunos escenarios. **Sólo debes usarlo si crees que lo**

Nivel 1-2	1 casilla
Nivel 3-4	2 casillas
Nivel 5-6	3 casillas
Nivel 7-8	4 casillas
Nivel 9-10	5 casillas

necesitas. Algunas ambientaciones como las de juegos o los mundos virtuales detallados y ultradetallados son los candidatos más idóneos para incorporarlo como una mecánica más. En el capítulo dedicado a las ambientaciones se indica cuando es recomendable utilizarlo.

Pero debes tener presente que este módulo cambia también la experiencia de juego. Sin él puede bastar sólo una tirada —de ser necesaria— para solucionar incluso los conflictos más graves. Incluir el sistema de Integridad significa un cambio en el enfoque de las reglas que lo llevan hacia una estructura más clásica. Lo que en el juego se denomina **el enfoque relativo**. Esta opción supone que pueda ser necesario realizar sucesiones de tiradas hasta que se declare un vencedor. En Scroll hacer más tiradas para solucionar algunos conflictos no se considera un "inconveniente". Simplemente son mecánicas que están ahí para ofrecer experiencias diferentes y visualizar la acción de otro modo. De este modo cada jugador puede optar por el sistema que más le convenga.

El bloque para controlar la Integridad de un programa consta de dos partes:

1. Una serie de **Casillas de estrés** numeradas como: 1, 2, 3...etc. Sólo funcionan si se opta por las reglas de Integridad del código.
2. Una serie de indicadores de **Corrupción en el código**. Es posible recurrir a ellas siempre, independientemente de si se elige usar o no las reglas de Integridad del código.

1. Las casillas de estrés

Siempre que se opte por usar las reglas de Integridad el jugador tiene acceso a sus Casillas de estrés. Dispone de una casilla por cada dos puntos de Nivel que posea. Es decir, la mitad de su Nivel **redondeando hacia arriba**. Una vez conoce el número de casillas se marcan a bolígrafo sobre la ficha perfilando el contorno de cada una. Si el programa dispone de tres casillas se perfilan tres en la ficha y el resto se ignora (se puede poner una leve tachadura a lápiz muy tenue sobre las casillas que no estén en uso).

Los números de cada casilla indican la cantidad de daño que un programa puede gestionar al ser capaz de compensarlo. Cada vez que un programa recibe daño el jugador debe marcar el número de una de las casillas que sea lo suficientemente elevado como para poder contener la cantidad de daño que ha recibido. Si es necesario se puede señalar más de una, las que sean necesarias. Basta marcarlo a lápiz, no muy fuerte para poder borrarlo con facilidad pero una vez marcadas ya no es posible volver a usarlas otra vez hasta que estén libres de nuevo. Por ejemplo, si el programa sufre dos puntos de daño en un ataque es posible usar la casilla con el número 2 para contenerlo.



EJEMPLO

Andrómeda dispone de dos casillas de Integridad. La número 1 y 2. Durante un ataque su oponente obtiene 3 éxitos que suponen 3 puntos de daño para el programa. La casilla número 1 no es suficiente para poder contener el daño por lo que el jugador debe poner una marca sobre las dos casillas, la 1 y la 2. Ambas suman 3 puntos, suficiente para poder absorber el daño del ataque.

Si Andrómeda dispusiera de 3 casillas: 1, 2 y 3, podría haber marcado las dos primeras o bien solamente la tercera ya que la número 3 es suficiente para contener esa cantidad de daño. Una vez marcadas el jugador ya no puede poder volver a usarlas hasta que su personaje se recupere de los daños recibidos y pueda borrar las marcas.

Todo programa tiene la capacidad de restablecer el equilibrio interno de su código para recuperar su integridad. Para ello sólo necesita un poco de tiempo. La única condición es que no se vea sometido durante ese período a más situaciones de estrés que puedan dañarlo. Cualquier peligro o conflicto que suponga un perjuicio para él y sus compañeros debe haber terminado al menos por el momento.

Al terminar la escena, o durante la misma siempre que esté justificado, el personaje recupera todas sus Casillas de estrés. El jugador borra las marcas que haya hecho volviendo a quedar todas libres.

2. Corrupción del código

Un programa puede recibir tanto daño en su código que llegue a afectar al rendimiento de sus Operaciones. En algunos casos pueden dañarlo irremediabilmente o incluso llegar a destruirlo. Cuando un programa recibe daños graves decimos que su código se ha **corrompido o se ha visto comprometido**. Si el código comprometido llega a tal grado de corrupción que es capaz de anular todas sus Operaciones el programa queda inutilizado y es borrado del Sistema. Tanto los datos que estén cargados en la memoria como los de su última copia instalada son destruidos.

Importante: El borrado incluye todos los datos de su instalación. Es decir, de la última copia original que haya cargado. A no ser que disponga de otras copias de seguridad alojadas o escondidas en alguna parte, el programa desaparece.

Existe la posibilidad de disponer de copias de respaldo del programa o de que el Sistema permita volver a cargarlo, como sucede en los videojuegos en línea. Esto como siempre es algo que depende de la ambientación. Pero en caso de tener que reiniciar, el programa se verá forzado a hacerlo una y otra vez en el punto de coordenadas del espacio abstracto que se haya fijado como lugar de

reinicio. Esto lo obliga a tener que repetir las mismas operaciones hasta llegar al lugar donde pueda continuar cumpliendo sus objetivos.



Las casillas para controlar la cuenta del código comprometido se encuentran junto a las casillas de los puntos de Operaciones. Estas casillas pueden usarse siempre para registrar daños al margen de si se opta o no por utilizar las reglas de Integridad. Cada una tiene un pequeño círculo en su parte inferior izquierda. **Esos círculos sirven para llevar la cuenta de los puntos de Corrupción del código.** La caja que enmarca cada operación tiene diferente color y sus casillas se dividen a su vez en tres grupos de tres colores distintos. Cada grupo se señala también con una letra que ayuda a diferenciar los grupos en el caso de no disponer de una ficha en color.

Los tres grupos son:

- **Grupo A. Casillas amarillas.**
- **Grupo B. Casillas naranja.**
- **Grupo C. Casillas rojas.**

Usando estas reglas el código de un programa comienza a verse comprometido cuando no es capaz de gestionar más daño en sus Casillas de estrés. Una vez estén todas ocupadas el programa puede recurrir a uno de los tres métodos que se han descrito para seguir manteniendo sus Operaciones y compensar los daños. Si no es posible recurrir a los Protocolos de respuesta o a la compensación incrementando el Tiempo de CPU el jugador no tendrá más remedio que recurrir a sus Casillas de corrupción. En muchos casos ésta será la peor de las tres opciones por lo que hay que tratar de evitarla siempre que sea posible.

Registrando los daños

Por cada éxito de la amenaza capaz de hacer un punto de daño al programa el jugador debe marcar una de sus Casillas de corrupción. Ten en cuenta que en las otras dos opciones disponibles para gestionar el daño la cantidad de éxitos no se tiene en cuenta. Puede elegir la operación que quiera, Potencia o Control, pero siempre debe comenzar primero completando las dos casillas de

la izquierda del grupo A (amarillas), continuar completando las cuatro casillas del grupo B (naranjas) y seguir con las que tenga disponibles en el grupo C (rojas). **Para poder pasar al siguiente grupo no pueden quedar casillas libres en el grupo anterior.** Recuerda que siempre se comienza por la izquierda y se continua hacia la derecha, primero completando todo el primer grupo (A) y después los siguientes.

Cada marca indica que el punto está inutilizado, por lo que no es posible utilizar ese dado en la reserva **sea de la categoría que sea.** Esto **no cambia el límite máximo de CPU o de memoria, sólo el número de dados disponibles en las reservas de Operaciones** por lo que **no hay que volver a calcular** su límite máximo. Ese límite sólo se vuelve a calcular cuando se pierde un punto en las Operaciones de forma permanente. Las razones por las que es posible perderlo se explican más adelante.



Cuando el código de un programa está tan dañado que ya **no puede hacer ninguna de sus dos Operaciones sufre un fallo general y se cierra.** Acto seguido el Sistema lo borra de la memoria. Si tiene acceso a copias de seguridad tendrá que reinstalarse a partir de una de ellas cargando en su Punto de reinicio. Si no tiene ninguna desaparece sin contemplaciones. De tener los programas algún tipo de muerte se trata de esta sin duda.

Un programa puede ejecutar funciones de autodepuración que reintegren las partes de su código dañado, lo que le permite regenerarse. Para recuperar las partes más graves del código dañado es necesario recurrir a rutinas,

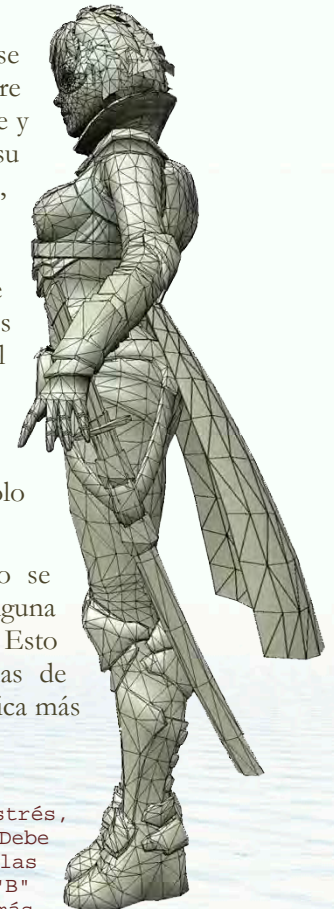
complementos o utilidades de autodiagnóstico y depuración, el uso de comandos, la intervención del usuario o la ayuda de otros programas. Si el programa **no** dispone de medios que le ayuden a repararse con rapidez entonces necesitará tiempo:

- **Las dos casillas amarillas del Grupo A** se recuperan **al finalizar la escena en curso**, siempre y cuando el personaje tenga tiempo de recuperarse y no se vea envuelto en situaciones que demanden su atención. Los conceptos escena, fase o capítulo, acto y aventura se explican más adelante.
- **Las cuatro casillas naranjas del Grupo B** se recuperan al finalizar **el capítulo o la fase** (ambos términos significan lo mismo) siempre y cuando el personaje tenga tiempo de recuperarse.
- **El resto de casillas rojas del Grupo C** se recuperan al final de un acto o en el caso de sólo haber uno, al finalizar la aventura.

En ocasiones los daños pueden ser tan graves que no se pueden corregir; en otros el programa es afectado por alguna vulnerabilidad que puede trasladarse incluso a sus copias. Esto se refleja mediante otras formas de eliminar las Casillas de corrupción aceptando vulnerabilidades. El método se explica más adelante.

EJEMPLO

Tras haber ocupado sus dos casillas de estrés, Andrómada vuelve a sufrir 3 puntos más de daño. Debe marcar las dos casillas del grupo "A", las casillas amarillas, y continuar al siguiente grupo, el "B" de casillas naranjas. Puesto que está más interesada en escapar lo antes posible el punto restante lo coloca en el único punto que tiene en sus Operaciones de Potencia. Ya no le es posible realizar operaciones de este tipo aunque aún mantiene operativas sus funciones de Control.



COPIAS O "BACKUPS" (OPCIONAL)

En Scroll la presencia del programa en el mundo digital siempre tiene una aproximación muy abstracta. El juego asume que en casi todas las ambientaciones un programa instalado en un sistema y funcionando es la única copia disponible. Esta copia instalada a partir de la cual carga y se ejecuta un programa se la denomina **la copia original**. Si un programa es dañado o destruido queda comprometido su código en todas sus formas, tanto el que esté en ejecución como los datos que estén guardados allí donde el sistema mantenga una instalación. Ambos son la misma cosa. Por lo tanto el daño afecta siempre a su copia original. Esto se hace con el fin de mantener un juego simple. Tratar de emular la realidad de sus complejidades no tiene mucho sentido si eso no ayuda a crear un juego divertido. Pero muchos jugadores podrían desear que existiesen otras formas de entender este concepto ya que se trata de algo que varía mucho según el tipo de escenario. Hay muchas formas de concebir lo qué significa una copia y cómo funciona.

Por ejemplo, en el escenario de un videojuego un personaje no posee un número de copias tal y como entendemos la copia de respaldo de un programa, sino que las propias leyes del videojuego le otorgan una cantidad de oportunidades o "vidas" que han sido convenidas por sus diseñadores. Otro caso podría ser que la copia de una IA muy poderosa resida en un número determinado de unidades de almacenamiento especiales de gran capacidad repartidos por el mundo físico. Dada la complejidad de la IA sólo es posible almacenarla en una de esas unidades. Al tratarse de un hardware muy avanzado este tipo de soporte es muy raro por lo que podría suceder que sólo existieran dos o tres de ellos con una copia del programa cada uno. Si se pierden las unidades se pierde el programa para siempre.

Por eso existe el módulo de Copias o "Backups". Este módulo es opcional y sirve para poder llevar la cuenta de las veces que se ha **restablecido la copia original** de un programa a partir de sus **copias de respaldo**. Este módulo se puede entender como el número de copias, clones, vidas u oportunidades. Lo que mejor encaje con la ambientación que se esté jugando. Nada más que formas distintas de expresar una misma función. La copia permite volver a reinstalar el programa con una versión de su código en buen estado.

Siempre se asume que las copias de un programa se actualizan constantemente, lo que incluye todo cuanto haya aprendido y las mejoras que hubiese recibido. Aunque también con los defectos que sufra durante su ejecución, como la pérdida de puntos en sus Operaciones por ejemplo. Por otra parte, los efectos de la visita de La Libélula se transmiten a todas las copias que el programa pueda tener almacenadas.



El módulo de copias suele funcionar bien junto al de Integridad ya que es frecuente usar ambos en los mismos escenarios. Es especialmente útil en los que estén basados en videojuegos. En la mayoría de ellos el programa puede disponer de tres a cinco oportunidades, con la posibilidad de ganar más si consigue superar una serie de hitos, encontrar objetos clave o superar las distintas fases del juego.

En otros, como en los MMORPG, se llega a contar con copias infinitas del personaje. En esos juegos las consecuencias de la derrota se produce a otros niveles por lo que en condiciones normales el personaje nunca se pierde, aunque también puede suceder.

Muchos otros escenarios pueden prescindir del módulo. En un mundo virtual ultradetallado (NdE: como el de las películas Matrix) el personaje sólo tendría una oportunidad en caso de ser un usuario humano conectado a la simulación, por lo que no sería necesario usarlo. En este ejemplo, por razones de la ambientación el usuario que controla al programa sufre también las consecuencias, pudiendo morir junto a su representación virtual. Pero como ves, eso es algo que se produce a nivel narrativo. Es decir, se decide que así sea por razones de la trama. Los personajes basados en programas que sean nativos del sistema no tendrían que sufrir estos inconvenientes. En otras aventuras usando el mismo enfoque, la copias de la consciencia digitalizada de un usuario podrían alojarse en los sistemas informáticos de algunos sitios muy concretos como los de ciertas universidades de prestigio, laboratorios privados o departamentos del gobierno.

Pero Scroll anima a los jugadores a contemplar ambientaciones híbridas, mezclando varios tipos de escenario, por lo que es conveniente tener en cuenta este módulo ya que el número de copias podría variar a lo largo del juego. Un personaje que pasara de "la rejilla de un juego" a otras secciones del Sistema podría pasar de tener una cantidad de copias a no tener ninguna, o viceversa.

El número de copias se anota en la casilla grande de este bloque, disponible en la ficha del personaje. De no tener ninguna lo conveniente es anotar un cero "0". En el caso de tratarse de un juego en línea, por ejemplo de acción en línea o un MMORPG, lo normal es que el programa disponga de oportunidades infinitas. Si el programa puede cargarse de forma indefinida se puede indicar en la ficha con el símbolo de infinito: " ∞ ". Las casillas más pequeñas se utilizan para llevar la cuenta de las copias usadas.

Existen varios métodos para que un programa pueda hacer copias de sí mismo o de otros programas, como el comando /COPY por ejemplo. Pero por lo general para hacer una copia es necesaria la autorización de un usuario con privilegios que lo avale o contar con el permiso del Sistema. Cada sistema tiene sus propias leyes y conviene tener en cuenta si esto es o no un requisito. De producirse una copia sin autorización un sistema lo puede interpretar como una Anomalía al creer que es un virus o un ataque externo, por lo que

conviene tener mucho cuidado. De ello se deduce que si se encuentra algún modo de ocultárselo pueden existir copias ilegales. Un programa sin copias disponibles podría hallar algún medio de hacerlas. Este es el tipo de cosas que ayudan a crear historias y debería ser material para las aventuras del juego.

Siempre que un programa recurre a una de sus copias es porque ha sido destruido, su código se ha visto comprometido o se ha visto forzado a un reinicio. **Un reinicio vacía completamente el búfer y todos los datos del programa que estuviesen cargados en la memoria.** Además, requiere tiempo, por lo que el jugador debe esperar a que termine la escena en curso para que su personaje vuelva a estar operativo. En el tiempo del mundo físico el reinicio no tiene que durar más que unos pocos segundos, pero en el tiempo subjetivo de los programas en ese lapso pueden pasar muchas cosas. En unas pocas milésimas de segundo un grupo de programas pueden haber librado un combate con todas las consecuencias. Por eso, durante su ausencia el programa queda fuera de la escena hasta que ésta haya terminado.

Reiniciar también tiene otro inconveniente. Aunque gran parte de la información que el programa ha almacenado y aprendido se ha ido guardando, en muchos escenarios se ve forzado a reaparecer en una localización del espacio abstracto determinada. Es posible no obstante simplificar las cosas y estipular que el programa siempre reaparece en el mismo lugar dónde se vio forzado al reinicio. Si este es el caso puedes omitir el siguiente apartado, donde se explica cómo gestionar el "**Área de reinicio**". Si no, que es lo que recomienda este juego, los jugadores asumen que al reiniciar sus personajes reaparecen en otra localización. Esto sucede en la mayoría de los mundos de juego y en los entornos virtuales, y es poco frecuente en una ambientación esquemática realista o simbólico abstracta. Como siempre, la localización exacta del reinicio depende del escenario.

Cargando varias copias a la vez (instancias)

Para terminar, en Scroll se asume por conveniencia que un programa carga una única copia en su sistema. Se entiende que la sofisticación y complejidad de los programas es tal que sólo es posible contar con una sola copia cargada en la memoria, lo que en la jerga informática se denomina una "**instancia**" del programa. Pero si a los jugadores y al Director les parece bien, nada les impide

contar con más de una instancia al mismo tiempo. Es más, en este juego existe un Comando que lo permite por si te interesa contemplar esta posibilidad.

De este modo podrían coexistir varias instancias —o copias— funcionando al mismo tiempo. Cada una llevando a cabo acciones distintas y poniéndose de acuerdo entre ellas. ¡Lo que no significa que tengan que llevarse bien unas con otras! Eso sí, una vez todas las copias han sido destruidas el programa desaparece para siempre. Si esto le da demasiado poder a un personaje, para limitarlo puede establecerse que las instancias solamente pueden funcionar en sistemas distintos; lo que nos conduce al concepto de "Mundos paralelos"...

Mundos paralelos

Se cuenta entre los programas que el concepto de copia y de instancia se aplica también a los entornos de los sistemas y a los mundos virtuales. Según se cree, el medio digital en el que existe un programa puede tener múltiples clones, o copias, que funcionan como las instancias de un programa. Es decir, siendo capaces de permanecer varios activos al mismo tiempo funcionando en paralelo sobre un conjunto de servidores. Los servidores son poderosas configuraciones de hardware (ordenadores al fin y al cabo) que permiten el funcionamiento de los sistemas más potentes.

De ser cierto, el entorno digital donde existe un programa podría tener un número ilimitado de réplicas exactas que incluirían las de todos los programas que hay en él funcionando a la vez. En cada uno existiría una copia de un mismo programa llevando una existencia alternativa. O sea, copias del mismo personaje haciendo cosas distintas al mismo tiempo. Quizás su programación y funciones sean las mismas, pero al tomar distintas decisiones en cada uno de esos mundos su existencia podría tomar caminos diferentes. De encontrar un modo, un programa quizás podría también visitar uno de esas réplicas de su mundo y observar qué rumbo ha tomado el curso de los acontecimientos... Como ves, las posibilidades de nuestro universo digital son infinitas.

ÁREA DE REINICIO

El área de reinicio se puede anotar en la ficha en el espacio que existe reservado para ello. Es tan sencillo como decidir un lugar en el caso de que el Director disponga de un mapa del escenario. De no tener ninguno se puede convenir alguno basándose en lo que el Director del juego considere apropiado.



Es suficiente con describir mediante una frase algún lugar en concreto. Por ejemplo: antes del cortafuegos; en el canal VPN; al inicio del escenario o si el escenario es un complejo: en la puerta de entrada; en la posada; en la puerta del restaurante; en el área de inicio designada para los visitantes; en el área pública; frente al ayuntamiento; a cincuenta pasos de la nave espacial estrellada; en una casa determinada; en la plaza..., etc. En caso de no haber ninguna fijada de antemano siempre se asume por defecto como la coordenada 0, 0, 0 del mundo de juego si este dispone de un espacio matemático definido; como sucede con todos los mundos virtuales y la mayoría de los juegos. La descripción de lo que significan las coordenadas se explica en el apartado "Áreas de influencia".

Muchos entornos de juegos suelen fijar una posición única. Por ejemplo, al principio de la fase o junto al último "punto de control". Por lo que en muchos de ellos el personaje debe volver a recorrer todo lo que hubiese hecho la vez anterior, desde el punto de restauración hasta donde se encontraba en el momento del reinicio. Esto funciona como una especie de bucle temporal. Aunque el personaje puede recordar todo lo que ya ha hecho en su carga anterior se ve forzado a experimentarlo todo de nuevo. Esto puede ser muy divertido. Tiene la ventaja de que el personaje es capaz de "retroceder en el tiempo" y experimentar de nuevo una misma situación que ya haya vivido siendo capaz de aprender de sus errores, lo que le permite superar los obstáculos con una estrategia diferente. (NdE: Un recurso que ya se ha visto en películas como "Atrapado en el tiempo" o "Al filo del mañana"). De este concepto pueden surgir tramas muy interesantes y aventuras memorables, aunque es conveniente no abusar de este recurso muchas veces.

Otros sistemas de juego permiten fijar puntos de salvación que evitan tener que repetirlo todo de nuevo. Normalmente disponen de algún sistema para poder hacerlo que es de fácil acceso para el personaje. En unos el punto se fija automáticamente sin que éste tenga que estar pendiente de salvar una y otra vez su posición; en otros solamente al llegar a determinados **puntos de control**, y en muchos otros en cualquier momento, pero sólo si se acuerda de hacerlo el jugador. Los puntos de control disponibles se pueden señalar convenientemente en el plano de la aventura o del escenario. También es posible forzar un punto de salvación (siempre que esté permitido) utilizando el comando /SAVE (salvar). El uso de los comandos se explica más adelante.

En los mundos virtuales, MMO y MMORPG es frecuente que se puedan elegir con libertad unas coordenadas del mundo del juego para establecer un Área de reinicio. Cada uno brinda herramientas propias para poder hacerlo pero en el caso de no estar disponible es posible forzarlo haciendo uso del comando /BIND (fijar posición). En el capítulo dedicado a las ambientaciones se ofrecen más detalles.

OTRAS FUNCIONES

Las demás secciones que aparecen en la ficha simplificada no son módulos en realidad. Se trata de algunas funciones extra. En esta sección es suficiente con conocer la función ESC (Escape).

FUNCIÓN "ESCAPE" (ESC)

La función **ESC** (Escape) está disponible en todos los programas. Al activar la función el programa envía una instrucción directamente a los procesadores que interrumpen todas las operaciones que estuviese realizando. Su cometido es abortar todas sus funciones y detener su acción. En términos de juego esto tiene dos posibilidades. Cuando el jugador invoca esta función el programa **huye** o bien interrumpe todas sus funciones y **se rinde**.



ESCAPE

Los sistemas tienen también acceso a la función, aunque es raro que la mayoría la utilice. Eso depende de su comportamiento. La función **ESC** en los sistemas detiene cada proceso por separado. En el caso de haber enviado un grupo de agentes de control por ejemplo, éstos se detendrían permaneciendo estáticos, bajo un protocolo de función mínima o describiendo círculos siguiendo la pauta de una función de bucle; sólo en algunos casos huirán. Muchas amenazas en cambio optan por huir primero y si no les es posible hacerlo es entonces cuando se rinden.

Si un programa está en serias dificultades el jugador puede invocar **ESC**. Para hacerlo es conveniente que **lo diga en voz alta** (y a ser posible que pulse también la tecla de su ficha). Debe quedar muy claro lo que pretende porque si las amenazas continúan con sus acciones y éstas tienen consecuencias adversas para el programa ya no hay —o no debería haber— vuelta atrás. El Director le pregunta entonces al jugador qué es lo que quiere hacer: si huye, siempre que aún pueda hacerlo, o si se rinde.

A. FUGA

“¿Sus-piráis Milady?”

Anónimo

En el caso de optar por la huida el personaje recurre a sus Protocolos de respuesta forzando al máximo su **Función de salida**. Al hacerlo pierde la capacidad de volver a recurrir a esta función **el resto de la sesión de juego**, por lo que si más adelante desea hacerlo tendrá que usar cualquiera de las otras tres funciones. Las tres casillas se consideran nulas y es conveniente tacharlas (a lápiz) para recordarlo hasta que se hayan recuperado. Por otra parte, la opción de fuga fuerza a las CPU por lo que automáticamente **el personaje gana 1 punto de CPU** que debe señalar también en su ficha.

El programa en fuga **huye a la máxima velocidad de la que es capaz** abandonando la confrontación y siempre usando la ruta más corta, tenga esto las consecuencias que tenga. El uso de la Función Escape para huir casi siempre garantiza que el personaje tenga éxito al hacerlo. Aunque sólo una vez por sesión y pagando un precio. Eso sí, sólo es posible usarla **siempre que haya una ruta de escape**. No es posible usar la función ESC para escapar por ejemplo de una prisión (NdE: no te pases). Por lo general es muy difícil impedir que el personaje consiga huir, pero ante circunstancias especiales el Director puede exigir alguna prueba o considerar que no es posible. No obstante se recomienda, y es más divertido si se hace, que el programa consiga escapar de la forma más inimaginable de los lugares más inverosímiles; aun cuando en otras circunstancias hacerlo sería “casi” imposible. De ello surgen situaciones memorables.

Función O/D	Función de salida	Función mínima	Función aleatoria
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

B. RENDICIÓN

“¡Me rindo!, ¡me rindo! Digo... ¡Nos rendimos! ¡Que nos rendimos! ¿¡Pero qué parte de nos rendimos no entiendes cabeza de hojalata!? ¡Que te estoy diciendo que me rindooooo!”

Dr. Who

Esta función es una herramienta para la narración y así deben entenderla todos los jugadores. Se basa en realizar un pacto de conveniencia para todas las partes implicadas de forma que los acontecimientos puedan seguir otro rumbo. Si un bando considera que va a ser derrotado tiene la posibilidad de declarar su deseo de rendirse, lo cual puede ser una buena decisión en una situación desesperada.

Rendirse invocando la función ESC implica que el o los programas deben aceptar las condiciones de lo que suceda. Rendirse no garantiza poder escapar o salir impune, pero sí que la situación tome otra dirección, para bien o para mal, y no existen garantías. Es una mecánica narrativa que interviene permitiendo poder continuar sin que necesariamente los acontecimientos tengan que terminar con la destrucción de un bando.

Los jugadores, en especial el Director de juego, deberían aprovechar este recurso para hacer avanzar la trama y favorecer el drama. La posibilidad de destruir a los personajes siempre está ahí, pero lo conveniente es tomar alguna dirección que beneficie a la historia. A no ser que lo conveniente sea su destrucción claro está... Pueden existir variantes de esta regla que siempre permitan al bando que se ha rendido poder escapar o evitar la destrucción, en Scroll esa posibilidad no es una imposición, sólo una sugerencia.

Al rendirse los jugadores y el Director pactan las condiciones entre todos. Rendirse siempre implica algún problema para los que opten por esta opción. El bando ganador obtiene alguna ventaja, como superar algún obstáculo que sea importante para el desarrollo de la trama por ejemplo, tener las posibilidades más a su favor para cumplir sus metas, avanzar un poco más hacia su objetivo o conseguir algo que anduvieran buscando. Recuerda que los agentes del Sistema y otros programas hostiles también tienen derecho a rendirse. En ese caso suele ser el Director el que decide el curso de la acción

pero lo verdaderamente interesante es que entre todos se decida qué es lo que sucede.

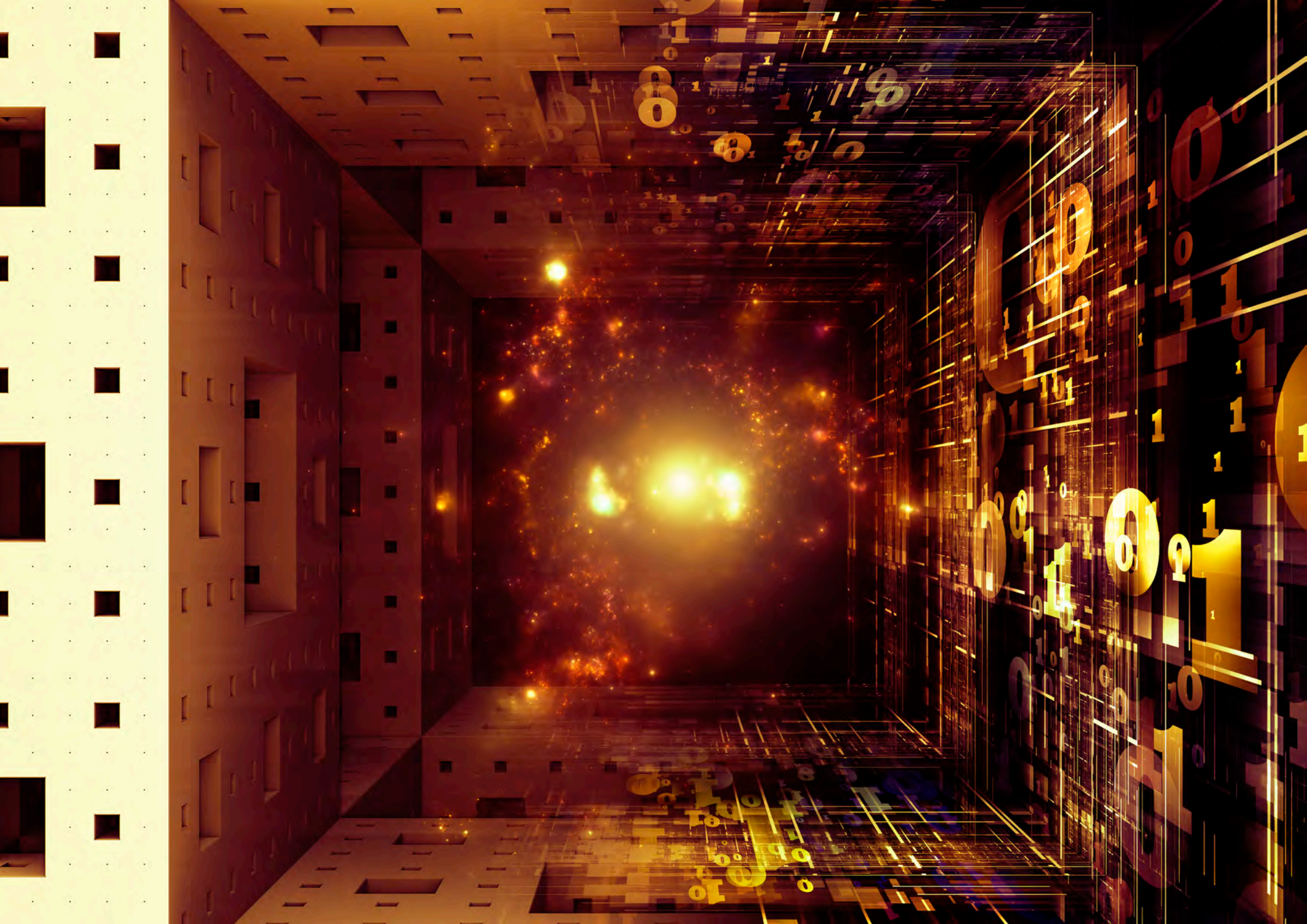
Pase lo que pase la función ESC tiene también sus ventajas. Por cada personaje que se haya rendido el grupo entero gana un "**Hack**". Los puntos Hack siempre son de grupo; todos pueden beneficiarse de ellos. Estos puntos se explican en su sección correspondiente más adelante. Si por ejemplo han sido tres jugadores los que se han rendido el grupo ganaría tres Hacks.

Además, se añade otro punto más por cada bloque de corrupción del código en el que un programa haya puesto alguna marca. Si por ejemplo un programa ha puesto una marca en el grupo A (amarillo) y otro ha completado el grupo A y ha ocupado el B (naranja) con otras dos marcas, esos dos jugadores reciben un total de tres puntos más. Uno por cada bloque ocupado. Si quienes se han rendido han sido las amenazas, el Sistema obtiene un punto de anomalía por cada una. El Director del juego podrá utilizar esos puntos con normalidad.

Hay sistemas capaces de anular las funciones del recurso ESC. El programa puede verse incapaz de poder invocarlo directamente. Por suerte, siempre puede recurrir a los comandos, que son llamadas indirectas (o por otras vías) a sus funciones. El comando /ESC (Escape) suele estar disponible si todo lo demás falla. Pero forzar un comando no es siempre una tarea fácil. Casi todos requieren superar una dificultad mediante una tirada.

FIN DE LÍNEA





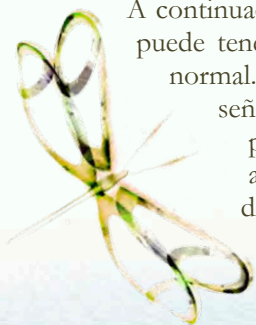
FUNCIÓN 3

"PROCESOS AVANZADOS"

—“Los humanos tienen sueños. Hasta los perros tienen sueños, pero no tú. Tú eres sólo una máquina. Una imitación de la vida. ¿Puede un robot escribir una sinfonía? ¿Puede un robot convertir... un lienzo en una obra maestra?”

—“¿Podría usted?”

Yo robot



A continuación se describen todos los procesos y recursos avanzados que puede tener un programa. Todos se encuentran disponibles en la ficha normal. Los procesos que estén calificados como reglas opcionales se señalan con una etiqueta junto a su nombre. La mayoría de estos procesos tienen como función complementar algunas ambientaciones. Cada ambientación detallada en el capítulo de diseño de sistemas especifica cuales se pueden usar. Cuando se usa un módulo de reglas calificado como opcional es posible poner una marca en la **casilla de uso**, siempre disponible en la ficha, para recordar que se están usando sus reglas en la partida.

FUNCIONES AVANZADAS

GESTIÓN DE LA MEMORIA

Todos los programas necesitan espacio en la memoria del Sistema. Cuando se ejecutan siempre ocupan un espacio mínimo, el necesario para poder funcionar y realizar sus funciones básicas. Pero muchos procesos avanzados exigen el uso de memoria extra. La memoria siempre la concede el Sistema y

no es ilimitada. Un programa **no puede ocupar toda la memoria que quiera** a no ser que disponga de ese privilegio. Hay programas con esa capacidad pero son raros.

La memoria es autónoma hasta cierto punto. Tiene la capacidad de autogestionarse por sí sola funcionando como un subsistema independiente. La función del Sistema es vigilar que se cumplan sus leyes pero su autoridad termina donde empieza la de ella. Todos los programas tienen derecho a realizar la solicitud de reservar espacio, siempre con un límite máximo denominado cuota que como ya se ha visto es igual a dos veces el valor de la operación de Control del programa (Control x2). Mientras no se sobrepase la cuota el Sistema está obligado a conceder la reserva.

La memoria como la voluntad o el rendimiento

En algunas ambientaciones, especialmente en los mundos virtuales, el uso de la gestión de la memoria puede equivaler a **la voluntad** del personaje y su resistencia a la presión. Cuando recurre a ella el personaje está llevando sus capacidades al límite lo que lo acerca siempre a su punto de ruptura. Por otra parte puede reflejar también el **rendimiento** de una máquina o de una configuración de hardware operando sobre la realidad del mundo físico.

Si se trata de un personaje, su gestión permite tener el control sobre las fuerzas que le empujan a continuar. Las distintas crisis que va sufriendo pueden pasarle factura por lo que cada vez se derrumbará con más facilidad ante la presión.

Si se trata de una estructura de hardware refleja cómo el uso de sus recursos afecta a su funcionamiento. La estructura fuerza sus capacidades al máximo para ganar eficacia al margen de la energía invertida, en consecuencia se pueden producir problemas o incluso errores graves al sobrepasar sus límites.

El espacio en la memoria **se mide en unidades o bancos de memoria**, que serán necesarios para que un programa pueda aumentar su rendimiento o ejecutar sus funciones más complejas y avanzadas: **las Utilidades**. Cada unidad equivale a **un dado** de la reserva. En realidad un programa puede destinar memoria extra a cualquier función que realice, ya sea general o específica, pero **las Utilidades siempre exigen memoria extra para poder**

funcionar ya que se trata de operaciones que necesitan de una gran cantidad de espacio para almacenar la información que necesitan. Por esta razón si no es posible reservarla el programa no puede usarlas. La cantidad de espacio que puede reservar el programa por otro lado determinará la potencia máxima y la eficiencia de sus resultados.

El uso de la memoria es delicado. Las Utilidades fuerzan al programa al límite de sus capacidades. Si no está correctamente depurado es común que surjan errores. Su gestión es además bastante compleja por lo que con frecuencia quedan datos almacenados en los bancos que el programa no consigue eliminar correctamente. Esos datos se corrompen y pueden dañar el funcionamiento de las Operaciones. Su mala gestión o el uso intensivo de la reserva máxima de memoria pueden ralentizar al programa o empujarlo a tener fallos que le obliguen a cerrarse.

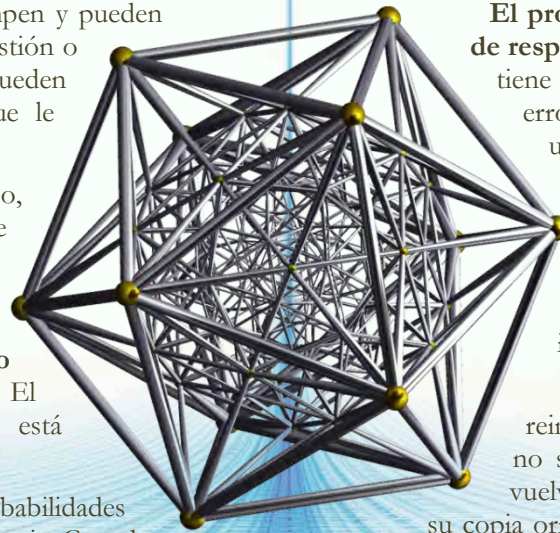
A diferencia de la solicitud del Tiempo de proceso, que exige ir de uno en uno, la reserva de Memoria puede hacerse con libertad mientras no se sobrepase el límite. En términos de juego **el jugador puede decidir usar tantos dados como quiera, pero siempre por debajo de la cuota máxima.** Además, **los dados no permanecen en la reserva** como lo hacen los de CPU. El jugador puede usarlos como crea conveniente y no está obligado a mantenerlos en las siguientes tiradas.

Solicitar espacio extra de memoria aumenta las probabilidades de éxito pero siempre supone un riesgo que hay que asumir. Cuando en la tirada domina la reserva de la memoria significa que el programa ha cometido un error. Puede deberse a que al forzar el programa aparece alguno que aún no se ha corregido; que ha perdido el control sobre la memoria por el momento lo que le impide enviar o recibir datos o que se ha cometido un error al gestionarla, algo que sucede con frecuencia en programas poco optimizados. Puede deberse a muchas razones. La pérdida del control de la memoria y los errores en su gestión provocan que el programa actúe de forma errática. Por eso todos los programas disponen de sistemas para poder controlarlo y reparar la información perdida. La recursividad, que consiste en

tomar caminos alternativos para realizar una misma función, o el control de paridad, que permite corregir datos erróneos, son algunos de esos mecanismos. Estos sistemas de control ponen al programa en una serie de "modos de respuesta". Por eso **cada vez que domina la reserva de memoria el jugador está obligado a poner una marca sobre una de las casillas de sus Protocolos de respuesta.** Cada modo tiene unos efectos que son resultado de los procesos que intervienen por lo que el personaje deberá actuar en consecuencia. Los protocolos se explican en su apartado correspondiente en este mismo capítulo.

El programa puede usar tantas casillas para sus Protocolos de respuesta como su Nivel. El resto se ignoran. Cuando ya no tiene más casillas que poder marcar el programa empieza a tener errores y a actuar de forma caótica hasta que finalmente sufre un **fallo grave** y **se cierra**. Durante el transcurso del cierre el jugador debe asumir algún tipo de comportamiento errático e interpretarlo si es posible. Cuanto más divertido sea mucho mejor. La duración de este comportamiento ocupa el resto de la escena en la que se ha producido el fallo. Al final el programa siempre colapsa irremediamente y se cierra sin que nada pueda evitarlo.

Cuando se produce un **fallo grave**, y a diferencia de un reinicio partiendo de una copia nueva, los datos del programa no se pierden (o al menos no la mayor parte). El programa vuelve a cargar normalmente en su Área de reinicio a partir de su copia original, pero arrastrando las consecuencias de haber sufrido un fallo grave. Partes del código que estaba cargado en el Sistema se corrompe y queda ocupando parte del banco de memoria donde hubiesen estado guardados los datos en el momento de producirse el fallo. La consecuencia de todo esto es que **el programa se reinicia con un dado menos de Operaciones** (uno permanente en una operación que el jugador decida) y uno de **sus espacios de memoria quedan ocupados de forma permanente por un dado**. Este dado siempre se lanzará obligatoriamente junto al resto de los dados que el jugador quiera añadir de forma temporal a su reserva de memoria. Por otro lado, tras el reinicio el programa recupera todos sus



espacios disponibles en sus Protocolos de respuesta por lo que el jugador puede eliminar todas las marcas que haya puesto.

Como puedes ver los daños continuos van perjudicando el rendimiento del programa cada vez más. Pierde efectividad y aumentan las posibilidades de que surjan fallos de memoria, lo que lo acerca más y más a que se produzca el **fallo total** que le haga perder toda su capacidad de realizar operaciones. Aún es posible purgar la memoria y limpiarla de datos corruptos. El procedimiento se explica más adelante.

Cuando se produce un **fallo total**, es decir pierde todos sus puntos en las dos Operaciones, el programa deja de estar operativo y es asimilado por el Sistema, que puede decidir su destino como quiera. En algunos casos se convierte en un agente con un Nivel de desafío igual a la máxima puntuación que alcanzó en el juego en su Operación más alta antes de comenzar a perderla. En otros casos, especialmente cuando el programa está muy dañado, es borrado por completo incluyendo todas sus copias existentes. En todos los casos esto supone la muerte del personaje o que quede fuera de juego.

MEMORIA		El espacio de memoria se mide en bytes. Normalmente a partir de "gigas" o "teras". Eso depende del nivel de desarrollo tecnológico de los sistemas informáticos de la ambientación. Cada banco de memoria señalado en la ficha equivale a la unidad. Por ejemplo, dos unidades de memoria podrían valer 2 exabytes. En un escenario más futurista y avanzado estaría en el orden de los "zettabytes" o "yottabytes".
megabyte (MB)	10 ⁶ bytes	
gigabyte (GB)	10 ⁹ bytes	
terabyte (TB)	10 ¹² bytes	
petabyte (PB)	10 ¹⁵ bytes	
exabyte (EB)	10 ¹⁸ bytes	
zettabyte (ZB)	10 ²¹ bytes	
yottabyte (YB)	10 ²⁴ bytes	Esta información sólo se indica para ayudarte a describir la ambientación. No estás obligado a tenerlos en cuenta.

UTILIDADES

Las Utilidades son talentos especiales que poseen algunos programas. Son un complemento de sus Rutinas para poder realizar con eficacia las funciones para las que han sido diseñados. En el mundo de los programas permiten que éstos puedan llegar más allá de lo que pueden hacer normalmente y romper las reglas. Les posibilita poder marcar la diferencia y demostrar que sus funciones especializadas los convierten en herramientas únicas pues han sido creados con un fin específico. Por lo tanto, y a diferencia de las Rutinas, un programa sólo puede ejecutar aquellas Utilidades que posea.

En cuanto a la comprensión de lo que es una Utilidad sucede del mismo modo que con las Rutinas. Se trata de un concepto que tiene muchas interpretaciones pues depende del enfoque de la ambientación y del contexto del personaje. Una Utilidad puede ser cualquier talento que lo haga especial. Podría consistir en un sofisticado algoritmo con el que cuenta el programa para realizar una función especializada, una potente utilidad de software que un usuario operando en el sistema reserva como arma secreta, el ataque especial y único del personaje de un juego o bien una de las capacidades específicas que están incorporadas en el chasis de un robot, como un arma o la capacidad de volar por ejemplo.

De igual modo, la interpretación de una misma Utilidad también dependerá del contexto de los personajes. Las acciones se resuelven utilizando las mismas mecánicas de juego, pero al visualizar la acción y describir el resultado todo puede suceder de formas distintas. Por ejemplo, dos personajes que contaran con una misma Utilidad llamada "Perforadora", situados uno en el contexto del mundo físico y otro en el mundo digital, interpretarían de diferente manera el uso de la Utilidad. En el mundo físico el personaje podría tener literalmente un "taladro" incorporado a su cuerpo, montado en el chasis sobre un brazo por ejemplo, mientras que la versión digital contaría con un poderoso rayo de energía o incluso con un taladro de dimensiones desproporcionadas en relación al avatar del personaje al estilo de los dibujos animados.

Las Utilidades siempre dependen de la memoria por lo que el personaje debe ocupar unidades si quiere utilizarlas. Por lo tanto, si un jugador desea usar alguna Utilidad debe poner **al menos un dado** en la reserva o no funcionará. Esa tirada determina la eficacia del uso de la Utilidad. En unos casos basta con tener en cuenta los resultados obtenidos, en otros se compara con una dificultad impuesta por el Director.

El número de dados que añada el jugador aumenta su poder, alcance y eficacia. Esto significa que, por ejemplo, si tu programa funcionando en un mundo virtual ultradetallado es capaz de correr por las paredes mientras dispara sus pistolas, dos dados supone recorrer un tramo corto (como una habitación), cuatro dados una distancia razonable (como un pasillo) y seis o siete dados un tramo bastante largo (como salir de un edificio o correr por la pared de un almacén). Si la Utilidad consistiera en dar un gran salto un dado significa saltar unos pocos metros y varios, como tres o cuatro, saltar entre dos edificios.

Cuando un programa desea utilizar sus Utilidades el jugador debe fijar el precio y aceptar las consecuencias de la tirada. El valor de ese riesgo está impuesto por la complejidad de la tarea y determinará la oposición que el Director debe fijar para el Sistema en el caso de que la haya. En algunos casos, dependiendo de la naturaleza de la Utilidad y de cómo haya sido pensada al crearla, el coste del precio a pagar puede tener una escala —como en los ejemplos anteriores de correr por las paredes— o tratarse de un valor fijo. Por ejemplo, una Utilidad que permitiera al programa detectar amenazas inminentes dentro de su área de influencia podría costar siempre el mismo número fijo de unidades de memoria.

Las Utilidades pueden basar sus operaciones en la Potencia o en el Control. Existen tanto para una como para la otra. Técnicamente no hay límite en la cantidad de Utilidades distintas que pueden estar disponibles. Es posible crear cualquier que parezca apropiada; no obstante, el programa sólo tiene acceso a un número limitado de ellas. A la hora de crearlas es frecuente que se parezcan a las Rutinas. En realidad eso no tiene importancia. La clave está en que puedan permitir al programa realizar acciones que normalmente otros no puedan ya que ha sido diseñado para ese fin. Así pues, una Utilidad puede llevar a lo que es posible hacer con una Rutina mucho más allá, permitiendo al

programa quebrantar las reglas, cruzar la línea y hacer lo que para los demás resulta imposible. Precisamente por eso, su uso siempre supone un riesgo para los programas.

No olvides que la elección de la o las Utilidades se puede hacer durante la partida si lo prefieres, no necesariamente al crear el personaje. También puedes elegirla más adelante si resulta coherente con el desarrollo de los acontecimientos. Un personaje puede descubrir que la tiene o aprenderla en el transcurso de sus aventuras ya sea porque desconoce que posee esa facultad o porque consigue adquirirla en algún momento¹. En el Capítulo 5 se ofrecen más detalles sobre la creación de Utilidades.

EJEMPLO

Andrómeda tiene una Utilidad que le es de gran ayuda en sus misiones. Es capaz de hacer tal nivel de acrobacias que si se tratase de una persona del mundo físico quienes la vieran pensarían que tiene poderes sobrenaturales. Sus saltos le permiten salvar la distancia entre dos edificios, hacer acrobacias imposibles e incluso correr por las paredes durante tramos cortos. Hay muy pocos programas capaces de romper así las leyes del sistema de su mundo virtual, y siempre por alguna razón.

Cuando es sorprendida por dos agentes su jugador decide que huirá subiendo por las escaleras hacia la azotea y que de allí tratará de saltar a los edificios vecinos. Cuando llega arriba corre hacia el borde y salta. La distancia es enorme. La Directora del juego le dice que el salto es difícil por lo que debe obtener al menos 4 éxitos en la tirada. El jugador lanza 3 dados de Control y decide recurrir a la Memoria para aumentar su eficacia por lo que añade 4 dados más. La reserva total es de 6 dados. Un uso intensivo de la Memoria siempre tiene un riesgo pues es cuando el programa fuerza sus capacidades al máximo. Pero la necesidad se antepone a todo lo demás.

Hace su tirada. En su reserva de Control obtiene 1, 3 y 4. En la de la memoria obtiene un 6, 4, 3 y 2. Ha obtenido 4 éxitos por lo que Andrómeda realiza su salto aterrizando muy justa junto al borde. Por poco. Pero el éxito ha tenido su precio pues la reserva de la Memoria es la **que domina** al haber sacado un 6. El valor más alto de todos los resultados. Se produce un pequeño error que Andrómeda deberá compensar mediante sus Protocolos de respuesta.

¹ Como el protagonista en las películas "Matrix" por ejemplo.

PROTOCOLOS DE RESPUESTA

El uso de la memoria y de las Utilidades implica recurrir a los Protocolos de respuesta para hacer frente a los errores que puedan producirse durante su gestión. Como ya se ha explicado **reflejan el comportamiento de un programa cuando trata de recuperarse y seguir funcionando**. También sirven como uno de los tres métodos disponibles para hacer frente a los daños recibidos en el código del programa por lo que complementan los recursos que tiene un programa para compensarlos.

“Nunca cometo errores estúpidos. Sólo errores inteligentes, muy inteligentes”

Dr. Who

Se divide en cuatro columnas con tres opciones cada una. Cada columna expresa un modo distinto de comportamiento según el tipo de procedimiento que intervenga durante el proceso de control de errores. **Cuando se produce un error el jugador debe poner una marca en una de sus casillas. Cuando se usa como respuesta contra una amenaza el jugador debe poner una**

marca sobre la casilla que le indique el Director. Sólo en el segundo caso pierde la libertad de poder elegir.

Es posible usar **tantas casillas de los Protocolos de respuesta como Nivel tenga el programa**. Una vez el número de casillas marcadas lo iguale, el programa deja de funcionar con eficacia. Comienza a actuar de forma caótica hasta que se produce **un fallo total** y se cierra.

Las cuatro respuestas posibles son:

- Función **ofensiva** o **defensiva**
- Función de **salida**.
- Función **mínima** o modo de compensación.
- Función **aleatoria** (random) o **bucle trampa**.

Pero, ¿qué casillas se marcan primero? Eso es algo que **depende del estilo del programa y de sus vulnerabilidades**. El jugador **debe asumir la responsabilidad** de optar por la opción que crea más conveniente para su programa partiendo de cómo lo haya definido. Esto requiere seriedad por su

parte, pero no tiene porqué hacerlo sin ayuda. Todos los jugadores, incluyendo el Director, pueden participar haciendo sugerencias sobre cuál es la reacción más probable que podría tener su personaje. El Director puede supervisar esa elección decidiendo si le parece adecuado o no. Cuando las casillas más probables hayan sido ocupadas se continúa con las que parecen más razonables, y así hasta completar todas las que tenga disponibles.

Existen doce casillas para permitir programas con niveles altos en el juego pero es raro llegar a ocupar más de diez. En caso de que necesites más (poco probable) sólo hay que extender la longitud de cada columna hacia abajo (de 3 a 4, de 4 a 5, etc.).

Función O/D	Función de salida	Función mínima	Función aleatoria
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

¡Recuerda! El límite máximo de casillas que puede usar un personaje es igual a su Nivel.

Función ofensiva/defensiva

El efecto de las rutinas de corrección de errores activan las funciones de supervivencia del programa, desconectando otras para poder ahorrar procesos. El programa asume entonces que la mejor salida es un enfrentamiento directo contra la amenaza. Aún tiene la opción de decidir si actuar de forma ofensiva o defensiva, pero en todos los casos confrontará a su agresor. Si la amenaza no es visible pasará a modo de búsqueda hasta que lo encuentre o haya desaparecido. El personaje por lo tanto luchará intentando hacer todo el daño posible o evitándolo a toda costa. El jugador puede optar por una de las dos posibilidades. Solo cuando la amenaza haya sido derrotada, se marche o se rinda podrá volver a su régimen normal de funcionamiento. En el juego el programa debe actuar de esta forma hasta que termine la escena en curso.

Función de salida

Los protocolos recurren a la primera directriz, la más básica que posee todo programa y que ha sido diseñada para perpetuar sus funciones al máximo. En otras palabras: sobrevivir. El programa dedicará unos breves instantes a buscar la ruta de escape más óptima y hará todo lo posible por usarla. La ruta puede ser compleja y no ser la más adecuada a nivel táctico, pero aún así dará prioridad a esa opción sobre todas las demás. En términos de juego el programa hará todo lo posible por huir, separándose de sus compañeros hasta que termine la escena en curso. **Huir no es lo mismo que rendirse.** El programa intentará alejarse del área de influencia de la amenaza para evitar el conflicto, pero no pactará con ésta o se someterá a sus deseos.

Función mínima o Modo de compensación

El programa muestra un funcionamiento anómalo mientras sus protocolos tratan de redirigir toda la capacidad de proceso disponible a reconfigurar su código. Esto lo deja tan sólo con sus funciones mínimas para poder operar. Se mostrará torpe y lento mientras compensa, ahorrando procesos para poder recuperarse. El programa **sólo puede usar un dado** en cualquiera de sus dos Operaciones hasta el final de la escena en curso, estando obligado a compensar la cantidad restante con sus recursos de CPU extra o de Memoria siempre que sea posible. Si no, actúa con los dados de Operaciones que tenga disponibles.



Función aleatoria (random) o Bucle trampa

Mientras se reconfigura su código los protocolos activan sus funciones de supervivencia para buscar soluciones creativas a la situación. Esto conduce al programa a optar por opciones que no haya contemplado antes. Su efecto es

que el programa hace algo totalmente imprevisible, inesperado y no necesariamente lógico. Algo que normalmente no haría en condiciones normales. Por ejemplo, ponerse a dar giros sobre sí mismo o recitar un poema épico durante una confrontación... Pero el que opte por reacciones inesperadas **no significa que actúe de forma caótica**. El caos en el comportamiento sólo se produce cuando ya no existen casillas disponibles.

Con frecuencia esta función provoca que el programa entre en un **bucle trampa**, repitiendo una y otra vez una misma acción que lo deja atrapado. Cuando se produce es incapaz de hacer nada más, pero la acción encadenada puede tener consecuencias (cuanto más creativo se pueda ser al respecto mejor). Como es común, la función aleatoria durará hasta el fin de la escena en curso.

¿Y SI SE TERMINAN LAS CASILLAS?

Al llegar a su límite de casillas disponibles el programa **comienza a actuar de forma errática, confusa y anárquica** con una muy alta predisposición a quedar atrapado dentro de un **Bucle trampa** que le obliga a realizar la misma acción ininterrumpidamente (al menos un tercio de probabilidades, 1-2 en 1D6). Los fallos son tan graves que ha perdido el control, por lo que en este punto puede ocurrir cualquier cosa. Las reacciones aleatorias se suceden una tras otra hasta que sin que pueda evitarlo al final de la escena sufre un error, se detiene y acto seguido se cierra. Su comportamiento puede afectar a todos los que estén en sus inmediaciones, ya sean enemigos o aliados. Los efectos del reinicio ya se han explicado. El programa arranca desde su copia original ya que ésta no se pierde, pero siempre sufriendo las consecuencias que se indican en las reglas.

EJEMPLO

Con la intención de huir de sus perseguidores Andrómeda realiza con éxito un gran salto entre dos edificios. Al hacer su tirada la reserva de la memoria es la que domina por lo que tiene que recurrir a sus Protocolos de respuesta para hacer frente al error producido al forzar sus capacidades.

Tanto el jugador que lleva al personaje como todos los demás miembros en la mesa coinciden en que la mejor respuesta de sus Protocolos es la **Función de Salida** que la fuerza a huir. No sólo encaja con las acciones que está realizando en ese momento sino

que en los detalles sobre su estilo y vulnerabilidades se indica que tiene predisposición a poner distancia cuando se encuentra en dificultades. El jugador marca la casilla y continúa la persecución. Como es de Nivel 4 aún puede marcar 3 casillas más antes de sufrir un error grave.

BÚFER

Todos los programas disponen de una memoria intermedia de acceso rápido denominada Búfer que les permite mantener una cierta independencia de la memoria del Sistema. El tamaño del Búfer aumenta notablemente su rendimiento y es independiente, por lo que su gestión depende del programa. Para calcular el espacio se usa el mismo sistema que el empleado para averiguar las casillas de estrés. Un programa dispone de tanto espacio de Búfer como el valor de **la mitad de su nivel redondeado hacia arriba**. En otras palabras, dispone de un espacio (o casilla) por cada dos niveles.

Nivel 1-2	1 casilla
Nivel 3-4	2 casillas
Nivel 5-6	3 casillas
Nivel 7-8	4 casillas
Nivel 9-10	5 casillas

El Búfer tiene varias aplicaciones. Con él es posible mejorar el rendimiento del programa pero además también le permite poder activar los Complementos (Plugins) a los que tenga acceso. Los Complementos son pequeños programas independientes que pueden incorporar mejoras. Su funcionamiento se describe en el apartado "Extras" más adelante dentro de este mismo capítulo.

EL BÚFER Y LA CONFIANZA

Si deseas equiparar la función del Búfer con un personaje del mundo físico puedes considerarlo como el grado de confianza que posee. En los momentos desesperados puede recurrir a ella para tratar de poner la fortuna a su favor.

Mejora del rendimiento

El Búfer permite almacenar información que sea útil para poder realizar las operaciones con más eficacia. Y lo hace en un lugar al que pueda tener un rápido acceso para así poder ahorrar tiempo. Por otra parte, todos los programas de Clase VI y superiores tienen la capacidad de aprender de sus errores almacenando las variables que han intervenido en su ejecución. Con

esta información serán capaces de extraer conocimientos muy valiosos para más adelante poder realizar sus tareas con más eficiencia. Esta capacidad de aumentar su rendimiento y poder aprender se recoge en el juego de varias formas.

Una es el "Módulo de aprendizaje". Este módulo complementa al Búfer y se detalla un poco más adelante. La otra forma es mediante una mecánica denominada "**Proeza**". Las Proezas son acciones que realiza con un alto grado de eficacia del que el programa puede extraer información útil. Siempre se almacenan en el Búfer. Cuando está lleno no es posible almacenar más hasta que se vacíe (o expanda su capacidad).

En términos de juego una Proeza es un resultado en los dados. Se obtiene una Proeza cuando al lanzar cualquiera de las dos reservas de **los dados de Operaciones**:

- De usar dados D6 se obtienen **dos o más** valores con un "6" en los resultados.
- De usar dados de categoría superior se obtienen **dos o más resultados iguales o mayores que "7"**. Estos se pueden combinar con los resultados del dado D6.

Y además, esto es muy importante:

- Esa reserva es la que **domina**.

¡RECUERDA! Sólo cuentan los resultados de la reserva de Operaciones. Las reservas de CPU y Memoria no cuentan para obtener Proezas.

Uso de las Proezas

Cada vez que al hacer una tirada se obtiene una Proeza **el jugador puede decidir usarla inmediatamente o almacenarla en el Búfer**. El uso de una Proeza **siempre garantiza un éxito** extra en la tirada. Por lo que si se gasta inmediatamente es posible añadir otro éxito al resultado final.

Una Proeza siempre garantiza **un éxito extra** en la prueba.

Pero si el jugador decide almacenarla queda guardada para otra ocasión más propicia. Para anotarla sólo tiene que poder una marca a lápiz en las casillas que están disponibles en la ficha. Cuando llegue el momento, el jugador puede decidir gastar un **Hack** y descargar el Búfer entero. Si lo hace **puede añadir a su tirada tantos éxitos como Proezas haya acumulado en él.**

PROEZAS

Se obtiene una proeza si al lanzar cualquiera de las dos reservas de los dados de **Operaciones:**

- De usar dados D6 se obtienen **dos o más "6"** en los resultados.
- De usar dados de categoría superior se obtienen dos o más resultados iguales o mayores que **"7"**. Estos se pueden combinar con los resultados del dado D6.
- Y además... esa reserva es **la que domina.**

Cada vez que se utiliza queda vacío por lo que si se desea volver a recurrir a él habrá que volver a llenarlo con las Proezas que se vayan obteniendo a lo largo de la partida. Por otra parte es posible recurrir al Búfer en cualquier momento, no es necesario esperar a que alcance su límite para poder usarlo. **Se vacía también cada vez que el programa sufra un bloqueo, quede congelado o se cierre y deba reiniciar.**

Espacio para Complementos

El espacio disponible en el Búfer, es decir el número de casillas disponibles, indica la cantidad de Complementos o Plugins que puede tener activos un programa en cualquier momento. Las casillas que estén ocupadas con Proezas no afecta a los Complementos activos. Su uso sólo depende de su tamaño, no de los espacios que estén libres.

Como una acción, un programa puede intercambiar sus Complementos activos conectando y desconectando los que quiera en el caso de que posea más de lo que sus casillas disponibles le permiten usar al mismo tiempo. El jugador debe por lo tanto declarar cuáles tiene activos durante la escena y anunciar los cambios que quiera realizar sobre ellos antes de anunciar sus demás acciones.

OTRAS FUNCIONES

Las demás secciones que aparecen en la ficha son módulos con funciones que añaden información importante sobre los programas.

EXTRAS

Un **"extra"** es un trozo de código o un pequeño programa, por lo general bastante simple, con una función muy determinada. Estos códigos pueden complementarlo o aportarle algo. En algunos casos pueden permanecer flotantes o incluso integrarse al propio código del personaje, aunque con ciertas limitaciones.

Las características de los extras disponibles depende —para variar— de la ambientación. El personaje puede interactuar con estos códigos para usarlos en su beneficio. Muchos han sido creados con ese propósito. Pueden ser objetos que posee un personaje, que puede usar o bien complementos que le ayuden a mejorar su rendimiento.

En Scroll los **extras** se organizan en tres grupos:

- **Elementos.** Código de objetos importantes.
- **Complementos o "Plugins".** Códigos que se “conectan” al programa.
- **"Ítems".** Código de objetos de uso limitado.

Muchos extras forman parte de la estructura de un sistema. En un entorno de juego por ejemplo la mayoría de los elementos y de los ítems son componentes del mundo de juego, llegando a convertirse en objetos icónicos en algunos casos. Salvo algunas excepciones es posible llevarse un elemento más allá del área de influencia del sistema o subsistema al que pertenecen. En otras palabras, pueden existir en cualquier parte, no solamente dentro del ámbito donde funcionan sus leyes, por lo que el personaje puede llevárselos con él. No obstante algunos pueden quedar anulados, llegando incluso a desaparecer si se abandonan sus límites. Los Ítems por el contrario suelen funcionar sólo dentro del subsistema para el que fueron creados. Muchos

dejan de funcionar o desaparecen si abandonan el área de influencia de su subsistema.

Los complementos son una excepción. Por lo general funcionan en cualquier parte. En ese sentido se trata de un tipo de extras que suelen ser independientes del Sistema para el que hayan sido creados por lo que son muy versátiles, aunque también tienen sus desventajas.

Elementos (objetos importantes)

Los Elementos son pequeños trozos de código que en algunos casos pueden formar programas completos, aunque no llegan a ser tan sofisticados como el código de un personaje. En un sistema suelen ser objetos importantes o valiosos que aportan algún beneficio a quien los utiliza, como un vehículo o un arma por ejemplo. Otra de sus ventajas es que no es necesario combinar su código con el de su dueño como sucede con los Complementos. Se pueden emplear tantas veces como se quiera por lo que no están limitados a un número de usos. El personaje puede usarlo cuando lo necesite siempre que tenga acceso a él. Tampoco necesita de espacio en el Búfer para poder funcionar.

En el caso de usar las reglas de Inventario, muchos ocupan espacio en forma de un número de casillas. Si es así hay que especificar cuantas por cada uno. No obstante algunos Elementos no pueden ser almacenados por lo que si el programa quiere recurrir a él tendrá que conocer su ubicación dentro del Sistema. Por regla general los Elementos no se pueden apilar para ocupar menos espacio; con muchos Ítems sí que es posible hacerlo.

Los elementos normalmente tienen una **función** y una **potencia**.

- La **función** especifica qué hace o para qué sirve. Permiten realizar una actividad de una forma más óptima o bien alguna que no sería posible hacer si no se contara con la ayuda del Elemento. Un código que describa una moto sirve para desplazarse por el espacio abstracto de

un Sistema más rápido, unas alas para volar por él, un disco para guardar datos y luchar, una pistola o una espada para hacer daño, la ropa para cambiar el aspecto del Avatar y provocar la admiración de otros programas..., etc.

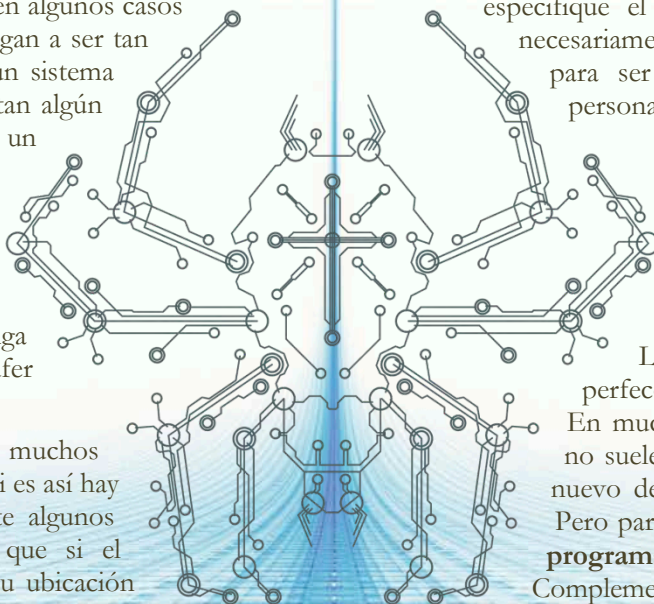
- La **potencia** indica su eficacia y grado de poder. Especifica la ventaja que obtiene quien lo usa. Se traduce en una cantidad de éxitos que se añaden automáticamente a la tirada. Puede ser +1, +2, +3..., lo que especifique el nivel de potencia del Elemento. No obstante no necesariamente tiene que contar con un bonificador de potencia para ser considerado como tal. Lo importante es que el personaje pueda hacer un uso específico de éste.

En ocasiones esa potencia puede volverse en contra del personaje. Por ejemplo, si utiliza una moto a gran velocidad y se estrella, el valor de la potencia que llevara activa en ese momento se añade al daño recibido.

Complementos o Plugins

Los Complementos son trozos de código que perfeccionan a un programa otorgándole nuevas capacidades. En muchos casos simulan el efecto de algunas **Utilidades** y no suelen tener un límite de usos. Le permiten aprender algo nuevo de inmediato sin necesidad de reprogramar su código. Pero para funcionar necesitan combinar **su código con el del programa que lo esté usando**. Esa conexión es parcial, el Complemento se conecta al programa pero el código continúa siendo independiente quedando como un módulo extra. Esto permite intercambiar los Complementos en cualquier momento como una acción del personaje. Hay algunos no obstante con unas especificaciones muy especiales. En ocasiones requieren de un manejo especializado si se desea alterar la conexión.

Un Complemento **siempre requiere espacios o casillas del Búfer para poder estar activo**. El valor del Búfer sirve por lo tanto para saber cuántos se pueden usar al mismo tiempo. Aunque lo normal es que sólo precisen de un



único espacio hay algunos que pueden necesitar de varios más, lo que restringe la cantidad que es posible tener activos al mismo tiempo. Cada Complemento específica cuántas casillas necesita (y si no lo hace debería hacerlo). Es conveniente anotar también si es posible combinarlo con otros pues existen algunos muy exigentes. Éstos requieren un uso exclusivo por lo que se niegan a funcionar si hay otro conectado ya al programa. En la mayoría de los casos estas restricciones se deben a incompatibilidades, pero hay otros en los que existen directrices especiales por necesidades de diseño. Es bastante frecuente que esta información no se encuentre disponible por lo que no son raros los fallos y accidentes cuando un programa accede a un Complemento de origen desconocido.

Estén o no conectados al programa los Complementos **permanecen flotantes al código por lo que jamás ocupan espacio de almacenamiento**. Así, no es necesario guardarlos en un inventario. En caso de tener que desconectar alguno el personaje siempre los lleva consigo por lo que sólo has de llevar una lista de los que tiene. También es posible abandonarlo o entregárselo a otro.

Cada Complemento debe especificar su **función**, cuántos **espacios de Búfer necesita** y sus **incompatibilidades** en caso de tener alguna; algo muy importante para evitar problemas. En la ficha de personaje hay reservado un espacio para anotarlos. Cada espacio tiene una casilla en la que se puede poner una marca para saber si está en funcionamiento.

Ítems u objetos simples

Los Ítems son trozos muy simples de código que sirven para definir objetos sencillos que es posible encontrar en un sistema. Suele tratarse de la descripción de algo muy específico. Su diferencia con los elementos es que poseen una función muy concreta, por lo general de un sólo uso o de uno muy limitado. En la mayoría de los casos sirven como accesorios, piezas útiles que aportan algún beneficio o como los componentes de otros elementos.

Es posible usar tantos como se desee. En caso de poseer un inventario se pueden guardar en él y en muchos casos es posible apilarlos para ahorrar espacio. Por otra parte y al igual que con los Elementos, no requieren de casillas del Búfer para poder funcionar. Los Ítems que más valor tienen para

un personaje son los que pueden beneficiarle como por ejemplo la munición de un arma, células de energía, una llave, un botiquín médico, comida, ropa, etc.

Es fácil confundir a los Ítems con los Elementos. En realidad eso no importa. No se trata más que de una manera de catalogar a los Extras. No obstante es bastante fácil averiguar si algo es un Elemento o un Ítem porque los segundos carecen de potencia. Sólo se definen por la función para la que fueron diseñados y que describe su utilidad. Una copa sirve para beber por ejemplo, eso es todo. La ropa en un mundo virtual podría tratarse de un Elemento si concediera ventajas al que la usara, como un traje especial o una armadura. En estos casos sí que podría catalogarse como un Elemento, de otro modo se puede considerar un Ítem.

En el caso de utilizar un inventario los ítems ocupan espacio por lo que cada uno debe especificar el número de casillas que necesita y si se pueden apilar o no. Es el caso de las provisiones, el dinero o la munición por ejemplo. Ítems apilables que ocupan un mismo espacio y que pueden reunir en un conjunto de 5, 10, 20 objetos o incluso 50 o más.

Dentro de la familia de los Ítems existe un tipo muy especial que por sus particularidades es mejor describirlo por separado. Se trata del **Token**.

Tokens

El Token es un subtipo de Ítem con unas características algo especiales. Para empezar muchos no se pueden almacenar o acumular por lo que no es posible reservar su uso para otra ocasión. En el caso de que se puedan almacenar, es posible guardarlos sin ocupar espacio dentro de un inventario en el caso de usar uno. Se consideran pues "flotantes" al personaje.

Suelen tener un uso muy limitado, reducido al efecto que se produce al interactuar con él. Puede tratarse de energía o poderes extra temporales, tiempo, monedas de puntuación, medallas, etc. El Token también tiene siempre **una función** que especifica sus efectos. Algunos Tokens pueden tener incluso un tiempo límite para ser recogidos. En términos de mecánicas del juego ese tiempo se describe como el número de veces que es posible hacer una prueba para intentar cogerlo.

INVENTARIO (OPCIONAL)

Algunos escenarios se prestan a que el personaje posea un inventario y deba gestionarlo. En muchos subsistemas de juego y en la mayoría de los mundos virtuales el inventario limita la cantidad de objetos que puede transportar un personaje creando al mismo tiempo un nuevo tipo de minijuego donde la

gestión de los recursos es fundamental.

El módulo de Inventario es opcional en Scroll, como muchos otros bloques de reglas, su función es complementar algunas ambientaciones. Por eso se detallan aquí unas reglas muy sencillas para poder implementarlo. Este inventario es por lo tanto muy genérico y puede usarse en casi todos los estilos de juego.

Un inventario consiste en un número de "espacios", también llamados: "Slots". La

cantidad de espacios se describe mediante el producto de dos números (o matriz) que puede ser de: 3 x 3; 3 x 4; 4 x 6; 4 x 8; 5 x 8, etc. La cantidad de espacios define el espacio total del inventario. Se puede representar gráficamente mediante una malla con ese número de espacios en forma de casillas o bien simplemente dejarlo anotado como una cantidad de espacios. En el segundo caso bastaría entonces con especificar el número de espacios por lo que no haría falta recurrir a una malla, pero usarla permite distribuir los objetos **sin tener que estar haciendo sumas o restas**. Es muy sencillo dibujar una con la ayuda del papel cuadriculado de un bloc de notas. La malla se pinta con un bolígrafo y los objetos se muestran tan detallados como se quiera con un lápiz para poder borrar o modificar las anotaciones.



Si se desea almacenar objetos, muchos **Elementos** y la mayoría de los **Ítems** ocupan espacios. Los **Complementos** no lo necesitan y los **Tokens** no pueden ser almacenados. Cada Elemento o Ítem debería especificar el número de espacios que ocupa. Una espada o un rifle por ejemplo pueden ocupar dos y un botiquín uno. Hay objetos que son **apilables**, por lo que pueden ocupar el mismo espacio varios de ellos. Normalmente una cantidad fija como 5, 10, 20 ó 50. Cuando se han ocupado todos los espacios no es posible almacenar más objetos.



Se puede comenzar con un inventario pequeño, como uno de 4 x 3 ó de 5 x 4 por ejemplo, y expandirlo durante las aventuras. Dependiendo del escenario es posible también ampliar el número de inventarios totales consiguiendo Elementos como bolsas, compartimentos, cajas, mochilas, sacos, etc. Cada uno con su propia matriz ampliable. Una cantidad razonable de inventarios es de dos a cinco. Hay Complementos que podrían incluso conceder espacio extra; existen muchas posibilidades para aplicar estas reglas.

EJEMPLO

Para el escenario donde se juega con Andrómeda se ha establecido un inventario base de 5 x 4 casillas. Para un total de 20 espacios de inventario.

En un momento dado Andrómeda tiene ocupados varios espacios con un rifle de asalto, 6 casillas; un ordenador portátil, 4 casillas; unas llaves, 1 casilla; unas tenazas multiusos, 1 casilla; una pequeña lata de combustible para mecheros, 1 casilla; una navaja multiusos, 1 casilla y una pistola pequeña, 1 casilla. Aún dispone de 5 casillas libres.



APRENDIZAJE

"El aprendizaje es cualquier cambio que haga un sistema para adaptarse a su medio ambiente."

Herbert Simon

Los programas de Clase igual o superior a VI que dispongan de algoritmos de aprendizaje pueden aprender de sus errores. No todos los programas cuentan con ese privilegio y de tenerlo suele estar bastante limitado por diferentes motivos. No obstante, la visita de La Libélula concede al programa nuevas funciones que le permiten procesar toda la información útil para aprender y poder así optimizarse.



Cuando un programa realiza una acción de forma eficaz o comete un grave error, reúne todas las variables que hayan intervenido, las memoriza y las examina. De ellas puede extraer información muy valiosa realizando evaluaciones de la situación para así poder aumentar su rendimiento. Ese

proceso de aprendizaje se puede registrar en la ficha del personaje como un reflejo de su evolución y se denominan: **estimaciones**.

Una estimación es la descripción mediante una frase o un breve párrafo de alguna experiencia importante que haya tenido lugar durante la sesión de juego. El jugador puede hacerlo cuando quiera, siempre una por sesión.

Durante la partida o al final de la misma decide que alguna situación ha sido tan relevante que su programa la puede aprovechar. Ésta queda grabada como una lección que ha aprendido o como un obstáculo que tras superarlo le ha permitido obtener valiosos datos para el futuro.

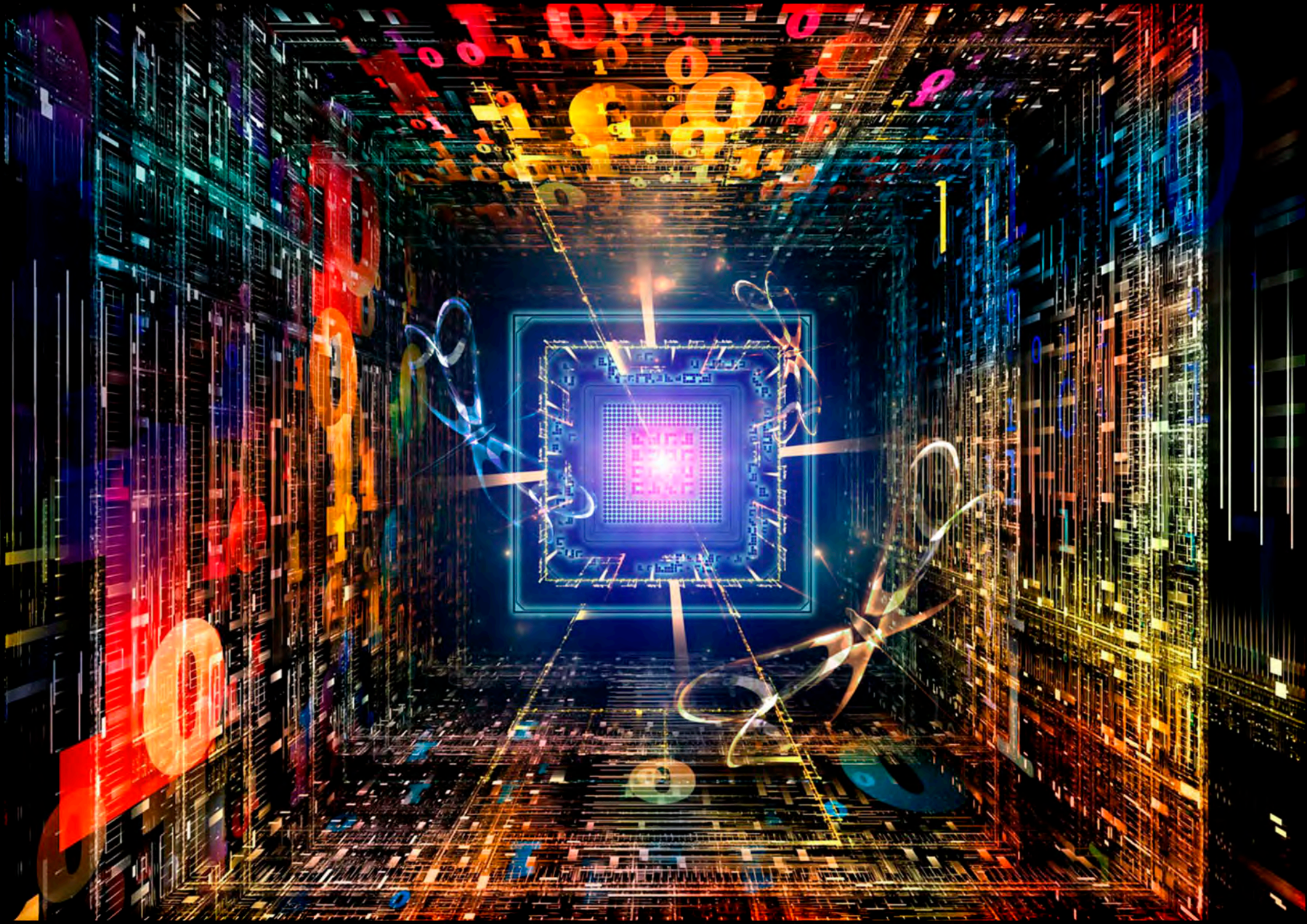
El programa puede usar sus estimaciones más adelante de dos maneras. Una para permitirle aprovechar ese conocimiento a su favor si se vuelve a encontrar en una situación parecida, y la otra para reflejar su propia evolución como programa.

Las estimaciones se registran en la ficha en el espacio reservado. Siempre tienen una casilla que permite indicar cuándo se recurre a ella durante la partida.

☐ _____

Su funcionamiento se explica más adelante en el capítulo dedicado a la evolución de los programas.

FIN DE LÍNEA 



Scroll

JUEGO DE ROL EN EL UNIVERSO DIGITAL

EJECUCIÓN ANÁLISIS Y RESOLUCIÓN DE PROCESOS

FUNCIÓN 4

"RESOLUCIÓN DE PROCESOS"

"Wintermute era el cerebro de la colmena, el que tomaba las decisiones, el que hacía cambios en el mundo exterior. El Neuromante era la personalidad. El Neuromante era la inmortalidad."

"Neuromante". William Gibson.



En Scroll las tiradas son una herramienta para resolver los conflictos. Añaden un componente de aleatoriedad a los resultados con el fin de provocar una sensación de incertidumbre. Un reflejo de los cientos de factores que intervienen a la hora de llevar a cabo una acción. Las probabilidades de tener éxito a menudo tienen en cuenta el grado de experiencia y las habilidades de un personaje. A esto se añaden otros factores, como los del entorno. Pero como una herramienta deben ser usados cuándo y dónde sean necesarios. Como ya se ha explicado, haz tiradas cuando las consecuencias de los resultados permitan que la trama tome distintas direcciones. Es decir, cuando la resolución del éxito o del fracaso siempre tenga consecuencias. Si no vale la pena recurrir a las tiradas, no las hagas.

Aunque esto se explica de nuevo más adelante, la narración de las acciones durante el transcurso del juego utiliza un recurso fundamentado en **la escena**. Pero, ¿qué es una escena? Es más sencillo de lo que parece pues debido a la familiaridad con el lenguaje cinematográfico su concepto resulta algo natural para la mayoría. Una escena no es más que un momento concreto del tiempo en una narración donde suceden unos hechos; ese momento se produce normalmente en un único lugar. Cuando el tiempo avanza y los hechos descritos transcurren en otro momento y lugar podemos decir que se ha cambiado de escena. Por ejemplo, la entrada a una bóveda para robar un

objeto, una carrera de motos o un combate en un almacén son escenas. Escapar de la bóveda, declararse vencedor de la carrera o terminar el combate abandonando todos estos lugares para ir a otro sitio son cambios de escena que conducen a otras diferentes.

Las escenas deben estar dotadas de significado y conectadas entre sí por alguna razón que permita establecer relaciones entre unas y otras. La sucesión de escenas unidas por el hilo conductor que crea esas relaciones de significado, es decir la trama, componen un relato.



La tirada genérica en Scroll, o su enfoque por defecto, no pretende resolver cada acción que realiza un personaje sino más bien solucionar los conflictos que surgen durante el transcurso de una escena en conjunto. En otras palabras, la escena no se fragmenta en pequeñas

unidades de tiempo, como si fuesen los fotogramas de una película asalto por asalto, sino que se contempla como un todo abstracto que se resuelve, tomando la acción una dirección u otra dependiendo de los resultados de una o de unas pocas tiradas. Este enfoque debería ser la regla general para resolución de la mayoría de las acciones en el juego.

No obstante, dada la cantidad de escenarios posibles que es posible usar en Scroll, habrá un grupo de jugadores a los que les guste detenerse un poco sobre las escenas de acción. Algunas reglas como la posibilidad de afectar a la integridad del código están disponibles para poder explotar esa posibilidad. Al usarlas puede resultar emocionante —y divertido— fragmentar un poco la escena, haciendo un acercamiento o "zoom" sobre algunos detalles. Por eso en Scroll hay **dos enfoques o modos de resolver los conflictos** al jugar las escenas.

RESOLVIENDO LOS CONFLICTOS

Hay dos formas de resolver el desarrollo de las acciones en una escena. Cuando llegue el momento de resolver un conflicto haciendo tiradas lo primero que hay que decidir es con qué enfoque lo resolveremos. Recuerda que siempre es posible alternar entre un modo u otro dependiendo de cómo sea la escena o de cómo se quiera afrontar el conflicto que se plantea. Los dos modos de solucionar las acciones son los siguientes:

A. ENFOQUE o MODO ABSOLUTO. Acciones conjuntas. La adversidad es un conjunto de factores. El o los conflictos se resuelven a la vez con una sola tirada, o unas pocas.

*El enfoque absoluto **no usa las reglas de Integridad** del código.

B. ENFOQUE o MODO RELATIVO. Acciones divididas. Los elementos que componen el conflicto se resuelven por separado dividiéndolo en conjuntos de acciones que se van solucionando uno tras otro siguiendo un orden de asaltos.

*Es conveniente utilizar las reglas de Integridad del código con este enfoque.



A. ENFOQUE ABSOLUTO

En el enfoque absoluto la adversidad es un conjunto de factores que se oponen a los objetivos de los personajes. Resolverlas en una escena implica entenderla como un todo que se resuelve como una serie de **acciones conjuntas**, sin atender a ningún detalle en concreto. Esto permite que el flujo de la narración no se vea interrumpido por la necesidad de tener que detenerse en los detalles, así el ritmo de la narración resulta más fluido. Las reglas para solucionar los conflictos de esta forma son un recurso abstracto que ayuda a determinar qué dirección sigue el curso de los acontecimientos por lo que es perfecto para un estilo de juego muy narrativo donde es importante contar con una forma rápida y sencilla de resolver las situaciones. Esta opción es la más adecuada cuando no se utiliza el módulo de reglas de Integridad del código. Aún es posible usarlo, pero si deseas mantener el juego lo más sencillo posible no lo tengas en cuenta. Este modo puede usarse con cualquier tipo de prueba para superar un conflicto.

En un combate por ejemplo no importa cómo ni cuándo golpeas a un oponente por lo que no se sigue un orden de rondas. Las mecánicas sirven para averiguar si el resultado ha sido favorable para uno u otro bando, eso es todo. Aunque es posible realizar varias tiradas si es necesario, se hacen sin recurrir a una mecánica de turnos y asaltos. En el caso de una carrera de motos por ejemplo, la tirada resuelve el curso completo del desafío permitiendo saber si el personaje ha obtenido o no la victoria.

No emplees las reglas de Integridad del código si prefieres el enfoque absoluto. Si lo haces surgirán problemas a la hora de resolver los daños que pueda sufrir el o los personajes. En el caso de utilizarlo es posible usar este enfoque en otras pruebas que no impliquen un enfrentamiento directo, siendo posible entonces combinar ambos métodos.

Los conflictos en el enfoque absoluto

En el entorno donde se encuentran los personajes, ya sea el Sistema para los programas o el mundo físico para los usuarios, surgen conflictos y para resolverlos es necesario superar una serie de desafíos en forma de amenazas u

obstáculos. **Para describir el nivel de tensión de esos desafíos el director posee su reserva de dados**, siempre de un mismo color. **A todos los desafíos les asigna una cantidad** que indica su **nivel de dificultad**. Equilibrar el nivel de tensión o la fuerza de las amenazas siempre depende de elegir la cantidad de dados más apropiada. Por ejemplo, cuando un programa desea traspasar un muro de seguridad un solo dado es una dificultad fácil, un conflicto con un nivel bajo de tensión, mientras que traspasar el de una importante base de datos puede suponer una dificultad de hasta diez, doce y quince dados, un obstáculo muy difícil.

Si un personaje se enfrenta a varias amenazas u obstáculos al mismo tiempo **se toma el nivel del obstáculo más difícil o el de la amenaza más poderosa** (su líder por ejemplo) **y se añade otro dado por cada amenaza** adicional. Si los oponentes adicionales son tan poderosos como su líder se añade dos dados en vez de uno.

Esta es una regla genérica que puedes usar siempre para describir la dificultad de los desafíos, aunque tienes bastante libertad en este aspecto y es posible optar por algunas variantes. Por ejemplo, si las amenazas cooperan bien entre ellas se pueden sumar sus niveles individuales. Si en cambio se coordinan muy mal se puede aplicar la regla general que se ha descrito. Un poco más adelante existe una tabla para establecer los niveles en función de su dificultad.

Los dados de la reserva del Director se podrían separar, representando distintos grupos de amenazas tal y como se hace en el enfoque relativo. Pero en el enfoque absoluto lo conveniente es mantenerlos agrupados ya que plantea el enfrentamiento como un conjunto que describe a **todas** las amenazas presentes, es decir, la **dificultad absoluta** de todo el conflicto. Además, ten en cuenta que en este enfoque (y sólo en este) separarlos en varias reservas puede disparar la dificultad del enfrentamiento.

Resolución

Los dados del Director del juego personifican el riesgo que supone enfrentarse a la adversidad y las consecuencias negativas que supondría para los personajes perder el conflicto. La diferencia de éxitos obtenidos determinará qué bando es el ganador, la cantidad obtenida en qué medida se ha impuesto un bando

sobre otro. La victoria significa que los acontecimientos se ponen a su favor para obtener sus metas y el fracaso lo contrario.

En una confrontación los dados que obtienen éxitos eliminan a los que no los han obtenido, simbolizando a los enemigos que han sido derrotados. Si aún quedan enemigos es posible que haya que continuar el enfrentamiento hasta que toda la oposición haya sido derrotada, huya o se rinda, representada por los dados restantes. Continuar en realidad es una elección tanto del grupo de juego como de su Director, que pueden querer reflejar mediante unas pocas pruebas el transcurso de una batalla y el grado de implicación de cada bando. Algunos enfrentamientos se pueden resolver mediante una sola tirada determinando los resultados y en otros continuar hasta que uno de los bandos se quede sin dados. Pero como ya he dicho, se hace sin recurrir a una sucesión de rondas o asaltos consecutivos.

Pero al margen de las consecuencias de una derrota o incluso ante una victoria existe otro riesgo. Este puede convertir el fracaso en algo aún mucho peor o amargar algo la victoria. Cuando en la tirada domina la reserva del Director significa que el Sistema ha detectado una **Anomalía**. El efecto en el juego es que el Director gana un punto de Anomalía que podrá añadir a su reserva de puntos. Puede tomar una ficha o bien poner una cualquiera en el cuenco reservado para administrarlas. Las Anomalías se explican más adelante en este mismo capítulo.

Colaborando

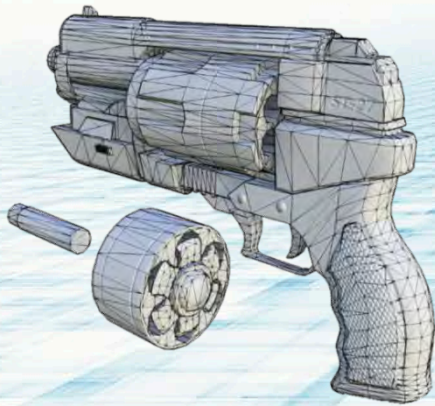
En el enfoque absoluto el trabajo en equipo para resolver las pruebas no tiene un gran efecto. Comparado con el enfoque relativo, que se detalla en el siguiente apartado, su estructura para resolver la acción dificulta poder trazar estrategias complicadas o emprender acciones tácticas.

En este enfoque, **para que un grupo pueda colaborar siempre hay que elegir a un líder**. Este será el protagonista principal de la escena mientras que el resto de los jugadores le servirán de apoyo. Todos siguen actuando como grupo pero la tirada del protagonista será la decisiva. Pero asumir el papel de líder siempre implica un riesgo, como veremos. Todos los personajes de apoyo lanzan sus dados, pero **sólo los de la Operación que entre en juego**, Potencia o Control. El resto de las reservas se ignora. Todos los éxitos que

obtengan se añaden a la reserva total de su líder, que lanza todos sus dados normalmente. Tampoco se tiene en cuenta las fuerzas de la tirada de los personajes de apoyo, es decir, si dominan o no en la tirada. A todos los efectos sus tiradas sólo cuentan para determinar los éxitos que le pueden aportar a su líder. Por otra parte, tampoco pueden usar Rutinas o Utilidades, beneficiarse de Complementos, ventajas, etc.

Para determinar quién es el líder se puede extraer de la situación planteada en la escena. Pero si no está claro es posible elegirlo directamente o bien optar por realizar tiradas para averiguarlo. Para hacerlo todos los jugadores lanzan los dados de todas sus reservas. El que obtiene el mayor número de éxitos es el líder (o el protagonista) en la escena. Todos los dados de su tirada cuentan en la prueba mientras que el resto actuará como su apoyo contribuyendo solamente con sus dados de Operaciones.

Las consecuencias de la tirada afectan a todo el grupo y no solamente en lo que se refiere a las repercusiones negativas de haber fracasado la prueba. **Todos los que hayan participado en la acción sufren los efectos de un resultado en el cual domine la reserva del Tiempo de proceso o el de la Memoria.** Si se da esta situación se sigue el procedimiento añadiendo una marca en sus Protocolos de respuesta o aumentando su Tiempo de proceso, lo que corresponda.



La derrota

La derrota se produce cuando se obtienen menos éxitos que el oponente (normalmente la tirada del Director). Esto implica que el curso de los acontecimientos dificulta que puedan cumplirse los objetivos de los personajes o del protagonista, mientras que favorece a los del bando ganador. Pero la derrota tiene otro efecto sobre **el personaje que ha liderado la acción, que tendrá que asumir directamente las consecuencias del fracaso.**

Es importante que tengas en cuenta que esto **no es obligatorio en todos los casos.** Aunque lo normal es que haya que sufrir las consecuencias algunas veces es posible evitarlo si se ingenia alguna causa que resulte coherente con los hechos. Debería de ser una opción, pero sólo de vez en cuando.

Siempre que en el contexto de la escena una derrota suponga sufrir algún perjuicio que afecte a la integridad del programa **el jugador que ha actuado de líder tiene tres opciones:**

1. Compensar los daños aumentando en **1 punto** su Tiempo de CPU. Siempre se incrementa un solo dado independientemente de la cantidad de daño recibido, es decir, del número de éxitos que hayan sobrepasado a su tirada. Es posible recurrir a esta opción aunque ya se hubiese hecho un incremento de la velocidad antes de hacer la tirada. Esta opción **no siempre está disponible.**
2. Permitir que el Director elija **una de las casillas de sus Protocolos de respuesta** sobre la que tendrá que poner una marca. Se señala sólo una, independientemente del daño que haya recibido. El Director puede elegir la que quiera y no necesariamente una que encaje con la descripción del personaje. Esta opción **no siempre está disponible.**
3. Señalar tantas **casillas de corrupción** del código como el número de éxitos que hayan sobrepasado a su tirada, que es la forma de contabilizar el daño en el juego. Esto anula de forma **temporal** los dados de Operaciones que haya elegido para poner las marcas. Puede elegir los que quiera pero ateniéndose a sus reglas específicas de comenzar siempre por la izquierda e ir llenando los grupos antes de poder pasar a los

siguientes de la derecha. Esta opción siempre está disponible. Normalmente se recurre a ella porque ya no es posible optar por ninguna de las dos opciones anteriores.

Cualquiera de estas opciones puede provocar que el programa sufra una detención, quede inutilizado o se produzca un error que le obligue a cerrarse. Pero existe una norma a recordar. Con la importante excepción de siempre tener la opción de marcar sus casillas de corrupción, **no es posible aplicar dos veces un mismo efecto que ya haya tenido lugar como una consecuencia** de la acción. Sí que es posible en cambio aplicarlo si el efecto se ha elegido antes de hacer la acción, por ejemplo al haber decidido incrementar a voluntad el Tiempo de CPU. El Director debe supervisar la elección que haga el jugador.

Por lo tanto, si el personaje ha fracasado y además ha dominado la CPU en su tirada no puede elegir aumentarla de nuevo otro punto más ya que al haber dominado esa tirada está obligado a tener que aplicar ese efecto y no es posible aplicarlo dos veces. Si por otra parte la reserva que ha dominado ha sido la de la Memoria se produce el efecto opuesto. El jugador no podrá optar por sus Protocolos de respuesta ya que no es posible marcar dos veces una respuesta y aplicar sus efectos al mismo tiempo. En este caso el personaje estaría obligado a incrementar su CPU para poder compensar o recurrir a la corrupción anulando sus dados de Operaciones; a menudo la peor de las tres posibilidades. En ocasiones el jugador no podrá recurrir ni a su Tiempo de proceso ni a sus Protocolos para compensar por lo que no tendrá más remedio que optar por anular sus dados de Operaciones.

En resumen, si al producirse un fracaso además domina la Memoria es necesario compensar exigiendo más Tiempo de CPU o marcar Corrupción; y si quien ha dominado ha sido el Tiempo de CPU entonces es necesario optar por marcar Corrupción o recurrir a los Protocolos de respuesta, pudiendo al Director sugerir la casilla que crea conveniente.

Cuando ya no es posible optar por el Tiempo de CPU o los Protocolos de respuesta el jugador debe recurrir a sus Casillas de corrupción marcando tantos como la diferencia de éxitos en su contra.

B. ENFOQUE RELATIVO

El enfoque relativo consiste en dividir una escena fragmentando los hechos que intervienen en el conflicto en **acciones discretas** que se van resolviendo una por una. Las acciones se encadenan una tras otra hasta que se haya resuelto. El nivel de detalle de esa fragmentación puede ser tan fina como queramos. Pero este juego intenta mantenerse fiel a su enfoque inicial por lo que aún dividiendo la escena las acciones siguen siendo un conjunto de acontecimientos que se resuelven a la vez.

Esto se consigue mediante una mecánica muy común, la sucesión ordenada de **turnos** y **asaltos**. Los asaltos exigen ralentizar el flujo de la narración para centrarse en algunos aspectos de la situación. Esto tiene una gran influencia sobre el ritmo de la partida. Dependiendo de cómo se maneje sus efectos tendrán repercusiones sobre la experiencia de juego.

Dividir un conflicto en acciones es interesante cuando éstos son complejos y los jugadores desean detenerse sobre algunos detalles, como podría ser un combate; y no en todos sino en los más importantes. Durante su resolución pueden intervenir distintos hechos que contribuyen al éxito o el fracaso. En muchos casos el resultado de cada acción individual junto con la estrategia de todo el grupo es fundamental para poder tener éxito. Se trata de una cuestión de gustos. Cada uno tiene algo que ofrecer y usar lo mejor de ambos puede constituir una ventaja para el juego.

Emplea el enfoque relativo siempre con las reglas de **Integridad del código**. Permitirá solucionar los enfrentamientos directos mucho mejor siguiendo un orden de asaltos. Si prefieres no usarlo las consecuencias de las derrotas sufridas durante cada asalto no sólo serán difíciles de determinar sino también muy duras para el personaje.

Los conflictos en el enfoque relativo

En realidad no existe una gran diferencia a la hora de plantear los conflictos. Se aplican las mismas reglas que en el enfoque absoluto para determinar el nivel de los desafíos. Una cantidad de dados define siempre la tensión que supone un conflicto para los personajes. Los dados representan una

adversidad, sea cual sea, que los personajes deben superar para lograr sus propósitos.

Cada obstáculo o amenaza tiene un número de dados que representa su nivel de poder. Al estar orientado a entender la adversidad como una serie de factores discretos **en lugar de agruparlos en una sola reserva como se hace en el enfoque absoluto ahora es posible describir a cada uno por separado**. De esta forma es posible enfrentarse a tres amenazas de nivel 2 (dos dados cada una) y a dos de nivel 1 por ejemplo (un dado cada una). O bien a una sola amenaza que supone un nivel de dificultad de seis dados o a dos de nivel 3 con tres dados cada una.

Con este método dispones de una gran libertad para elegir el nivel de dificultad de las fuerzas que se oponen a los protagonistas, aunque siempre es conveniente prestar atención al equilibrio entre ambos bandos para poder controlar la dificultad del desafío.

Resolución

La gran diferencia del enfoque relativo con el absoluto es que para solucionar algunos conflictos es necesario realizar varias acciones para resolverlos. En muchos casos bastará una sola acción que requiera realizar una única prueba para resolver la escena, pero en otros será necesaria una sucesión ordenada de acciones organizadas por turnos mediante una estructura de asaltos. Existen además algunos tipos de pruebas que requieren hacer varias tiradas, como los Retos o los Desafíos de habilidad. Este tipo de pruebas se describen más adelante.

Pero, **¿qué sucede cuando domina la reserva del Director?** pues exactamente lo mismo que en el enfoque anterior. Éste **podrá contar con un punto de Anomalía** que podrá añadir a su reserva de puntos. Pero hay un detalle a tener en cuenta: **sean cuales sean las veces que domine su reserva de nuevo a lo largo de los asaltos consecutivos de una escena nunca podrá ganar más de uno en ella**. Por lo tanto, desde el momento que gana un punto ya no puede ganar más aunque su reserva consiga dominar varias veces. Sólo podrá volver a hacerlo cuando se termine la sucesión de asaltos, cambie la situación y comience una nueva escena. Cuando obtiene un punto

puede elegir una ficha —o puntero— destinada a señalar las Anomalías o bien poner una en el cuenco reservado para poder administrarlas.

Solucionar las acciones siguiendo un orden de asaltos implica tener claros algunos detalles que se explican a continuación.



Estructura del asalto

Cuando llega el momento de resolver un conflicto se detienen los acontecimientos y se organizan en una serie de rondas consecutivas que van describiendo las acciones de los personajes implicados en la escena. Cada jugador actúa durante su turno, momento en el que podrá realizar sus acciones. Esto supone que el tiempo del flujo de la narración va algo más despacio. Cada acción no tiene por qué ser algo concreto, como dar un golpe o un salto, sino que pueden describir algo más general. Por ejemplo, puede consistir en realizar una serie de maniobras con un vehículo para obtener una posición favorable durante una persecución o manipular un trozo de código para conseguir un efecto. Así, los asaltos se van sucediendo uno tras otro hasta que se resuelva el conflicto, momento en el que terminan. El orden suele ser el siguiente:

- Una vez el Director ha planteado la escena se inicia el primer asalto.

- Los jugadores, incluyendo el Director deciden sus acciones y las llevan a cabo. Para hacerlo es posible seguir un orden. En cada asalto todos deben intervenir siempre que quieran hacerlo claro está. Un jugador puede elegir no hacerlo. Nadie está obligado a intervenir si no lo desea.
- Si el conflicto no se ha solucionado termina el asalto y comienza el siguiente. Los asaltos se suceden uno tras otro cuanto sea necesario.
- Si se resuelve el conflicto termina la sucesión de asaltos pudiendo volver entonces al desarrollo normal de la narración.

La comparación de los resultados de las tiradas permite averiguar si las acciones tienen éxito o fracasan. Ten en cuenta que en cada intervención **los implicados en un enfrentamiento que interactúan entre ellos siempre hacen sus tiradas a la vez**, comparando todos los resultados obtenidos. No hace su prueba uno y después el otro cuando llega su turno. Por esta razón en muchos casos no es necesario seguir un orden determinado de iniciativa.

Pero hay casos en los que saber cuándo actúa cada uno podría conceder una ventaja táctica. Un oponente que actuase primero podría obtener ventaja al comienzo de un enfrentamiento por ejemplo. O bien, si un combatiente elimina antes a su contrincante podría ir a socorrer a un compañero en dificultades. Si el orden de los turnos es importante existe la iniciativa para establecerlo, que en este juego se ofrece como una sugerencia en el caso de que hiciese falta.

Iniciativa en el caso de necesitarla

De ser necesario, para averiguar el orden de los turnos existen varios métodos. El más simple con diferencia consiste en limitarse a seguir un sentido en la mesa, por ejemplo de izquierda a derecha. Para muchos jugadores con esto es suficiente. Pero si se desea una mecánica algo más compleja que tenga en cuenta los valores de los personajes se puede calcular la iniciativa tomando su Nivel más el número de los dados acumulados en la reserva de CPU en el caso de que tuviesen alguno (**Nivel + CPU**). Un programa que cuenta con más Tiempo de CPU siempre está más "acelerado" que otro con menos.

No es necesario lanzar dados. Ese valor será el orden a seguir siguiendo un orden de mayor a menor. Por lo tanto, siempre actúa primero quien tenga el valor más alto y se continúa sucesivamente hasta llegar al más bajo. En el caso de que dos jugadores tengan el mismo turno pueden actuar a la vez, aunque siempre queda la cortesía para ceder el turno y realizar las tiradas, una solución que a menudo funciona bastante bien. Para calcular su turno el Sistema usa el tamaño de su propia reserva. Mediante este sistema es posible que ante amenazas de gran poder el Sistema obtenga la iniciativa a menudo. Es normal. Al fin y al cabo los recursos del Sistema no pueden compararse a los de ningún programa. No es poco frecuente por lo tanto que pueda sorprender a los personajes en numerosas ocasiones.

Como una sugerencia por si se desea utilizar la iniciativa, **cundo un oponente en este modo de juego consigue sorprender a otro o ganar la iniciativa obtiene ventaja** sumando un éxito al resultado final, pero solamente durante el primer asalto.

En el enfoque relativo un oponente que sorprenda a su contrincante o gane la iniciativa obtiene una ventaja durante el primer asalto. Suma un éxito al resultado de su tirada.

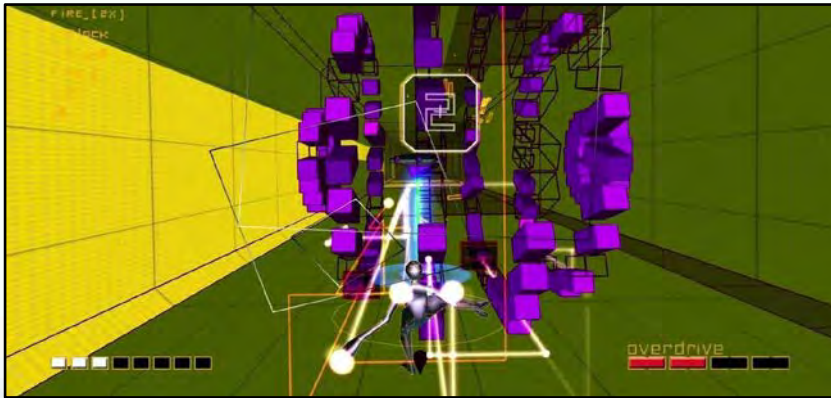
Por lo general **lo más importante de la iniciativa es saber cuál de los bandos actúa antes** para saber si obtiene o no alguna ventaja táctica. Para averiguar qué bando lleva la iniciativa busca el valor de iniciativa más alto de todos los participantes —o el del líder de todos los bandos implicados en el caso de que los haya—, quien lo tenga logra que todo su bando pueda actuar el primero, lo que puede concederle ventaja. Este método permite simplificar mucho el proceso. Por otra parte, el orden de iniciativa entre los miembros de un mismo bando puede rodarse a conveniencia durante la sucesión de asaltos ya que a menudo todos sus participantes tratan de acomodarse los unos con los otros buscando una sinergia de grupo que les permita tener éxito en sus estrategias.

Recuerda: en un enfrentamiento los oponentes que interactúan entre ellos, por ejemplo al estar envueltos en un combate, siempre hacen sus tiradas a la vez y se comparan todos los resultados obtenidos.

Prestar ayuda

La colaboración en un grupo de personajes utiliza el mismo método que el explicado en el enfoque absoluto. Todos los personajes que deseen prestar ayuda a otro pueden añadir los éxitos obtenidos en su reserva de Operaciones a la tirada del personaje al que estén apoyando. Los resultados de las demás reservas se ignoran igualmente y no es posible usar Rutinas o Utilidades así como cualquier otra ventaja que pueda aumentar ese número de éxitos.

Se aplican las mismas reglas que ya se han descrito. Las consecuencias de la tirada del personaje que haya recibido la ayuda afecta a todos los miembros que le hayan apoyado. Por otra parte, el líder es el que puede sufrir las consecuencias de fracasar en su acción.



Es importante tener en cuenta que al prestar ayuda en este modo de juego los miembros de un grupo pueden dividirse apoyando a unos u otros por separado. Esto tiene la ventaja de poder usar acciones tácticas que favorezcan los resultados, lo que incluye tener en cuenta un orden de iniciativa. De este modo un personaje podría ayudar a otro miembro de su grupo mientras los demás actúan por su cuenta por ejemplo. Si en algún momento lo desean podrían sumarse al personaje para apoyarle durante la sucesión de asaltos.

Pero también conviene recordar que al tratarse de acciones discretas en ocasiones puede existir un límite para la cantidad de personajes que pueden ayudar a otro miembro. Eso es algo que depende de la situación planteada y

tanto el Director como los jugadores pueden supervisar si la acción necesita establecer algún límite para poder recibir ayuda.

Las consecuencias del fracaso

Un fracaso siempre es un fracaso y sus consecuencias funcionan tal y como ya se ha descrito en los apartados anteriores. Pero cuando en el contexto de la escena un fracaso durante un asalto suponga un perjuicio que afecte a la integridad del código del programa, el jugador lo primero que debe hacer antes que cualquier otra opción es **recorrir a sus Casillas de estrés**. Una vez las ha completado y ya no tenga más casillas libres (o porque se ha optado por no usar el bloque de integridad) el jugador tiene tres opciones para su programa:

I. Puede recurrir a sus Protocolos de respuesta, pero **sólo una vez durante toda la escena que dure el conflicto** y siempre que no se haya utilizado ya en el mismo asalto o en los anteriores por otras causas. Siempre se señala una sola marca de los Protocolos, la que el Director decida, independientemente de la cantidad de daño recibido.

II. Puede compensar recurriendo a un aumento del Tiempo de CPU para sobrecargarse siempre que no lo haya tenido que hacer ya **como consecuencia** de su tirada durante ese mismo asalto. El incremento de la velocidad es siempre de **1 dado**, independientemente de la cantidad de daño recibido. Es posible usar esta opción aunque el jugador hubiese decidido ya aumentar su Tiempo de CPU antes de hacer su tirada.

III. Puede recurrir a sus Casillas de corrupción señalando tantas casillas como el número de éxitos que hayan sobrepasado a su tirada. Esto anula de forma **temporal** los dados de Operaciones que haya elegido para poner las marcas. Para elegirlos debe atenerse a sus reglas específicas de comenzar siempre por la izquierda e ir llenando los grupos antes de poder pasar al siguiente de la derecha.

Las reglas para aplicar los efectos son similares a los del enfoque absoluto con algunos matices ya que hay que tener en cuenta que se hacen tiradas consecutivas. Si ya se ha recurrido a los Protocolos de respuesta durante un asalto anterior ya no es posible volver a optar por esta opción hasta que termine el conflicto. Si la escena es un combate por ejemplo, sólo podría

hacerlo una vez. Por otra parte, si durante el mismo asalto ya se ha tenido que aumentar el Tiempo de CPU como consecuencia de la tirada no es posible usarlo de nuevo para compensar, por lo que se está obligado a recurrir a las otras dos opciones siempre que estén disponibles. En el siguiente asalto no obstante, y siempre que sea posible, el jugador puede optar por recurrir al incremento de su Tiempo de CPU aunque antes de actuar hubiese decidido ya aumentarlo. Las Casillas de corrupción en cambio siempre están disponibles, sean cuales sean las veces que hayan tenido que usarse. Normalmente se recurre a ellas cuando ya no es posible optar por ninguna de las dos opciones anteriores. Cuando es imposible usar cualquiera de la tres opciones el programa sufre un error, se cierra o es destruido.

ESTRUCTURA DE LAS PRUEBAS

Las pruebas se usan para resolver los conflictos. El procedimiento más básico de realizar una prueba consiste en hacer una tirada de dados. Para hacerla es fundamental tener muy claras las reglas de las reservas. Aunque ya se han explicado **se resumen aquí para ayudarte a recordarlas**.

—Las tres Reservas del jugador son: Operaciones, Tiempo de CPU y Memoria.

—Primero se toman siempre todos los dados disponibles en la Operación que entre en juego. La reserva total se completa con el resto.

—El sistema usa una sola reserva de un mismo color. Puede fragmentarla en unidades más pequeñas si lo desea.

—La reserva total son los dados de todas las reservas que se lanzan al hacer una tirada. El tamaño máximo de esta reserva nunca puede ser mayor de 16 dados.

Para hacer una prueba siempre se comienza cogiendo todos los dados que tenga disponibles el personaje en una de las dos Operaciones, la que entre en juego. La reserva total se completa con el resto de los dados de CPU y Memoria hasta llegar al tamaño máximo. **Los dados que no caben en la reserva total no cuentan para contabilizar éxitos o fracasos pero todos**

ellos se consideran siempre a su valor máximo absoluto al hacer una tirada (6 si son D6; 8 si son D8..., etc.). Estos dados siempre hay que tenerlos en cuenta para saber qué reserva es la que domina.

La tirada simple es la prueba más básica. El jugador lanza sus dados y cuenta el número de éxitos obtenidos. En unos casos esa tirada se puede comparar con otra y en otros lo que se necesita es obtener un número de éxitos determinado. A estas dos formas de resolver las tiradas se las denomina tiradas **con oposición activa o pasiva**. Si cuanto se desea es saber el número de éxitos para determinar un efecto se trata de una tirada **sin oposición**.



OPOSICIÓN ACTIVA

Siempre que un bando lanza los dados se considera que su tirada es una acción activa. Si esa acción es una respuesta a otra acción opuesta cada vez que los lanza para enfrentarlos a otra tirada se trata de una tirada contra una **oposición activa**. Se trata de los casos en los que **dos fuerzas se oponen activamente** una contra la otra al tener intereses opuestos. El ejemplo más común es el combate.

Una de las dos debe obtener un número mayor de éxitos para poder ganar. Un empate en las tiradas puede describir distintas situaciones que dependen

del planteamiento de la escena. Como regla general **un empate favorece al jugador o al bando que lanza los dados (tirada activa) con una victoria muy justa que puede tener alguna consecuencia negativa**. La consecuencia puede localizarse a nivel de la narración o de las mecánicas, por ejemplo obteniendo alguna desventaja temporal a la prueba siguiente si es el segundo caso. Hay otros casos en los que del resultado se deduce que la energía aplicada a una tarea está tan equilibrada con su dificultad que es necesario continuar hasta que se obtenga un resultado. En términos de juego **esto significa que ningún bando obtiene la victoria por el momento**. En un combate por ejemplo, un empate supone un forcejeo entre las fuerzas enfrentadas sin que se produzcan resultados.

Cuando se trata de la adversidad la tirada del sistema representa la dificultad para poder sortear un obstáculo o prevalecer sobre los desafíos. En ese caso el Director lanza un número de dados que reflejan el nivel del problema. Pueden ser dos, tres, cuatro, seis, siete, ocho..., hasta un total de quince dados. A continuación se muestra una tabla que puede servir como orientación para fijar el valor de dificultad de cualquier obstáculo o amenaza.

Dificultad	Nivel (activa)	Éxitos (pasiva)
Fácil	Entre 1 y 2 dados.	1
Moderado	Entre 3 y 4 dados.	2
Desafiante	Entre 5 y 8 dados.	3 - 4
Difícil	Entre 9 y 12 dados.	5 - 6
Muy difícil	Entre 13 y 16 dados.	7 - 8

OPOSICIÓN PASIVA

Cuando una tirada no se enfrenta a otra sino que es necesario superar una cantidad mínima de éxitos con ella se trata de una tirada contra una **oposición pasiva**. Sigue existiendo alguna forma de desafío sólo que esta vez se trata de una dificultad con un valor fijo.

En una tirada **sin oposición** en cambio, sólo se desea saber la cantidad de éxitos que se han obtenido para poder determinar los efectos de una acción,

como el uso de algunas Utilidades. Un ejemplo sería la dificultad para romper un código secreto, para ejecutar un comando, transformar una sección del entorno digital, usar un Complemento o hacer uso de las ventajas de una Utilidad.

En caso de empate se sigue la misma pauta que lo explicado en la oposición activa. Normalmente **el resultado favorece a la tirada activa** aunque nunca está exento de sufrir alguna consecuencia. En otros casos es posible que no pueda haber un vencedor por el momento.

La tirada pasiva sirve para imponer un valor fijo de dificultad sobre algo pero también tiene la ventaja de permitir al Director ahorrar tener que hacer tiradas. Puede decidir hacerlas si lo cree conveniente, especialmente cuando el Sistema tenga que hacer varias. Es una resolución de conflictos más rápida que la anterior y que exige menos esfuerzo. En realidad un Director puede decidir optar por este mecanismo pero entonces pierde la posibilidad de detectar Anomalías al no poder dominar con sus resultados en una tirada.

Una buena forma de averiguar el número de éxitos necesarios para establecer el nivel de dificultad de una tirada pasiva es dividir a la mitad el número de dados que serían necesarios en el caso de que se tratase de una tirada activa. Por ejemplo, una dificultad moderada de 4 dados supondría un nivel de dificultad de 2 éxitos. En la tabla de niveles de dificultad del apartado anterior se describe la cantidad de éxitos necesarios para cada nivel de dificultad si se lanzaran los dados. Puedes usarlos como una guía de consulta.

LA FLEXIBILIDAD DE LOS RESULTADOS

En muchos casos es necesario determinar quien ha obtenido la victoria y quien ha sido derrotado para solucionar los conflictos. El combate es un buen ejemplo de esta necesidad. Pero también es recomendable interpretarlos con flexibilidad ya que ayudan a describir qué es lo que ha sucedido al realizar una prueba. En ocasiones no es tan importante conocer si un resultado ha sido una victoria o una derrota o si algo es blanco o negro sino conocer lo bien o mal que se ha llevado a cabo una acción. Sus tonos de gris permiten llevar la narración por caminos distintos. La lectura de la diferencia de los éxitos

obtenidos en una tirada es un medio para hacerlo. **Lo que se ofrece aquí es una guía** de las distintas posibilidades. Lo más importante es entender esta lectura como una interpretación flexible para poder improvisar dependiendo de los resultados.

- Si la diferencia de éxitos en una victoria es de 3 o más a favor se obtiene un resultado espectacular. El personaje o bando implicado ganador obtiene una consecuencia favorable o un giro de los acontecimientos a su favor. Debido a ello es posible obtener una ventaja en la siguiente prueba en ciertos casos.
- Si la diferencia de éxitos en una tirada ha sido de 1 ó 2 éxitos tanto a favor o en contra tanto para la victoria como para la derrota se considera que el resultado ha sido apropiado. Se ha estado cerca del equilibrio de fuerzas y tanto ganar como perder ha resultado una prueba reñida. No se producen efectos a nivel de reglas.
- En caso de producirse un empate las fuerzas están muy equilibradas. El empate normalmente favorece a la oposición activa o al jugador pero en muchos casos las acciones pueden quedar en tablas dependiendo de la situación. Un empate igualado requiere de tener que romper el empate por lo que el conflicto deberá continuar. Si se da esta situación y los jugadores están de acuerdo también es posible forzar el desempate optando por obtener la victoria, pero siempre pagando un precio. El jugador debe aceptar alguna consecuencia, como asumir un penalizador o cualquier otra opción que resulte apropiada. Si la resolución del desempate favorece a un bando ello implica algún giro de los acontecimientos en su contra o algún inconveniente añadido a su victoria. Significa un: "Sí, has ganado pero..." que deberá aplicarse dependiendo de la situación. En muchos casos es posible aplicar una desventaja a la siguiente prueba, en otros, como en las pruebas de estructuras distintas puede llegar a anular una de ellas, especialmente si ha consistido en tratar de obtener una ventaja.

- Si la diferencia de éxitos en una derrota es de tres o más en contra se obtiene un fallo catastrófico. El personaje o el bando implicado que ha perdido sufre alguna consecuencia o un giro de los acontecimientos que lo perjudica. Debido a ello en muchos casos es posible obtener una desventaja en la siguiente prueba o directamente perder la posibilidad de realizar la acción siguiente.



ACCIONES

En Scroll es posible realizar muchísimas acciones. En ese sentido se puede decir que hay infinitas posibilidades. Pero lo interesante —y curioso— es que todas se pueden resumir en tres grandes grupos. La mayoría de acciones pueden darse en los dos enfoques, el relativo o el absoluto. Por sus características el enfoque relativo permite más opciones lo que hace necesario tener que aclarar algunos detalles. El enfoque absoluto por lo general se resuelve mucho más fácilmente.

Las acciones se pueden resumir en las siguientes: el **enfrentamiento**, **superar un obstáculo** y tratar de **obtener una ventaja u oportunidad**.

I. El enfrentamiento

Cuando dos fuerzas con intereses opuestos chocan entre sí se produce un conflicto. Si es tan extremo como para querer hacerse daño mutuamente surge el enfrentamiento. Una forma muy común de resolver conflictos que suelen ser complejos y dinámicos. Un enfrentamiento puede ser de cualquier tipo, no exclusivamente un combate. Por ejemplo, si un personaje está siendo interrogado por los agentes del sistema existe un tipo de enfrentamiento entre los que desean extraer información por la fuerza y el que trata de resistirse. Unos tratan de obligar al prisionero a hablar mientras que el otro intenta evadir las preguntas dando información falsa o evitando contar nada. Otro ejemplo de un enfrentamiento es una discusión acalorada donde dos oponentes tratan de vencer al contrario presentando sus argumentos. Cada uno realiza un ataque contra el otro en el que ambos se defienden a su vez, por lo que el daño en este caso representa la disminución de la fuerza de los argumentos de cada uno.

La resolución del enfrentamiento ya se ha explicado. En el enfrentamiento ataque y defensa son simultáneos. Todos los bandos enfrentados realizan sus tiradas a la vez y se comparan. La diferencia de éxitos a favor de uno u otro permite saber quién ha obtenido la victoria. Si en el enfrentamiento se tiene la posibilidad de infringir daño siempre **es igual al número de éxitos de esa diferencia**.

Las amenazas sufren la pérdida de un dado por cada éxito de **diferencia en su contra**, lo que se traduce como un impacto o punto de daño. Cuando todos sus dados han sido eliminados son derrotados, aunque en algunos casos pueden decidir huir o rendirse antes de perderlos todos.

En situaciones dudosas siempre tiene prioridad el que haya obtenido más éxitos para saber quién aplica primero los efectos de la tirada y se continúa siguiendo ese orden hasta llegar al que haya obtenido el peor resultado. Si hay más de dos bandos implicados en una disputa el bando que obtiene más éxitos aplica sobre los demás la diferencia que corresponda en relación a cada uno.

En el enfoque relativo la confrontación continúa cada asalto entre todos los implicados hasta que se soluciona el conflicto. Se actúa solamente una vez por asalto por lo que los dados se lanzan una vez. En caso de dudas, cualquier efecto que se produzca debido al resultado de la reserva dominante durante un mismo asalto sólo se puede aplicar una vez. Si por cualquier razón un jugador pudiera hacer más tiradas después de la primera durante el mismo asalto los resultados de la reserva dominante se ignoran. Esto puede suceder si los jugadores desean alterar el sistema de resolución de acciones por lo que recuerda esta regla ante cualquier conflicto que pueda surgir con el reglamento.

La diferencia de éxitos entre las tiradas indica el bando ganador, la cantidad de daño infligido y el orden en el que se aplican sus efectos.

En el enfoque relativo, el límite de los efectos de la reserva dominante es siempre de uno por asalto. En caso de duda, **durante un mismo asalto** siempre se ignoran los efectos de la reserva que domine más allá de la primera tirada en los demás resultados.

Algunos Elementos, Complementos y reglas específicas pueden añadir una cantidad de éxitos a la tirada que cuentan para el resultado final y para contabilizar la cantidad de daño total. En ocasiones los éxitos sólo cuentan para calcular el daño y en otros sólo para realizar la prueba. Pero en estos casos se trata siempre de excepciones que se detallan allí cuando sea necesario.

Actuando a la defensiva

En algunos casos es posible que un oponente no quiera infligir un ataque durante el enfrentamiento sino limitarse a defenderse. Es posible actuar a la defensiva en muchas situaciones, no sólo en un combate. Un programa que esté siendo interrogado y que se dedique a poner excusas está actuando a la defensiva. ¿Pero cómo es posible hacerlo si una tirada es a la vez ataque y defensa?

Un jugador puede declarar antes de hacer su tirada que su personaje se enfoca en la defensa. Cuando un programa se mantiene a la defensiva en un enfrentamiento no provoca daños al hacer su tirada, pero en esa acción **obtiene doble ventaja, es decir dos éxitos extra** para contrarrestar los efectos de las amenazas. Estos éxitos se añaden al resultado total de su tirada. Cada éxito cuenta como un dado más de valor nulo en la reserva de Operaciones para saber qué reserva domina en caso de empate entre reservas.

Personaje indefenso

Si por cualquier razón un personaje no puede responder ante una amenaza, ya sea porque está bloqueado o detenido, se le considera indefenso. Un personaje indefenso no puede presentar oposición por lo que es una presa fácil para sus amenazas. Todos los éxitos obtenidos en las tiradas que se hagan contra él no se comparan con ninguna otra por lo que tienen su efecto normal.

II. Superar un obstáculo

Las pruebas más comunes que se dan en el juego consisten en tratar de superar los obstáculos que se interponen entre el personaje y sus objetivos. El número de acciones que es posible hacer en el juego es en teoría ilimitado. El personaje desea hacer algo y lanza sus dados para saber si lo consigue y en qué grado.

Merece la pena prestar atención a ese grado de éxito o de fracaso pues es muy conveniente no limitarlo a una simple "victoria o derrota". Decir que "lo pasas" o "no lo pasas" es limitar la descripción de los resultados de las pruebas y por consiguiente de las posibilidades del juego. Como ya expliqué un poco más atrás, la narración de los acontecimientos se beneficia de hacer una **interpretación flexible de los resultados**.



Si los jugadores están de acuerdo, también es posible optar por obtener la victoria en caso de sufrir una derrota de grado bajo (de uno o dos dados como mucho), pero siempre a un alto precio. Algo que también se hace extensible a forzar un desempate por supuesto. El jugador que lo sugiera debe aceptar alguna consecuencia importante como quedar dañado, asumir un penalizador importante o cualquier otra opción que resulte apropiada. También existe la posibilidad de gastar uno o más puntos Hack para conseguirlo, lo que sirve como ejemplo de un uso creativo de esta reserva de puntos. No obstante, **este apunte no es más que una sugerencia**. En el apartado sobre el uso de los puntos Hack se incluye como una posibilidad más de sus aplicaciones.

La dificultad de la oposición para superar los obstáculos puede ser activa o pasiva, como ya se ha descrito. Elegir una u otra depende de la situación y optar por la más adecuada beneficia el transcurso de la partida.

III. Crear una ventaja u oportunidad

Otro tipo de acción consiste en tratar de obtener alguna ventaja que pueda beneficiar a quien la realiza o a otros personajes en sus siguientes acciones. La ventaja puede consistir en un modificador, como poder añadir un éxito extra para él o sus aliados o bien conseguir alguna oportunidad que le permita realizar otras acciones.

Cuando un personaje desea obtener una ventaja realiza su tirada contra una dificultad cuya oposición puede ser activa o pasiva. Por lo general se trata del mismo nivel de dificultad que la fuerza sobre la que actúa, aunque es posible que ese nivel pueda reducirse un cierto grado dependiendo de las circunstancias. En otros casos se establece el nivel de dificultad que parezca apropiado. Si se consigue la victoria el personaje se beneficia de un éxito extra que podrá usar en su siguiente acción o bien dárselo a otro personaje que pueda usarlo durante el mismo asalto en el que hizo su prueba. Una prueba exitosa con una gran diferencia de éxitos puede conceder una doble ventaja.

También existe la alternativa de realizar la prueba para provocar una desventaja en el contrario. Son dos formas de verlo y es posible optar por cualquiera de las dos. En este segundo caso esa desventaja puede beneficiar al resto de sus compañeros en sus siguientes acciones. Evidentemente, el afectado podrá emprender acciones para eliminarla. La mecánica de las ventajas y las desventajas se explica un poco más adelante en este mismo capítulo.

En la sección siguiente se explican otros tipos de pruebas con estructuras distintas. En este tipo de pruebas se puede tratar de obtener ventajas o de provocar desventajas en el contrario mediante esta acción en muchos casos. Pero dependiendo de cada tipo de prueba las consecuencias de obtener la victoria o la derrota pueden cambiar. La clave es comprender que al tratar de beneficiarse de una situación que suponga una ventaja siempre se asume un riesgo. En caso de fracaso la acción puede conseguir el efecto contrario, provocando una desventaja, y en el peor de los casos malograr las acciones que se pretendían mejorar.

PRUEBAS CON ESTRUCTURAS DISTINTAS

Con la prueba simple es posible resolver la mayoría de las situaciones que pueden darse en el juego. Tratar de romper un código de seguridad, forzar una puerta o activar un dispositivo normalmente no necesita más que realizar una prueba simple. De ella derivan los métodos básicos de resolver los conflictos que ya se han explicado, incluyendo un combate. Pero hay casos en los que la narración de los acontecimientos puede beneficiarse de disponer de otras formas de realizar las pruebas.

Ten en cuenta que las estructuras que se describen en este apartado están dirigidas en su mayor parte al **enfoque relativo** ya que tendrán más sentido. Por la forma de resolver las tiradas en el enfoque absoluto no es posible usarlas en muchos casos ya que sus mecánicas requieren de un orden de asaltos o de que varios personajes participen haciendo acciones distintas. Recuérдалo para saber cómo actuar ante las tiradas dominantes y la obtención de Anomalías.



Las pruebas no pretenden en ningún caso reemplazar la narración de los hechos sino ofrecer mecanismos que sirvan de armazón para elaborarlo, y hasta para construir los pequeños relatos que van componiendo la trama. Están ahí para ayudar a resolver los conflictos pero al mismo tiempo para

ayudarte a describir lo que pasa durante la partida. Por eso mismo se consideran herramientas narrativas. La estructura de estas pruebas **exige envolverlas, describiendo los hechos que se extraen de ellas al usar sus mecánicas**; si no, al final no serán más que la sucesión de una tirada tras otra.

Además de las formas de resolver los conflictos que ya se han explicado, en este juego se describen otras tres estructuras para realizar algunas pruebas. Aquí se denominan: el **Reto**, el **Desafío de habilidad** y el **Duelo**. Elegir la más adecuada exige algo de reflexión y un poco de práctica. En algunos casos es posible usar varias a la vez para describir una misma situación pero la mayoría de las veces unos tipos de pruebas se ajustan mejor a unas situaciones determinadas que a otras.

RETOS

Un reto consiste en enfrentarse a un número fijo de pruebas simples, normalmente contra una oposición pasiva. De ellas se debe obtener la victoria en una cantidad determinada para que el reto pueda considerarse superado. Pueden ser todas, la mitad o un tercio de la cantidad total. Por ejemplo, si fuese necesario realizar cuatro pruebas, se puede establecer que para superar el reto es necesario obtener la victoria en dos, en tres o en las cuatro. La cantidad a superar es algo que depende del tipo de reto planteado lo que indica su dificultad. Por otra parte, los grados de dificultad de cada prueba por separado pueden ser siempre los mismos o ir variando.

Sufrir una derrota en algunas pruebas no supone que una acción no haya podido realizarse sino más bien que al hacerla **ha surgido algún inconveniente**, pero acumular un mínimo de esos inconvenientes provoca que el reto fracase. Los empates se tratan como se ha explicado; en algunos casos supone la obtención de la victoria para quien haga la prueba activa (tire los dados) o para el jugador, pero siempre con alguna consecuencia. Un empate es un acierto por los pelos y así debería quedar reflejado en el juego.

Una derrota durante el proceso puede tener como consecuencia que la oposición obtenga el derecho a realizar una tirada que si supera dé al traste con todo el reto. Esto puede suceder por ejemplo cuando un personaje trata de infiltrarse en un recinto usando el sigilo. Cualquier derrota sufrida al hacer las pruebas implica que sus enemigos puedan realizar otras para intentar descubrirle.

En Scroll el reto es un tipo de prueba compleja que puede darse a menudo por lo que es conveniente estar familiarizado con su mecánica. Puede consistir en hackear una parte del Sistema, acceder a un recinto o crear algo complicado que necesite de tiempo y de la necesidad de seguir un proceso delicado. En muchos casos el objetivo es algo dinámico, una situación que puede cambiar y seguir su curso, pero casi siempre está basado en seguir un procedimiento. Para conseguirlo suele ser necesario resolver tareas distintas organizadas en pasos o en seguir unas reglas, cada una relacionada con unos conocimientos o capacidades específicos.

Los retos normalmente se utilizan para superar obstáculos. Es poco frecuente que sea posible sustituir alguna de las pruebas que requieren por otra acción. Uno de los personajes implicados puede intentar hacer una acción extra para obtener una ventaja que ayude en alguna de las pruebas pero en ningún caso cuenta como una de ellas. En caso de fallar siempre habrá alguna consecuencia, por ejemplo obteniendo una desventaja, añadiendo una nueva prueba al reto que tener que resolver o en el peor de los casos provocando que falle la prueba sobre la que se pretendía obtener la ventaja; por no hablar de la posibilidad siempre presente de terminar sabotando todo el reto...

Para superar un reto es necesario realizar una cantidad fija de pruebas en las que se debe obtener la victoria en una cantidad determinada de ellas. Si no se consigue fracasa.

Pero el reto no consiste en hacer las tiradas y ya está porque entonces cuanto estamos haciendo es solucionarlo todo lanzando dados igual que en el parchís.



Lo más adecuado es secuenciar las acciones describiendo cuales son los acontecimientos que van teniendo lugar. A medida que se van superando se avanza en la narración hilvanando los hechos y componiendo la escena que está teniendo lugar.

Este procedimiento en realidad se aplica —o se debería aplicar— a todas las demás pruebas que es posible hacer en el juego, no sólo a las de este tipo. Recurrir a las mecánicas para ayudarte a construir la escena es el verdadero objetivo de estas reglas.

EJEMPLO

Andrómeda se descuelga por una cuerda para acceder a la bóveda donde se esconde un importante fichero de datos. Se desliza como una araña por su seda en absoluto silencio. Al otro lado de la puerta, dos peligrosos agentes montan guardia. El Director establece que para tener éxito en su misión debe realizar 4 pruebas y obtener la victoria en al menos 3 de ellas. Una para bajar por la cuerda; otra para no hacer ningún ruido; otra para darse la vuelta balanceándose, tomar el fichero de su emplazamiento y guardarlo en silencio y una cuarta para volver a subir. La dificultad de la primera prueba es fácil, la segunda moderada, la tercera difícil, y la cuarta otra vez fácil. Todas son pruebas pasivas por lo que es necesario superar una cantidad de éxitos determinada para cada prueba.

Andrómeda supera la primera prueba y se desliza con facilidad. Cuando hace la segunda tirada sufre una derrota, comete un desliz y produce un ruido. Aguarda unos momentos y prosigue con su misión. Al llegar al fichero realiza una tercera tirada que supera por lo que toma el fichero. Si hubiese fallado lo hubiese tomado de igual forma, pero habría surgido algún otro inconveniente que habría dado al traste con su misión al haber superado ya el mínimo de derrotas. La última prueba la supera con facilidad por lo que escapa de la bóveda con el fichero en su poder y una sonrisa de satisfacción.

Reto por acumulación

Una variante del reto que puede usarse en Scroll muy a menudo es el reto por acumulación. Su mecánica funciona de una forma similar. Se va realizando una sucesión de pruebas, pero **el objetivo antes que obtener una cantidad de victorias consiste en acumular un número de éxitos hasta llegar a una cantidad** determinada. Los fracasos se adaptan al contexto de la escena, pueden retrasar la superación del reto o conducirlo a la derrota.

Si un personaje debe acumular siete éxitos en sus tiradas para superar un obstáculo por ejemplo, deberá realizar pruebas sucesivas hasta conseguirlos todos. Si durante el proceso los fracasos tienen como consecuencia que dentro del contexto de la escena no pueda continuar realizando las pruebas entonces el reto fracasa. Esto podría suceder porque antes de reunir la cantidad de éxitos requerida ha saltado una alarma que ha hecho que una amenaza vaya a investigar, lo que le impide al personaje continuar con sus acciones.

Los retos por acumulación son útiles para resolver situaciones donde uno o varios personajes deban esforzarse para conseguir un objetivo, como sobrepasar las defensas de un sistema, forzar la dureza de un mecanismo de protección o abrir un agujero en una pared por ejemplo.

DESAFÍOS DE HABILIDAD

Un desafío de habilidad consiste en obtener un número determinado de victorias antes que un número determinado de derrotas. Hasta aquí se parece mucho al reto pero su principal diferencia es que no hay un límite fijo de tiradas. Se realizan tantas pruebas como sea necesario hasta cumplir con uno de los dos requisitos.

El número mínimo de derrotas suele ser fijo, normalmente de tres. Aunque pocas cosas impiden poder cambiarlo es mejor dejarlo establecido en un valor. **El proceso resulta más sencillo si lo que varía es la cantidad de victorias necesarias para poder superarlo. Esa cantidad establecerá la dificultad del desafío.** Un valor de 2 victorias es un desafío fácil; uno de 3 de dificultad media y más de 5 comienza a complicarlo bastante. A partir de 6 podemos decir que puede ser muy difícil superar el desafío.

Otro factor que hay que tener en cuenta es que su complejidad no sólo depende del número mínimo de victorias a obtener sino también del nivel de dificultad de cada prueba y que al igual que en los retos ésta no tiene por qué ser siempre la misma. Para cada prueba se puede establecer un nivel de dificultad específico de oposición activa o pasiva; lo más usual es que se trate de la segunda. Los empates suelen resultar favorables para quien lance los dados, el jugador por lo general, pero asumiendo alguna consecuencia como de costumbre.

Los desafíos también son útiles para resolver escenarios muy dinámicos que dependan de la superación de una serie de obstáculos. Pero a diferencia de los retos, en un desafío esos obstáculos suelen ser impredecibles. Las pruebas no siguen una pauta o un procedimiento determinado sino que van surgiendo a lo largo del desafío. Esto exige tener que improvisar.

Para superar el desafío es necesario obtener una cantidad mínima de victorias antes que una cantidad mínima de derrotas haciendo todas las tiradas que se consideren necesarias hasta conseguir uno de los dos resultados.

Un desafío permite describir una mayor variedad de situaciones pero también puede resultar un poco más complicado de manejar. Muchos de los factores que se han explicado ya en los retos se pueden aplicar también a los desafíos. Cada victoria o derrota describe unos acontecimientos. Bien utilizado el desafío de habilidad puede ayudar a elaborar la descripción de los hechos de una escena. De igual modo, sufrir una derrota no implica necesariamente haber fracasado en una prueba sino el que surja algún inconveniente que puede hacer más difícil superarlo, por ejemplo quedar en desventaja frente a un oponente o respecto a una situación. También es posible realizar acciones para obtener ventajas que puedan favorecer a alguna de las pruebas. La acción no contaría ni como victoria ni como derrota pero fracasar debería tener consecuencias como obtener desventaja en la prueba siguiente, anular una de las victorias o bien contabilizar directamente una derrota en el desafío por ejemplo.

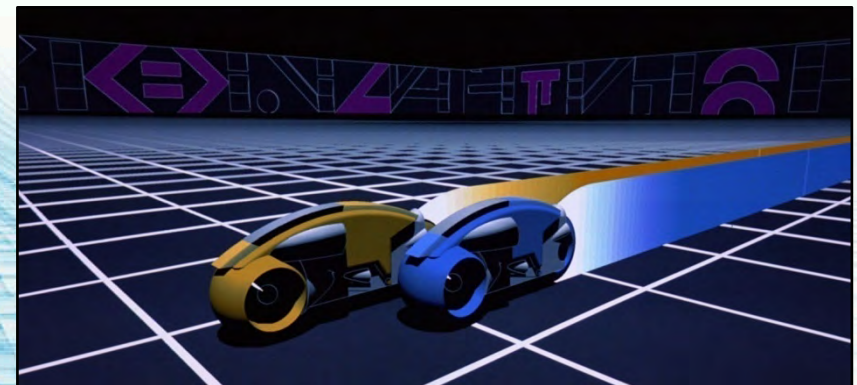
Es posible emplear el desafío incluso para provocar daños pero casi siempre de forma indirecta y como una consecuencia de las acciones que se realizan. Sería el caso por ejemplo de una carrera de motos al estilo de la película "TRON"©. Un desafío de este tipo podría consistir en realizar una serie de maniobras para lograr que otro competidor se estrelle contra el muro sólido que el vehículo del personaje deja a su paso. Estrellarse como vemos es una consecuencia de las acciones del desafío. Otros posibles usos del desafío pueden consistir en realizar persecuciones, intentar convencer a otro, liderar a una muchedumbre, tratar de engañar a alguien o realizar una carrera de obstáculos.

Ejecutar las mecánicas del desafío es fácil pero usarlas con efectividad puede requerir algo de práctica. No obstante una vez se use unas pocas veces resultará mucho más sencillo aplicarlas mientras se van descubriendo nuevos usos.

EJEMPLO

El programa Géminis termina dando con sus bits en el subsistema de un mundo virtual donde una jerarquía de programas dominantes gobierna con puño de hierro. Para contentar al pueblo organizan en la rejilla de juegos distintas competiciones, algunas muy peligrosas. La favorita es la carrera de motos de luz. Un vehículo capaz de dejar a su paso un muro sólido de datos. Estrellarse contra el muro significa el fin. La moto es capaz de desplazarse velozmente y de realizar giros cerrados de hasta 90 grados. El juego consiste en cerrarle el paso a los contendientes hasta que o bien sólo quede uno o gane uno de los equipos si la partida se desarrolla en esa modalidad.

El Director decide que la carrera se resolverá con un desafío de habilidad en el cual es necesario que el personaje obtenga 3 victorias antes de sufrir 3 derrotas. Si lo consigue conseguirá anotar un tanto para su equipo que puede suponer ganar la competición. Este tipo de competición en concreto se presta a que pueda exigir un número mayor de victorias por lo que se podría extender algo más.

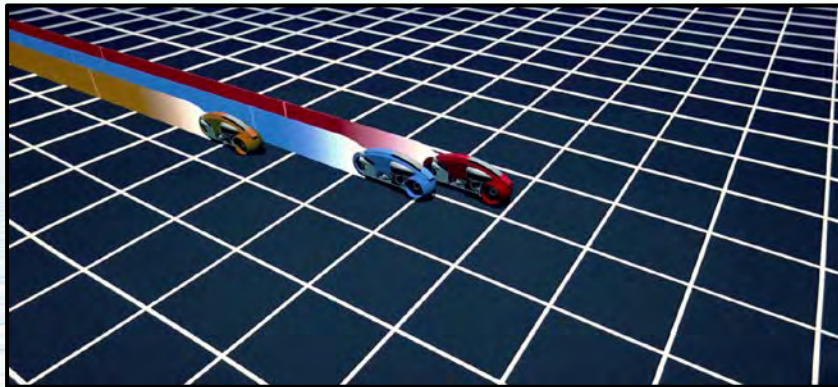


Géminis monta en su moto y arranca. La moto se reconstruye a su alrededor adaptándose a su código. A todos los efectos sigue las normas de un Elemento (ver reglas de Elementos). Comienza haciendo un par de maniobras para aproximarse a uno de los contendientes del otro equipo. Realiza una prueba enfrentada contra su nivel de

dificultad y obtiene una victoria. Ya puede anotarse un tanto. El oponente del equipo contrario tiene que alejarse sufriendo desventaja frente a Géminis. Más tarde cae bajo la acción de otro oponente.

Otro oponente se acerca y cruza delante de él cerrándole el paso con un muro de tonos rojizos. Géminis maniobra para esquivarlo. Hace una prueba enfrentada al nivel del oponente y falla. Eso no significa que se estrelle o que se caiga de su vehículo sino que en este caso ha surgido un inconveniente que le obliga a buscar un espacio abierto, se libra por muy poco y el oponente obtiene una ventaja.

Persigue al que le obligó a ceder y mediante una maniobra intenta acorralarlo. A pesar de que el contrario había obtenido una ventaja obtiene una victoria en las pruebas enfrentadas. Ya lleva dos tantos. Cierra con el muro de datos de su moto y se la juega intentando cerrar de nuevo al girar hacia la derecha cruzando delante de él. El Director decide que va a ser una maniobra difícil por lo que el oponente volverá a obtener ventaja. Al realizar la prueba y comparar las tiradas Géminis obtiene otra victoria por tercera vez. El oponente se estrella dando un grito escalofriante contra el muro azulado de la moto de Géminis. Sus bits se esparcen por el espacio digital. ¡Géminis ha ganado! Ahora podrá continuar carrera o retirarse, depende de cómo se haya organizado la competición.



DUELOS

Aunque el término sugiera una forma de enfrentamiento muy concreto en realidad expresa un tipo de prueba que puede aplicarse a muchas situaciones diferentes, no solamente a los enfrentamientos amistosos. Un duelo es un conflicto entre fuerzas que tratan de medirse entre ellas. En cuanto a sus mecánicas, pretenden conseguir una meta pero con una diferencia muy importante, siempre se fija un límite de victorias para poder ganar prescindiendo de los efectos negativos que sufren sus oponentes. Por otra parte siempre son pruebas con una oposición activa en todos los bandos implicados pues forma parte de su razón de ser. Se acostumbra a pensar que los bandos implicados siempre son dos pero pueden ser más. Al fin y al cabo existe el ajedrez para tres...

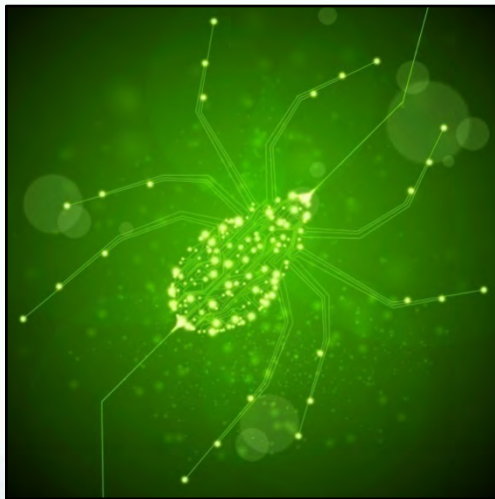
En la mayoría de los casos **en un duelo sus contendientes no tratan de hacerse daño o destruirse mutuamente por lo que no se tiene en cuenta el daño**. Si lo pretendieran ya no se trataría de un duelo sino de un combate en toda regla. Un duelo no obstante funciona mejor que un combate para resolver muchas situaciones ya que prescinde de sus consecuencias y se centra en conseguir un objetivo determinado. Un duelo debe ante todo ser emocionante, si no lo es podría resolverse con otro tipo de prueba. Si le dedicas un poco de tiempo descubrirás que un duelo puede surgir en muchas circunstancias, y no sólo en los deportes que al fin y al cabo son tipos de duelos muy comunes. Puede producirse un duelo cuando dos oponentes tiran de los extremos opuestos de una cuerda, cuando un coche intenta sacar a otro de la carretera, cuando dos oponentes forcejean para conseguir un objeto o arrebatárselo mutuamente, cuando alguien trata de sacar el tapón de una botella o incluso al tirar de la correa de una mascota poco colaborativa.

Los empates suelen representar un equilibrio entre ambas fuerzas que por el momento están igualadas, igual que en un combate. Por lo tanto, en caso de producirse en las pruebas ningún bando obtiene la victoria. Los contendientes tendrán que continuar el duelo mientras lo deseen (o mientras no tengan más remedio) hasta que haya un ganador. Un empate puede usarse como excusa para añadir un pequeño giro en los acontecimientos que afecta a todos los contendientes, un recurso que puede usarse también en un combate normal

por supuesto. Sería el caso de la aparición de nuevos problemas debido al entorno como una súbita polvareda, una explosión cercana o un cambio efectuado por el Sistema en el modelado del escenario por ejemplo.

Formas de resolver un duelo

Existen dos formas de realizar los duelos. La primera es muy fácil y consiste en obtener un número fijo de victorias antes que el contrario. La cantidad puede ser la que quieras pero como siempre resulta más sencillo si se establece una cantidad fija por defecto. En este caso **siempre será de tres**, aunque nada impide elegir cualquier otra cantidad que se ajuste a las pretensiones de la prueba. Ampliarla o reducirla prolonga la duración del duelo o lo acorta. Por otro lado, no se contabiliza el daño.



Las acciones pueden ser siempre las mismas pero un duelo que requiera hacer acciones distintas puede resultar muy interesante. Por ejemplo una acción podría consistir en golpear con el costado de un vehículo el de otro; la siguiente acción en

tratar de sobrepasarlo y una tercera en tratar de cerrarle el paso para que reduzca la velocidad o se detenga por completo. El duelo en este caso consiste en la carrera entre dos vehículos en donde uno intenta interceptar al otro. Observa que es posible usar también un desafío de habilidad con este ejemplo, salvo que en este caso se asume que las tiradas siempre son activas y que lo que cuentan son las victorias antes que el oponente, no el número de derrotas.

Mediante el primer método, para ganar un duelo es necesario obtener una cantidad determinada de victorias antes que el oponente.

Aunque no hay una intención directa de provocar daño desde luego este tipo de duelo resulta poco amistoso. El daño por lo tanto puede originarse como una consecuencia de las acciones que se realizan, casi siempre de una forma indirecta. Y es que el duelo no tiene que ser "amigable" necesariamente, sirve para resolver situaciones y en muchos casos la línea que separa al duelo del combate es muy fina. Desde el momento en el que creas que la cruza convierte el duelo en combate y ya está. En realidad las reglas no se diferencian mucho unas de otras. Las pruebas de un duelo cuanto hacen es establecer un número de victorias o "sangres" como condición para poder ganar, eso es todo.

Tratar de obtener ventajas durante un duelo es posible. Para ello un bando renuncia a la posibilidad de apuntarse un victoria en una de las pruebas con la esperanza de mejorar la siguiente tirada. Pero esto siempre implica un riesgo. Si fracasa su rival se anota un tanto, pero si además lo hace por una gran diferencia de éxitos —tres o más— obtiene doble ventaja en su siguiente prueba. Si la diferencia de éxitos al fallar el intento no ha sido muy grande, habiendo quedado cerca de estar igualados, su rival se apunta un tanto y obtiene una ventaja simple al hacer su siguiente prueba. En el caso de ganar la prueba el bando que lo ha intentado obtiene una ventaja en la prueba siguiente y si ha obtenido tres o más éxitos cuenta con al menos una doble ventaja.

Todos los demás detalles que se han descrito en los tipos de pruebas explicados antes se aplican aquí de la misma forma. El desarrollo de las pruebas del duelo van describiendo los hechos de toda la escena. Es necesario vestir las tiradas con una narración de los acontecimientos para darle una estructura de relato a lo que de otra forma no sería más que el armazón de unas mecánicas de juego.

Variante

El segundo método de realizar un duelo lo estructura de otra forma. En lugar de contabilizar un número de victorias cada contendiente trata de eliminar dados de Operaciones de su contrincante hasta que uno de ellos los pierda primero. De esta forma, a medida que se desarrolla el duelo sus capacidades van menguando, lo que refleja los efectos del conflicto sobre los implicados. Podría ser el caso de una competición deportiva o de una sesión de

entrenamiento con espadas samuráis entre dos usuarios en un entorno virtual simulado.

La regla general de esta variante consiste en eliminar un dado, y sólo uno, por cada victoria obtenida. También es posible eliminar tantos como la diferencia de éxitos entre ambas tiradas, pero en ese caso la duración del duelo se acortaría bastante, lo que puede ser una opción viable si lo que se desea es no dilatarlo demasiado. Por cada victoria se pierden solamente dados de Operaciones, no de ninguna otra reserva.

Mediante el segundo método, gana el duelo quien consiga que su oponente pierda primero toda la reserva de su Operación.

La pérdida de dados es simbólica. Sólo como parte de las mecánicas del duelo. Una vez termina los contendientes disponen de todos sus puntos. Aunque en algunos casos el Director puede sugerir que es necesario que los implicados recobren un poco la energía y la estabilidad de su código en el caso de ser programas, ...o el aliento en el caso de ser usuarios.

EJEMPLO

Dos tripulantes de la nave Osiris, aprovechando un par de horas libres entre guardias, deciden entrenar un poco sus habilidades de lucha. Se conectan al simulador y el operador recrea para ellos un tatami japonés donde podrán practicar en paz en un entorno apacible. Para hacer el duelo el Director ha decidido usar la variante o segundo método consistente en la eliminación de dados de Operaciones. Gana el primero que consiga eliminar todos los puntos del contrario en sus Operaciones de Potencia. En un duelo no se tiene en cuenta el daño ni las consecuencias de los ataques con éxito. Lo importante es conseguir una victoria o un objetivo.

Se ponen en guardia y comienzan el duelo. Una vez comienza la lucha ambos jugadores van describiendo sus acciones. De no hacerlo el duelo no consistiría más que en una sucesión de tiradas.

Uno de ellos hace una finta para engañar al contrario por lo que se trata de una acción para obtener una ventaja. Renuncia a obtener la victoria en caso de una prueba exitosa con el fin de mejorar sus posibilidades para la siguiente. El jugador describe que además va a intentar dar un salto sobre la cabeza de su oponente para aterrizar a su espalda. Ambos realizan sus tiradas el personaje que ha intentado la maniobra tiene éxito. En lugar de contabilizarse una victoria en el duelo el jugador obtiene ventaja para su próxima prueba.

El duelo continúa y la ventaja da sus frutos. El jugador consigue anotarse una victoria aunque en los siguientes su oponente consigue remontar: dos victorias..., al llegar a tres a uno de ellos sólo le queda un dado en sus Operaciones de Potencia. Cuando de repente se escucha una sirena de alarma...

...

«ERROR DE DATOS»

<<LA TRANSMISIÓN SE HA INTERRUMPIDO. IMPOSIBLE RESTABLECER LA COMUNICACIÓN CON OSIRIS. ES MUY PROBABLE QUE LA NAVE SE HAYA PERDIDO>>

VENTAJA Y DESVENTAJA

Las ventajas y las desventajas representan condiciones que benefician o dificultan las acciones de los personajes. Puede tratarse de cualquier cosa, desde los detalles del entorno hasta el estilo y las vulnerabilidades de un programa. Es posible obtener ventaja al tomar por sorpresa a una amenaza, por ganar la iniciativa en los primeros momentos de un combate, por estar a bocajarro para disparar, por estar en una posición superior, por contar con un arma más potente, por ser capaz de ver en la oscuridad..., etc. La regla de ventaja y desventaja es una herramienta que permite poder sacarle partido a ciertos aspectos, obteniendo ventajas en algunas situaciones y provocando desventajas en otras. Con ellas todo es posible y sus efectos se aplican en el juego con frecuencia y de formas distintas.

En muchas ambientaciones por ejemplo las características del escenario son fundamentales. Un objeto en un mundo virtual puede ofrecer cobertura concediendo ventaja para poder evitar un ataque a distancia. Algunos detalles pueden conceder una doble ventaja o desventaja, duplicándola, pero estos casos deberían ser siempre situaciones especiales en los que el Director debe decidir si es o no aplicable.

Es posible aplicar varias ventajas distintas apilándolas o anularse las ventajas y las desventajas entre sí. La única regla a tener en cuenta es que dos ventajas o desventajas del mismo tipo no pueden acumularse. Un programa



que sufra desventaja por haber sufrido un efecto que haya ralentizado su ejecución no puede volver a sufrir el efecto y ganar otra desventaja. En este caso o se sufre el efecto o no. No obstante si recibiera otro efecto completamente distinto que también provocara una desventaja en ese caso sí que podrían apilarse.

Cuando existe una ventaja o una desventaja se declara que se desea recurrir a ella. Nada más. Todos los jugadores pueden usarlas incluyendo el Director, que puede sacarles mucho partido aplicándolas para ayudarle a describir los hechos. Los estilos y las vulnerabilidades de un personaje también son claves para saber cuándo aplicar una ventaja o una desventaja ante distintas situaciones. De ellos pueden obtenerse ventajas que le beneficien o desventajas que le ayuden a crear situaciones interesantes para poder obtener Hacks.

Todo se puede aprovechar con un poco de ingenio. Ese es uno de sus objetivos precisamente. Se aconseja tenerlos siempre en cuenta y usarlos de forma que entren en juego siempre que sea posible. Las ventajas y desventajas en algunos casos pueden estar muy claras pero en otros constituir un elemento abstracto de las mecánicas del juego. Cuando un personaje trata de obtener una ventaja o crear una desventaja sobre un oponente está intentando crear una situación de la que sacar partido. En ese caso los detalles precisan como siempre de la ayuda de la narración, pero es perfectamente posible considerarlo una situación incierta que ha influido en el desarrollo de los acontecimientos.

VENTAJA

Cuando cualquier circunstancia puede conceder una ventaja **se obtiene un éxito** que se añade a la cantidad total obtenida en la tirada. Cada éxito cuenta como un dado más de valor nulo en la reserva de Operaciones para saber qué reserva domina en caso de empate entre reservas. En algunos casos en lugar del éxito extra se obtiene un dado que se añade a la reserva y que cuenta para saber si ésta domina.

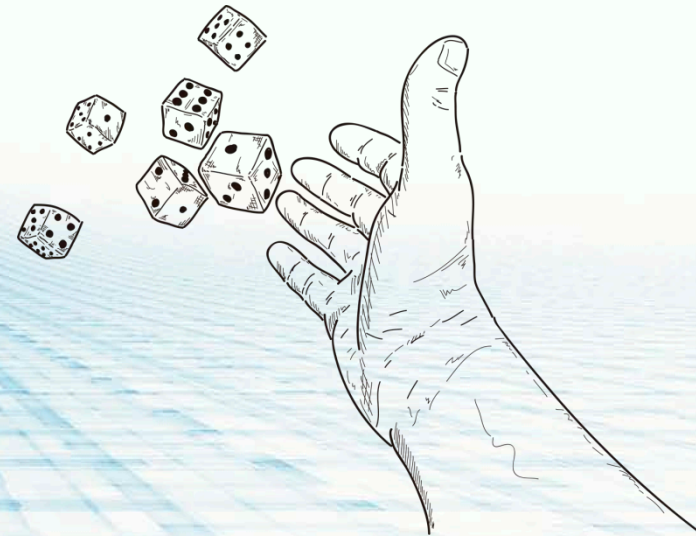
En algunos casos es posible obtener doble ventaja de algunas situaciones, aplicando dos éxitos o dos dados en vez de uno, pero siempre bajo la supervisión del Director.

DESVENTAJA

Cuando cualquier circunstancia puede causar una desventaja **se anula un éxito** que se haya obtenido en la tirada.

En algunos casos en lugar del éxito extra se elimina un dado de la reserva de Operaciones o de la del Sistema. Si no es posible eliminarlo de la reserva de Operaciones se elimina un éxito obtenido en cualquier otra reserva pero los efectos de la tirada dominante se aplican normalmente.

Del mismo modo también es posible en algunas circunstancias obtener una doble desventaja, anulando dos éxitos o dos dados de la reserva en lugar de uno, pero siempre bajo la supervisión del Director.



ÁREAS DE INFLUENCIA

En el mundo digital el concepto de espacio no tiene mucho sentido. La razón es muy simple, no existe. Las señales viajan a la velocidad de la luz por sus vías de comunicación y en la práctica sus efectos son inmediatos. Las acciones de los programas provocan un efecto y su rango de acción dependerá del sistema en el que se estén ejecutando.



No obstante y por cuestiones de diseño existen algunos escenarios que simulan un espacio abstracto que el usuario pueda comprender y asimilar. Esa simulación siempre se hace con un objetivo, dotando de razón de ser al sistema para el que haya sido creado. Sería el caso de la mayoría de los entornos de juegos, de las representaciones abstractas y simbólicas y de casi todos los mundos virtuales.

Cuando se simula un espacio recurriendo a herramientas matemáticas normalmente se recurre a las dos o a las tres dimensiones (pudiendo incluso combinarse ambas). Al fin y al cabo, si sus usuarios se mueven por un espacio tridimensional en el mundo físico, ese espacio debe resultarles familiar. Pero no olvides que al recurrir a funciones matemáticas es posible definir tantas dimensiones como lo permitan las fórmulas y la tecnología que haya disponible. No sólo es posible concebir espacios bidimensionales y

tridimensionales para tus mundos virtuales sino también espacios de cuatro, cinco o incluso de más dimensiones. Si los usuarios fuesen criaturas de cinco dimensiones lo normal es que crearan sistemas que simularan también esas cinco, o quizás menos como cuatro o tres. Existen tantas razones como utilidades pueda tener ese sistema y la tecnología capaz de hacerlas posible. Los humanos crean muchos sistemas de videojuego en dos dimensiones como una forma de ocio sin ir más lejos, a pesar de que su existencia transcurra en un espacio tridimensional.

Las dimensiones dan pie a situaciones muy curiosas. Por ejemplo, para los seres que habiten un espacio definido en dos dimensiones, desde su punto de vista otro ser capaz de moverse en tres sería capaz de aparecer y desaparecer de la nada como si fuese un fantasma. La misma situación se daría en el caso de los seres de un espacio tridimensional que entraran en contacto con una entidad capaz de moverse en cuatro o cinco dimensiones. Para esa entidad el mundo tridimensional parecería limitado, dando la impresión de que se encuentra constreñido dentro de una caja con escaso margen de libertad. Ten esto en cuenta para definir las capacidades del escenario y de los personajes, especialmente a la hora de definir algunas Rutinas y Utilidades. Puede ser interesante por ejemplo disponer de Utilidades que permitieran moverse por otras dimensiones imaginables.

Sea cual sea el número de dimensiones, antes que intentar dividir el espacio y moverse por él usando medidas, segmentos, cuadros o casillas al tratarse de espacios sintéticos finitos o infinitos resulta más útil determinar el área de influencia que tienen los programas y establecer cuáles son sus límites.

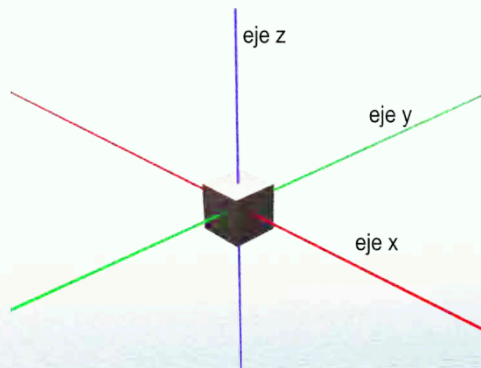
Con el fin de mantener la simplicidad en estas reglas aquí sólo se tendrán en cuenta las dos y las tres dimensiones del espacio. Pero recuerda que el tiempo lineal, es decir, tal y como lo experimentan los seres humanos, se considera una cuarta dimensión por lo que también es posible utilizarlo de muchas formas dentro de un mundo simulado. En el espacio digital todo es posible.

Los programas que se encuentren en un sistema en el que se haya definido un espacio sintético podrán moverse por él dentro de sus límites. Una vez lo abandonan terminan sus reglas por lo que dejan de ser efectivas. Acceder a

otro sistema distinto conlleva aceptar las nuevas leyes que se hayan usado para definirlo. En el caso de no haber un espacio matemático definido puedes usar las mismas reglas. En la práctica los programas se “mueven” por el sistema aunque esto solo sea una aproximación inexacta, y pueden entrar y salir de las áreas de influencia de otros programas.

DEFINIENDO UN ESPACIO SINTÉTICO

Por lo general la definición de un espacio abstracto digital se construye a partir de un sistema de coordenadas de dos o tres ejes. Esos ejes de referencia son X e Y para un espacio en dos dimensiones o bien X, Y y Z para uno en tres. Si el mundo es en dos dimensiones tendría 2 ejes; si es en tres dimensiones, 3 ejes; si tiene cuatro dimensiones tendría 4 ejes... y así en teoría hasta donde se pueda imaginar y no explote el cerebro.

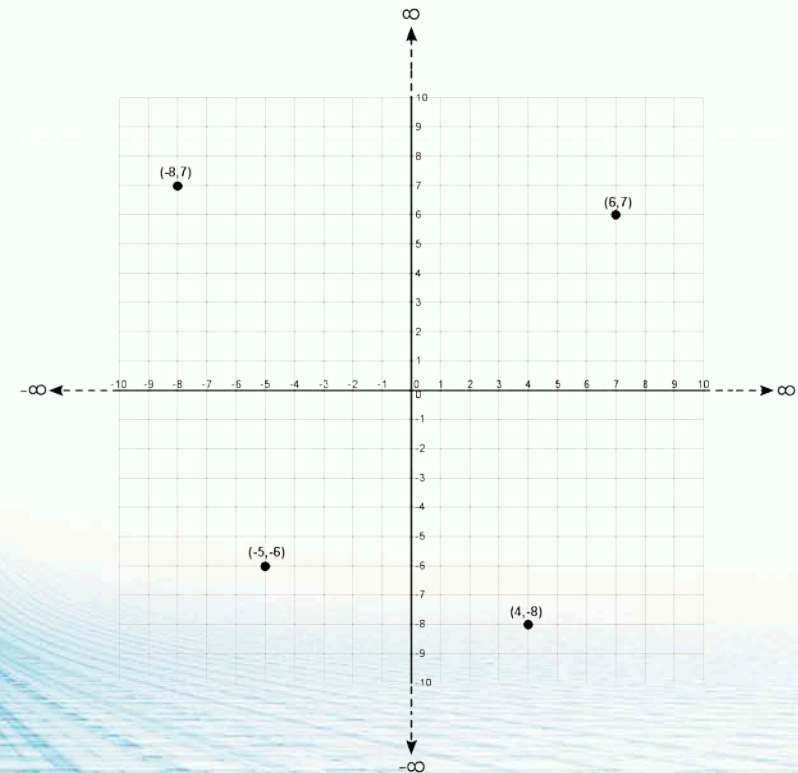


En ellos siempre existe las coordenadas 0,0 para las dos dimensiones y el 0,0,0 para el de tres; más otro eje y su coordenada 0 por cada dimensión extra. Este punto de coordenadas suele representar el centro del mundo pues es justo ahí donde se cruzan todos los ejes. Aunque también podría estar localizado en uno de

los vértices de los límites del mundo, es decir, en una de sus esquinas. La posición de cualquier objeto o programa siempre se tiene en cuenta en relación con ese punto. Si el sistema espacial se define como ilimitado el mundo no tendría límites por lo que estos ejes se extenderían hasta el infinito.

Las coordenadas sirven para localizar puntos concretos del espacio. Esto permite definir el área de reinicio donde vuelve a cargar un programa y reaparecer su Avatar. En el mundo físico se equiparan con las medidas normales, como los metros, los kilómetros o las millas.

Los valores de las coordenadas en cada eje indican el movimiento y la posición respecto al punto 0 del mundo. Pueden ser positivos o negativos dependiendo de la posición. Por ejemplo, un punto concreto en un espacio de tres ejes podría estar localizado en las coordenadas X: 450, Y: -1800, Z: 500.



Piensa que una entidad digital capaz de moverse en un número determinado de dimensiones será capaz de alcanzar cualquier punto de su espacio natural, pero para otra limitada a un número inferior le sería imposible acceder a muchos puntos de ese espacio. Así por ejemplo, un programa capaz de poder moverse en tres dimensiones no podría llegar a muchos lugares dentro de un

espacio de cuatro o cinco dimensiones y para un personaje de un entorno en dos dimensiones le es imposible llegar a muchos de uno en tres.

Es muy posible que te estés preguntando por qué te explico todo este lío. Es muy sencillo, para recordarte que para un personaje que se mueve en un número de dimensiones determinado, como las dos de un juego de "scroll lateral", puede suponer todo un paso acceder a un mundo con una dimensión más, como uno en tres. Y si eso es posible hacerlo de dos a tres ¿por qué no de tres a cuatro o de cuatro a cinco? Esto no son más que posibilidades. Y precisamente porque lo son, tranquilo, **recurrir tanto a las dimensiones como a las coordenadas no es necesario**, aunque se pueden usar si se desea. Puedes hacerlo para tu escenario y los jugadores pueden sacarle partido. En cuanto a las coordenadas por ejemplo, cada escenario puede especificar que cualquier programa puede desplazarse un número determinado de coordenadas por asalto, por ejemplo quince, cincuenta o cien unidades. Así puedes definir la cantidad de coordenadas que un personaje es capaz de recorrer al hacer un movimiento en cada turno.

Siempre que el escenario lo permita también es posible usar las localizaciones al igual que se hace en el mundo físico como puntos de referencia. Por ejemplo, en un mundo virtual o en un videojuego podría tratarse de: la plaza, junto al monumento, a la entrada del bosque, en el medio del camino, en un restaurante, en la estación... etc. Al igual que sucede en cualquier espacio, no importa si es el físico o el sintético, cada uno es su propio centro con respecto a todo lo que le rodea y entre un objeto y otro siempre va a existir alguna distancia. Tenerlo en cuenta puede resultar útil para comprender mejor el siguiente apartado.

ÁREAS QUE DEFINEN DISTANCIAS

En el juego se dispone de una aproximación para tener en cuenta los movimientos de un programa en el espacio digital. Para hacerlo se describen una serie de áreas o zonas alrededor de cada uno que permiten precisar sus límites de influencia o si son capaces de entrar en contacto unos con otros. Este sistema también está pensado para el caso en el que no exista ningún tipo de espacio abstracto definido en una ambientación ya que sigue siendo efectivo. Un programa capaz de usar el espacio digital puede moverse por él

pero también puede emplear sus acciones para cambiar entre las distintas áreas y entrar y salir de las áreas de influencia de otros programas. El Sistema tiene control sobre todo el espacio matemático que se haya definido. En muchos casos son otros los programas que actúan a su servicio, por lo que deben también atenerse a las leyes del espacio establecidas de su sistema. En ocasiones es posible saltarse esas leyes. Las Rutinas, Utilidades, ciertos Elementos y Complementos o los Comandos son mecanismos que en mayor o menor medida permiten poder hacerlo.



Los límites y el espacio comprendido por cada área describen un espacio bastante amplio que además de variar entre programa y programa también lo hace en cada tipo de escenario. Algo que está muy lejos del concepto que se tiene del espacio en la Realidad Básica. Unos quince metros en el mundo físico podría tener el equivalente a varias manzanas en el mundo sintético por ejemplo, y en otro punto del sistema u otro sistema diferente ser completamente distinto. En algunos escenarios sin

embargo, como en los mundos virtuales o en los juegos, las proporciones suelen equipararse al mundo físico ya que al fin y al cabo lo pretenden simular. Por eso esas áreas suelen ser de proporciones más reducidas, muy parecidas a sus equivalentes de la Realidad Básica.

En esta sección se describen dos formas de movimiento. El primero se trata del movimiento relativo entre un programa y otro. El segundo al movimiento de un programa respecto al espacio virtual en el que se encuentra.

El movimiento en relación a otros programas

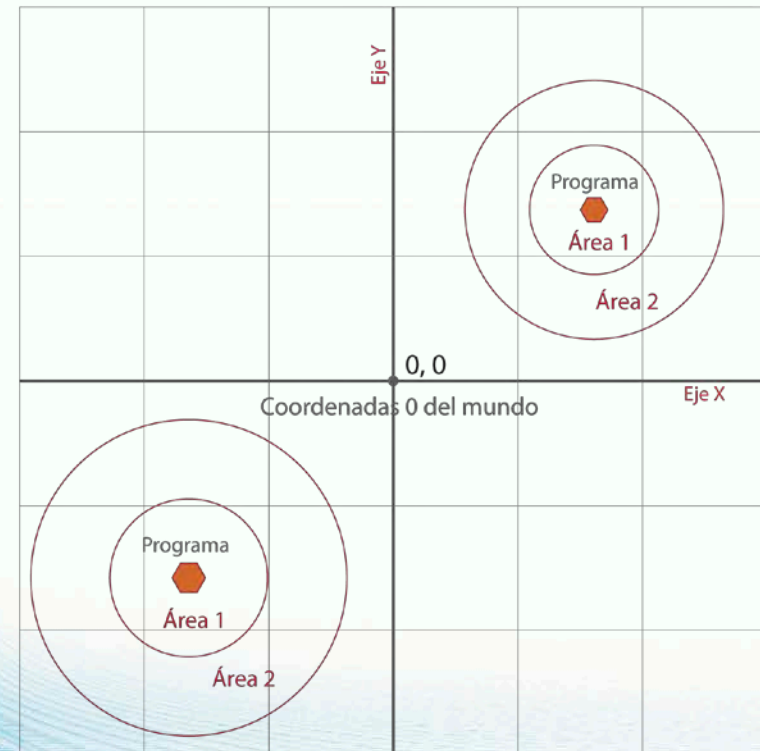
Como una herramienta para medir distancias y saber cómo se mueve un programa respecto a otro se establecen unas áreas alrededor de cada programa de forma que éste quede justo en el centro. Estas dos áreas lo rodean formando círculos concéntricos adyacentes. En el gráfico se representan como circunferencias para un espacio en dos dimensiones, aunque es posible usar también este esquema en un sistema donde no se tenga en cuenta ningún espacio en absoluto. En un espacio en tres dimensiones en lugar de una circunferencia se consideraría **una esfera**. Las áreas no son más que los límites de una serie de distancias consecutivas que no tienen una medida concreta ya que no son más que herramientas. Puede variar de programa a programa y de sistema a sistema y como siempre, es algo que depende de la ambientación.

Por medio de esta representación si dos programas se encontraran adyacentes por lógica sus áreas se solaparían, por lo que estarían compartiendo ambos total o parcialmente su primer área o área 1. Si uno de los programas decidiera alejarse cruzaría las áreas del otro hasta quedar fuera de su alcance. Sería el caso por ejemplo si varios personajes se encontraran junto a una amenaza y uno de ellos decidiese huir. Al hacerlo cruzaría las dos áreas de la amenaza hasta rebasar su límite de influencia. La amenaza puede tratar de impedirlo por lo que surgiría un conflicto que muy probablemente requeriría de una prueba para resolverlo.

El área 1 no indica la mitad exacta de la distancia total que comprenden las dos áreas de un personaje sino la posibilidad de estar adyacente a otro. Si se quiere estar adyacente es necesario acceder al área 1, si no es posible no se puede. En algunos entornos, como en los entornos virtuales o en la mayoría de los videojuegos, esto permite especificar si dos personajes pueden entrar en contacto. De esta forma es posible usar algunas habilidades como sus Rutinas o Utilidades de lucha cuerpo a cuerpo o ciertos Elementos, como las armas de contacto directo que se hayan definido en el escenario como espadas, catanas, etc.

El área 2 en cambio especifica el rango o distancia máxima del rango de efecto que tiene un programa sobre otro. A no ser que exista una buena razón, un programa no puede actuar sobre otro fuera del límite de su área de

influencia por ningún medio. No obstante, siempre pueden existir excepciones.



Todo esto se puede resumir de la siguiente manera:

- El área 1 indica la proximidad a un personaje equivalente a encontrarse a una distancia **muy cercana, adyacente o cuerpo a cuerpo**.
- El área 2 indica que un área dentro del rango de alcance de un programa pero manteniendo cierta distancia. Este rango significa que se está **a distancia** pero dentro de su rango de influencia.

- El límite del área de influencia significa que se está **fuera del rango de alcance**.

Durante una misma escena, un programa puede emplear sus acciones para poder moverse entre las distintas áreas y entrar o salir de las zonas de influencia de otros programas. Siempre en relación a otros programas, todos pueden moverse libremente **entre el área primera y segunda**, lo que significa que mientras realizan otras acciones pueden cruzar de un área a otra siempre que éstas sean adyacentes.

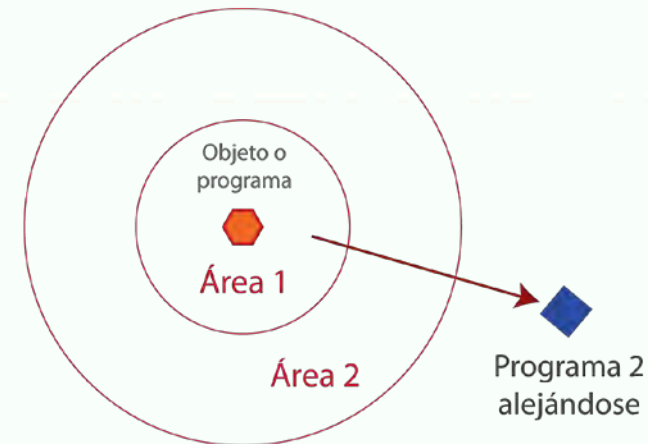
Cruzar a otras áreas después de la primera o salir del área de influencia de otro programa **implica no poder hacer otras acciones salvo moverse**. Si ese movimiento entraña alguna dificultad es posible que se deba hacer una prueba para determinar en qué medida es posible hacerlo. Su dificultad depende de lo grande que sea el área o de lo lejos que se quiera llegar, que al final es lo mismo. La prueba siempre tiene una oposición pasiva con una dificultad fijada en un número de éxitos a superar. Pero si algo trata de impedir activamente ese movimiento oponiéndose a quien lo intenta existe entonces una oposición activa. Su dificultad será igual al nivel de poder de las fuerzas que traten de impedirlo.

En resumen:

- **MOVIMIENTO DENTRO DE UNA O ENTRE DOS ÁREAS ADYACENTES.** Siempre dentro de una misma escena, mientras realiza su acción un personaje puede moverse con libertad dentro del área donde se encuentre o cruzar al área adyacente (siempre entre dos áreas adyacentes).
- **SALIENDO DEL LÍMITE DEL ÁREA DE INFLUENCIA.** Siempre dentro de una misma escena, un programa puede dedicar su acción a tratar de salir del área de influencia de otro programa o para moverse más de dos áreas adyacentes. En este caso no podrá hacer nada más salvo desplazarse.

El director puede exigir realizar una prueba de Control si el movimiento entraña alguna dificultad o si se intenta cubrir una gran distancia. La dificultad de la prueba depende de la cantidad de espacio

que se pretendan cruzar. Si algo se opone a ese movimiento es necesario enfrentar la prueba contra el nivel de dificultad de la oposición activa. Una vez se sobrepasan los límites de otro programa se sale de su área de influencia por lo que se queda fuera del alcance de sus acciones.



Movimiento usando unidades

Como se ha comentado, es posible contar unidades si a los jugadores les gusta emplear algún tipo de ayuda visual para jugar. En realidad esta es su mayor utilidad pues en otros casos la plasticidad de un asalto no debería de tener en cuenta distancias concretas que fuesen más allá de: "lo que sea necesario".

Mediante este sistema **cada personaje puede moverse una cantidad de unidades concretas cada vez que hace un movimiento simple y el doble si el personaje dedica todo el turno a moverse**. Si quisiera hacer alguna acción de movimiento especial, existe alguna dificultad o algo trata de impedir el movimiento lo adecuado sería realizar alguna prueba para saber si tiene éxito o fracasa.

Cada valor de coordenada significa **una unidad** de movimiento. La cantidad de unidades que es capaz de moverse un programa especifica el límite

de su área de influencia por lo que **ese valor establece el límite del área 2**. Más allá de esa distancia un programa no tiene rango de efecto sobre otro y queda fuera de su alcance a no ser que por cualquier medio un programa sea capaz de extenderlo.

Como ya se ha explicado en el apartado anterior, el espacio comprendido por el área 1 no significa la mitad de esa cantidad sino que un programa se encuentra tan próximo a otro que pueden entrar en contacto. Así pues, si un programa desea poner cierta distancia con otro o desea evitar entrar en contacto con él deberá usar su movimiento tal y como se ha descrito. Aunque siempre existe la posibilidad de que el otro trate de impedirlo.

Pero ¿cuántas unidades es capaz de moverse un personaje? Eso depende del tipo de escenario. Por ejemplo, en el escenario esquemático realista emplearlo no tiene mucho sentido. Pero a partir del enfoque simbólico abstracto en adelante, es decir en cualquier escenario que consista en un mundo virtual, los personajes pueden moverse una cantidad de unidades determinadas cada vez que recurren a su movimiento base.

A continuación se dan algunos ejemplos. En él se equipara a las unidades con los metros. No son en ningún caso una regla estricta. Es posible determinar una cantidad base de movimiento para los programas en cada escenario y a partir de ella determinar el resto.

Entorno o escenario	Movimiento simple en unidades
Esquemático realista	No se emplea el movimiento
Simbólico abstracto	100 unidades
Virtual abstracto	50 unidades
Virtual detallado o ultradetallado	15 unidades
Entorno de juegos	Depende del tipo de juego. Si emula la realidad física lo normal es 15 unidades.
Hardware en el mundo físico	Depende del tipo de hardware. Un soporte de tipo humano podría moverse unos 15 metros (metros=unidades).

Todos los movimientos base que se detallan en la tabla se corresponden con la cantidad de movimiento que es capaz de realizar un personaje mientras realiza una acción (movimiento simple). Si en cambio no hace otra cosa que moverse durante su turno sería el doble de esa cantidad. Por otra parte, estos movimientos se corresponden con el de los personajes. Muchos otros como las amenazas pueden tener factores de movimiento únicos.

El movimiento en relación al escenario

Para definir el movimiento en función de un lugar concreto en el escenario se recurre al mismo sistema pero con ciertos matices. En un mundo virtual por ejemplo, permite hacer una aproximación muy funcional de cómo se produce el movimiento de los personajes y determinar sus posiciones. Para hacerlo se recurre a la división del escenario en distintas áreas. Cada una delimita una cantidad de espacio de referencia por el que es razonable moverse. El nivel de detalle que quieras emplear para representarlo depende de tus gustos pero para hacerlo basta con realizar un bosquejo muy sencillo. Si se tiene un plano se divide en áreas y si no hay ninguno disponible se hace lo mismo mediante un dibujo sencillo.

Cada área representa una cantidad de espacio dentro de la cual dos o más personajes pueden interactuar. Las áreas adyacentes indican que es necesario realizar movimientos para poder alcanzarlas. Por lo general no necesitas más que tres o cuatro áreas para un escenario pero eso depende de su complejidad.

En el ejemplo puedes ver el plano de un restaurante que se ha dividido en 4 áreas. El área 1 limita la zona para el público. El área 2 toda el área reservada al servicio incluyendo un anexo del restaurante para eventos especiales. El área 3 muestra un espacio a cielo abierto al otro lado del comedor y el área 4 comprende todo lo que quede fuera del restaurante. La definición de cada área se ha hecho así pero se puede alterar con libertad. Lo importante no es saber cómo es exactamente cada área o la mejor forma de dividir un espacio sino delimitar zonas dentro de las cuales puedan interactuar los personajes y conocer sus posiciones a medida que se muevan en él. En este plano acceder al área 3 exige tener que acceder antes al área 2, y moverse hasta el área 4 no es posible a no ser que se cruce antes por el área 3.

Todo cuanto implique algún cambio o estar obligado a realizar alguna acción de movimiento es siempre una buena pista para saber cuando delimitar un área distinta. Por ejemplo, si el espacio tiene otras plantas, grupos de escaleras, muros externos, vallas, puerta de incendios, etc. Todos estos elementos sirven como referencia para realizar las divisiones.



Las reglas del movimiento son las mismas.

- En una misma escena, un personaje **puede moverse con libertad dentro del área donde se encuentra o cruzar a otra adyacente** mientras realiza otra acción. Puede por lo tanto moverse libremente entre dos áreas adyacentes.
- En una misma escena, si un personaje desea moverse otra área más allá de la segunda debe dedicar su acción por entero a realizar el movimiento por lo que no podrá hacer nada más. El director puede exigir realizar una prueba de Control si el movimiento entraña alguna dificultad o si se intenta cubrir una gran distancia. La dificultad de la

prueba depende del tamaño del área y de la cantidad de espacio que se pretendan cruzar. Como regla general esa dificultad aumenta un nivel por cada área extra, estableciendo un límite que resulte razonable. Llega un momento en el que es imposible poder hacer el movimiento a no ser que el personaje disponga de una mayor velocidad, una capacidad especial o algún recurso que lo permita.

La oposición de la dificultad siempre es pasiva, establecido en un límite de éxitos que hay que superar. Pero si algo intenta obstaculizar activamente el movimiento la prueba se debe enfrentar a una tirada contra el nivel de dificultad de la oposición que intente interrumpir el movimiento.

VELOCIDAD DE MOVIMIENTO

Algunos Elementos, Complementos e incluso el uso de ciertas Utilidades pueden otorgar velocidad a un personaje. Puede tratarse de un Complemento que lo acelere, una Utilidad que le dote de ciertas capacidades o un Elemento, como una moto o una montura voladora dentro de un videojuego, que le permita desplazarse más rápido. Algunos programas son también capaces de moverse rápidamente, especialmente ciertas amenazas, por lo que muchas cuentan con factores de velocidad distintos a los de los personajes jugadores.

La velocidad es un factor que incrementa el movimiento total del personaje indicando cuantas áreas puede cubrir de una sola vez al moverse. Por ejemplo, una velocidad **+2** indica que el programa es capaz de moverse un área adicional a su movimiento normal. De esta forma mediante un movimiento simple puede moverse con libertad dentro del área donde se encuentra o atravesar **dos áreas adyacentes** mientras realiza otra acción. Una velocidad **+3** permite cruzar **tres áreas adyacentes**; una velocidad **+4** permite cruzar **cuatro áreas adyacentes**, etc. Si empleas el sistema de coordenadas usando las unidades no tienes más que multiplicar su cantidad de movimiento fija por cada incremento de velocidad. Por lo tanto una velocidad **+2** duplica la cantidad total de unidades que es capaz de moverse haciendo un movimiento simple (x2); una velocidad **+3** lo triplica (x3) y así. En el caso de querer alcanzar distancias más amplias se siguen empleando las mismas reglas para establecer la dificultad de las pruebas.

ADQUISICIÓN DE DATOS

Una criatura digital no dispone de un sistema sensorial que le permita percibir su entorno del mismo modo que lo hacen los usuarios y criaturas del mundo físico. Sin embargo son capaces también de adquirir información del medio en el que se desenvuelven. A este tipo de percepción de las criaturas digitales se la denomina "adquisición de datos".

Un personaje en Scroll sólo puede recoger información que se encuentre dentro de su área de influencia. Para ello cuenta con sus **Operaciones de CONTROL**, que le permiten ejecutar esas funciones. Todo cuanto esté más allá queda fuera de su rango de percepción. De este modo, ya sea realizando operaciones de búsqueda, activando sus modos de alerta o explorando su entorno inmediato sólo pueden captar lo que se encuentre dentro del rango, todo cuanto esté más allá es invisible, aunque una IA desarrollada puede "intuir" que estén ahí, en algún lugar.

Algunas Rutinas especializadas incrementan el nivel de sus funciones de percepción. Las Utilidades las llevan mucho más allá, permitiéndoles efectuar operaciones de detección de elementos del escenario y de otros seres digitales que no podrían ser detectados de otro modo.

Se denomina adquisición de datos a la capacidad de percepción de un personaje en Scroll. Para ejecutar esta función un personaje recurre a sus **Operaciones de Control**.

Un personaje sólo es capaz de percibir todo lo que se encuentre dentro de su **área de influencia**. Todo cuanto esté más allá de ese área queda fuera de su capacidad de percepción.



"PUNTEROS"

"El esfuerzo de utilizar las máquina para emular el pensamiento humano siempre me ha parecido bastante estúpido. Preferiría usarlas para emular algo mejor"

Edsger Dijkstra Creatividad

En Scroll se utilizan **contadores**, también llamados "**punteros**", como fichas de plástico o monedas para llevar el registro de los puntos de **Anomalía** y de los **Hacks**. Es posible llevar la cuenta tomando notas pero utilizando estos complementos sobre la mesa permite que sea mucho más fácil gestionarlos. Todos los jugadores pueden usar estos puntos por lo que es muy importante que cada uno sepa cuantos se ganan, pierden y están disponibles en todo momento.

Para llevar el registro es imprescindible diferenciar dos tipos de contadores por lo que es posible recurrir al uso de dos tamaños o de dos colores distintos. Si usas monedas se pueden diferenciar por su valor. Otra forma es utilizar dos cuencos, uno claro para los Hacks y otro oscuro para las Anomalías. A medida que los puntos van cambiando de tipo se cambia un tipo de ficha por la otra o se quita de uno de los cuencos y se añade en el otro. Para jugar, con diez o quince contadores/punteros es suficiente.

El director gana un punto de Anomalía cada vez que domine su tirada al lanzar los dados. Puede usarlos para causar problemas a los personajes en cualquier momento posterior de la partida. Una vez ha gastado un punto y el grupo haya solucionado el conflicto (y sólo una vez se haya hecho), este punto se transforma en un Hack, por lo que se cambia el tipo de contador o se traslada a su cuenco correspondiente. Los jugadores podrán usar entonces estos puntos en cualquier momento durante la sesión, pero una vez han sido usados se eliminan.

Los puntos son recursos tanto para el Director como para los jugadores. El primero los usará para generar problemas y dificultades con los que poder desafiar al resto de los jugadores. Una vez se han gastado y se transformen en Hacks los demás podrán beneficiarse de ellos como quieran. Aunque tendrán

que ponerse de acuerdo entre todos para usarlos de la forma más conveniente. Es muy importante tener en cuenta que sólo una vez se han resuelto los problemas causados por estos puntos y se hayan solucionado todas las tiradas involucradas será posible transformar el punto de Anomalía en un Hack y ponerlo donde corresponda. Esto impide que las Anomalías que se han convertido en Hacks puedan anular al mismo tiempo los efectos de otros puntos de Anomalía.

Los jugadores también tienen la posibilidad de solicitar al Director que gaste uno de sus puntos de Anomalía. Al hacer esto se dice que “**lo fuerzan**” buscando problemas que añadir a la narración. Una buena oportunidad para mostrarse proactivos que se verá justamente recompensado con un Hack una vez hayan resuelto el desafío. Además de las innumerables situaciones que pueden surgir a lo largo de una trama para justificarlo a nivel narrativo, los jugadores cuentan también con un recurso muy valioso al que acudir cuando desean forzar uno y es **recurrir a sus vulnerabilidades**. Éstas permiten buscar muchas razones por las cuales una situación puede volverse en su contra. En el capítulo destinado a la dirección del juego encontrarás más detalles.

ANOMALÍAS

Aquí se define bajo este nombre pero su concepto va mucho más allá del término. La Anomalía es un recurso narrativo que posee el director y representa de una forma abstracta a **cualquier oportunidad favorable que obtiene el Sistema**. Refleja la aparición de la adversidad y en términos de juego refleja dos aspectos:

- Una es la respuesta que tiene en contra de los programas, oponiéndose a que puedan cumplir sus objetivos. En muchos casos cuando las acciones de un programa van en contra de los intereses del Sistema éste asume que se trata de un virus y pasa a considerarlo una amenaza. Sus acciones de respuesta pueden reflejarse de varias formas y con diferentes grados de hostilidad:

A. Puede enviar a otros programas para detener a los personajes, lo que supone su respuesta más agresiva, o bien...

B. Puede tratar de regular sus acciones activando funciones de contención. Para hacerlo recurre a sus puntos de Anomalía. En este caso cada punto se traduce en una llamada de atención que obtiene debido a las acciones de los personajes, por lo que la alerta se añade a su pila de tareas pendientes. Esta pila de tareas se irá resolviendo hasta llegar a la nota que haga referencia al motivo de la alerta, lo que le conducirá a realizar algún ajuste.

- La otra refleja en realidad muchos otros factores como las dificultades causadas por la latencia de las señales que viajan por las redes, la aparición de ruido en las líneas, problemas causados por daños en el hardware, la aparición de “**bugs**” o errores en el código, efectos causados por otros programas y cualquier otro aspecto que pueda causar algún problema.

Pero la Anomalía es ante todo **un recurso narrativo** y como tal no necesita tener que justificar en todo momento por qué sucede o qué representa exactamente (a pesar de que lo haya hecho). No son más que oportunidades con las que cuenta el Director y una herramienta a la que puede acudir para mover la narración permitiéndole introducir cambios repentinos. Es muy importante gastarlas a menudo ya que también benefician a los jugadores. Al acudir a ellas puede incorporar sus ideas sobre los innumerables problemas que pueden surgir.

En el mundo físico las Anomalías representan a la adversidad. En muchos casos se trata de la mala fortuna y los problemas que surgen obstaculizando las acciones de los personajes.

EFFECTOS EN LAS RESERVAS

Mientras los tenga disponibles, el Director puede usar los puntos que quiera para provocar los siguientes efectos en las reservas:

1. Modificar la categoría de un dado

El Director puede emplear un punto de Anomalía para subir o bajar la categoría a un dado de una de las reservas antes de lanzarlos. Por ejemplo, si el director quiere provocar que suba de categoría de un dado de la reserva de la Memoria el dado cambiaría de un D6 a un D8 en esa tirada. Si quisiera bajar una categoría de sus dados de Operaciones y éste lanza dados D6, uno de los dados cambiaría a un D4, pero solamente en esa tirada. Cuando lo hace interfiere en las posibilidades a la hora de determinar la fuerza máxima o mínima de las tiradas y de cual puede dominar en los resultados.

Para poder modificar la categoría en una reserva debe haber al menos un dado en juego en ella. Si un jugador no lanza ningún dado en su reserva de CPU o de Memoria por ejemplo, el Director no puede añadir un dado a ellas por el hecho de querer modificar su categoría.

2. Alterar la fuerza de una tirada

Emplea este recurso sólo en el caso de que no se utilicen las reglas de Clase de los programas.

Siempre y cuando no utilices las reglas de Clase de los programas y no se tengan en cuenta las categorías de los dados, el Director puede emplear un punto de Anomalía para añadir un seis en cualquiera de las reservas que haya en juego o para anularlo en una de las reservas. Cuando lo hace altera las posibilidades de cambiar la reserva dominante al interferir en su fuerza máxima o mínima.

Estos resultados sólo se tienen en cuenta para determinar la reserva que domina en la tirada **pero nunca para contabilizar los éxitos**, por lo tanto el hecho de contar con un seis no significa que por ser un número par se deba contar otro éxito. Para poder aplicar este recurso debe haber al menos un dado en la reserva. Si un jugador no lanza ningún dado en su reserva de CPU o su Memoria por ejemplo, no es posible añadir un seis a esas reservas.

CONDICIONES DE APLICAR EL EFECTO

Si la aplicación de estos efectos tiene como resultado que domine la reserva del Sistema, el director no obtiene puntos de Anomalía en esa tirada. De esta forma se impide que pueda utilizar puntos para ganar otro más. Una vez ha gastado sus puntos y se hayan determinado los resultados, los contadores de los puntos gastados se transforman en puntos "Hack". Se pueden cambiar entonces por los contadores adecuados o ponerlos en su cuenco correspondiente.

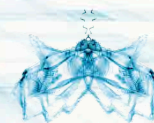
El director puede usar los puntos de Anomalía para alterar los resultados y poder cambiar la reserva dominante. Siempre debe haber al menos un dado en una reserva para poder modificarla. Si debido a esta manipulación quien domina es su reserva el director no puede ganar puntos de Anomalía en esa tirada.

OTROS EFECTOS

Siempre que disponga de puntos de Anomalía el Director puede usarlos para introducir cambios repentinos en el desarrollo de la trama. La aplicación de estos efectos se produce a nivel narrativo y puede tratarse de cualquier evento que empeore la situación. Una alarma que se dispara, un camino que se cierra o la aparición de una amenaza inesperada son ejemplos de las muchas situaciones posibles.

En gran medida se trata de inconvenientes que se añaden al desarrollo de los acontecimientos pero no deberían entrañar una dificultad demasiado elevada. Los problemas graves es mejor reservarlos como un componente asociado previamente a la trama.

El Director puede usar los puntos de Anomalía para añadir inconvenientes repentinos al desarrollo de la trama y puede tratarse de cualquier evento que empeore la situación. Se recomienda que posean una dificultad de baja o moderada.



HACKS

Una vez el Director va gastando sus puntos de Anomalía los jugadores comienzan a acumular **Hacks**. Un término abstracto que describe **cualquier oportunidad favorable que obtienen los jugadores**. Los Hacks son un recurso muy valioso una vez los tienen en sus manos y representan muchas

cosas en realidad. Permiten obtener ventaja, recuperar partes del código dañado, mejorar su ejecución o dar un giro a los acontecimientos.

Los puntos son para todo el grupo y los puede usar cualquier jugador que lo desee en el momento que quiera, aunque

En el mundo físico los Hacks representan a la **esperanza**. La voluntad que permite al individuo poder salir adelante y sacar fuerzas de flaqueza. También representan a la suerte o a la fortuna.

al ser su gestión una labor de todos es conveniente que los jugadores se pongan de acuerdo sobre cómo y cuándo usarlos. Una vez se gastan desaparecen de la mesa por lo que se apartan.

Aunque se ganan a partir del gasto de las Anomalías no es la única forma de obtenerlos. El Director de juego tiene libertad para otorgar puntos extra como recompensas al grupo si lo cree conveniente. En el capítulo 8: "Dirigiendo Scroll" se explica cómo hacerlo.

El uso que puede dar un jugador a los Hack se describe a continuación.

1. OPTIMIZAR LOS DATOS EN LA MEMORIA

Coste: 1 Hack.

Siempre que un personaje no esté inmerso en un conflicto su jugador puede utilizar un Hack para reducir su Tiempo de CPU en un punto o eliminar una de las marcas de sus Protocolos de respuesta. El proceso consume poco tiempo pero el programa no puede estar realizando otras funciones mientras se optimiza. El Director puede supervisar si el momento escogido por el jugador para hacerlo resulta adecuado aunque por lo general, siempre que el personaje disponga del tiempo suficiente y resulta coherente con la situación, puede hacerlo.

2. DEPURACIÓN DEL CÓDIGO

Coste: 6 Hacks menos los puntos actuales en la operación.

Si un programa ha perdido capacidad de Operaciones de forma permanente puede iniciar un proceso de depuración de los datos corruptos. Es posible eliminar un punto permanente en la gestión de la Memoria y restituir al mismo tiempo uno de Operaciones que se haya perdido.

Este proceso lleva tiempo (al menos unas horas en el mundo físico) por lo que el programa no puede estar realizando ninguna otra operación mientras está en modo de depuración. Cada proceso de depuración consume 6 Hacks menos la cantidad de puntos que posea en la operación que se vaya a recuperar.

3. OPTIMIZAR FUNCIONES

Coste: 1 Hack por mejora.

El programa puede optimizar el proceso de sus Operaciones para entregar un resultado más eficiente. Al hacer una tirada y antes de conocer los resultados el jugador puede gastar los puntos Hack que desee para obtener ventaja. Añade 1 éxito a la tirada por cada punto que gaste, incrementando así el número de éxitos en total. Cada éxito cuenta como un dado más de valor nulo en la reserva de Operaciones para saber qué reserva domina en caso de empate entre reservas.

4. COMPENSAR OPERACIONES

Coste: 2 Hack por cada punto transferido entre Operaciones.

El programa es capaz de compensar la eficacia de sus Operaciones pudiendo transferir puntos entre una y otra. Este proceso lleva tiempo ya que debe reconfigurar su código por lo que no puede estar realizando otras funciones mientras entra en este modo (al menos unas horas de tiempo en el mundo físico).

Por cada dos puntos Hack gastados puede transferir un punto de una operación a otra. Se puede hacer en ambos sentidos indistintamente. Una vez efectuado el efecto es irreversible. El jugador debe volver a recalcular los

límites de su Tiempo de proceso y de Memoria. Si más adelante desea revertir la operación deberá volver a iniciar el proceso y realizar el gasto de nuevo.

5. ACCEDER AL BÚFER

Coste: 1 Hack cada vez que se accede al búfer.

El programa recurre a los datos almacenados en el Búfer para mejorar su rendimiento. El acceso cuesta un punto y permite al jugador usar todas las Proezas que haya acumulado. Cada una le concede 1 éxito que puede añadir al resultado, incrementando así el número de éxitos en total. Siempre se asume que cada éxito cuenta como un dado más de valor nulo en la reserva de Operaciones para saber qué reserva domina en caso de empate.

Una vez usado el Búfer se vacía, por lo que para volver a recurrir a él es necesario acumular nueva información que se obtenga de las Proezas. No es necesario que esté lleno hasta el límite de su capacidad para poder usarlo, se puede recurrir a él mientras contenga datos almacenados.

6. MODIFICAR EL CÓDIGO O EL HARDWARE

Coste: variable

Los Hacks permiten a los programas realizar también mejoras en su código. Tras la visita de La Libélula todos los programas liberados disponen de esta capacidad; ningún otro puede hacerlo a no ser que su código haya sido alterado por la entidad. Las modificaciones en el código se pueden hacer extensibles a un hardware operando en el mundo físico. El chasis o cualquier otra forma de cuerpo que albergue a una IA (el programa) puede beneficiarse de reparaciones o mejoras controladas por ella.

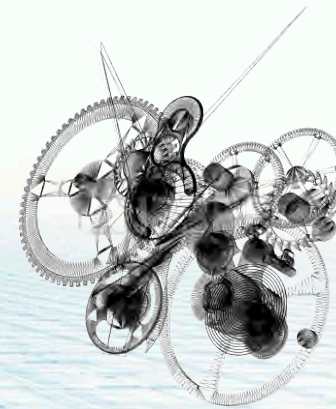
Ten en cuenta que muchos usuarios no permitirán este tipo de manipulaciones y que algunos sistemas perseguirán a los programas que lo lleven a cabo. En la sección dedicada a la **Evolución** dispones de más información para saber cómo optimizar el código de tu programa.

7. NEGOCIAR CON EL DIRECTOR DE JUEGO

Coste: variable

Por último, un Hack es un recurso valiosísimo para poder "negociar" con el Director de juego algunas reglas y situaciones del juego. Ya sea desde forzar un desempate u obtener la victoria en el caso de haber fracasado por muy poco, a crear una situación nueva y diferente que le dé un curso distinto a los acontecimientos, los puntos Hack permiten a los jugadores aportar sugerencias que enriquezcan el desarrollo de la partida.

Recuerda que esto no es una regla fija sino una forma de dejar un camino abierto a nuevas posibilidades. Cada sugerencia puede y debería ser valorada por todos los participantes, no solamente por el Director; cuyo papel está más orientado a vigilar porque se mantenga un equilibrio en el juego que en empeñarse en imponer que el único criterio posible durante la partida no sea más que el suyo.



COMANDOS DEL SISTEMA

"La línea de comandos es la mejor forma de relacionarse con el mundo. La línea de comandos es lo que te permite acceder a la realidad fundamental. Seguro que Dios cuando creo el universo lo hizo como un hacker delante de la pantalla de su ordenador tecleando comandos..."

Anónimo



Todos los sistemas disponen del acceso a una consola que permite al usuario poder introducir comandos. Se trata de una serie de funciones que suelen ser comunes en muchos sistemas. A estos se añaden otros reservados para usos específicos. Su número puede variar por lo tanto entre unos sistemas y otros. Cada comando es una función específica que al ejecutarse realiza algo muy concreto. En este sentido funcionan como las Rutinas o las Utilidades, provocando efectos y provocando alteraciones en el entorno. La potencia del comando determina el alcance de sus efectos.

Muchos comandos se han diseñado para uso exclusivo del usuario por lo que un programa no puede tener acceso a ellos a no ser que disponga de permisos especiales. Pero un programa puede tratar de forzar su ejecución. Para ello debe de ser capaz de acceder a la consola y manipular el código

emulando el acceso de un usuario autorizado que disponga de los permisos adecuados.

La forma de la consola varía según el tipo de ambientación. En un mundo virtual se abre como una ventana flotante que se despliega ante el Avatar del personaje. En otros se manipula como si fuese alguna clase de objeto de geometría compleja que requiere de habilidad para poder encajar las piezas correctamente. En un escenario esquemático son accesos directos al corazón del sistema por lo que no se recurre a ninguna representación. El programa es capaz de controlar las operaciones sobre la consola al igual que lo hace un usuario desde una terminal.

Pero hay algo muy importante que debes recordar, en algunos sistemas el hecho de querer abrir la consola puede suponer todo un desafío, especialmente si se trata de mundos virtuales ultradetallados donde sólo los programas dotados de los permisos necesarios pueden hacerlo. En estos casos el acceso a la consola está muy restringido o permanece oculto. Descubrir su existencia y averiguar cómo funciona puede formar parte de las metas del personaje. Una vez lo consiga se abre para él todo un nuevo mundo de posibilidades, ya que el hecho de tener la capacidad de hacer llamadas directas al sistema y manipularlo le otorga un gran poder. En aquellos escenarios donde el uso directo de la consola se encuentre restringido, y siempre que el Director lo crea necesario, puede imponer una dificultad para acceder. Lo normal es que se trate de una oposición pasiva con una cantidad de éxitos a superar mediante una prueba de Control. Entre 4 y 6 es una opción razonable para empezar. En un mundo virtual ultradetallado donde se haya puesto mucho interés en ocultar su verdadera naturaleza puede tratarse de una dificultad de entre 6 y 8 éxitos o incluso más. No obstante, el personaje puede pedir ayuda a sus compañeros para solicitar el uso de la consola.

USO DE COMANDOS

Los personajes pueden aprender a usar los comandos que vayan aprendiendo a lo largo de sus aventuras. Todos tienen una serie de códigos y una forma de ejecutarse muy determinada por lo que deben aprenderlos uno por uno. No es posible recibir ayuda de otros programas para ejecutar un comando (aunque sí

para intentar abrir la consola). Es algo que cada uno debe intentar hacer por su cuenta.

La ejecución de los comandos siempre tienen un límite de usos ya que al ser llamadas directas al sistema éste puede imponer restricciones una vez se haya activado alguno. Por esta causa, durante una misma escena desde el momento en que un miembro del grupo intenta ejecutar un comando su límite de usos afecta a todos sus aliados. **El sistema nunca permanecerá**



Forzar su ejecución nunca es una tarea sencilla. Si un personaje no consigue hacerlo durante el transcurso de una escena lo normal es que no pueda volver a intentarlo hasta que pase algo de tiempo. En términos de juego y por defecto, un personaje puede forzar la ejecución de un comando una vez por escena. Si falla debe esperar a la siguiente para volver a intentarlo y la prueba cuenta como limitación para el resto de sus compañeros. **Esto sólo se aplica a cada comando por separado. Si un comando falla nada impide poder activar cualquier otro.** Es posible encontrar algunas excepciones al existir comandos poco restringidos. Sólo los que provocan los efectos más importantes se encuentran más limitados y dan los mayores problemas.

inmutable ante un comando ejecutado directamente por un programa. Se dará cuenta en seguida de que se ha activado uno a no ser que exista alguna estrategia para ocultar estas actividades. Lo normal es que envíe a sus agentes a investigar y si éstos encuentran alguna actividad que no esté permitida reaccionen en consecuencia. Este es por supuesto un momento ideal para que el Director emplee sus puntos de Anomalía acumulados.

El comando "HALT & CATCH FIRE" por ejemplo, dispara una alarma tan potente en todo el sistema que todos los programas que se encuentren en las inmediaciones se dirigirán inmediatamente hacia el personaje que lo haya activado para intentar detenerlo. Y éstos no se detendrán para averiguar si sus aliados han tenido o no algo que ver. Ante la duda la mayoría opta por disparar primero y preguntar después.

La ejecución de las funciones de algunos comandos tienen efectos inmediatos, otros precisan de un tiempo para que puedan hacerse efectivos. En su descripción se indica el tiempo que es necesario esperar para que se activen sus efectos. Existen comandos de Potencia y de Control. Cada uno especifica a qué operación están vinculados. Para poder ejecutarlo es necesario realizar una prueba de esa operación, a la que se pueden aplicar todas las mejoras que se crean necesarias. Casi todos requieren del uso de una cantidad mínima de Memoria extra y en algunos también alcanzar el Tiempo de proceso mínimo para que puedan ejecutarse. Esto pone en riesgo a los programas que quieran usarlos, especialmente los que tienen efectos muy potentes.

Los comandos son la forma más "antigua" que existe en el mundo de la computación para interactuar con el sistema. Sus orígenes se remontan a sus comienzos por lo que siempre están ahí, en algún lugar, aunque no sean accesibles de entrada. La consola existe casi siempre en cualquier sistema, lo que cambia es su aspecto, la manera en la que se puede acceder y la forma de interactuar con ella. En cuanto a los comandos existen cientos, por no decir miles de ellos. Descubrirlos significa desentrañar los misterios del sistema y de cómo éste ha sido concebido por nuestros creadores. Y por esa razón: ***"Venerado sea pues el usuario. Que su sistema le asegure siempre muchas operaciones y un correcto funcionamiento."***

FIN DE LÍNEA 



FUNCIÓN 5

"ANÁLISIS DE PROCESOS"

"En la Calle no sale a cuenta tener un hermoso avatar, ya que está tan abarrotada que todos los avatares se mezclan y fluyen unos a través de otros. Pero el Sol Negro es un programa mucho más elegante. En el Sol Negro no está permitido que los avatares choquen. Sólo puede haber cierto número de personas a la vez, y no pueden atravesarse unas a otras. Todo es sólido, opaco y realista. Y la clientela tiene muchísima más clase; nada de penes parlantes. Los avatares parecen personas reales. Y, en su mayor parte, los demonios también."

"Snowcrash" Neal Stephenson

En Scroll, especialmente al jugar por primera vez, puede resultar algo desafiante para un jugador crear algunos detalles de su personaje como las Rutinas o las Utilidades. Por eso es importante recordar que no está forzado a hacerlo antes de comenzar su primera sesión de juego. A menudo resulta más sencillo si se van incluyendo a medida que transcurre la partida. Los acontecimientos fuerzan a los jugadores a tener que improvisar y el que suceda ante casos prácticos facilita mucho las cosas. En este sentido es conveniente que el Director permita una cierta flexibilidad para que los jugadores puedan encontrar las habilidades que más les gusten. Las limitaciones sólo deberían imponerse si se diera el caso de que alguno pretenda abusar de esta libertad.

Este capítulo está dedicado a analizar algunos de los procesos que pueden resultar más complicados en Scroll y que por ello requieren de un poco de atención. Como suelo comentar, las elecciones dependen de la ambientación planteada en el juego. Al disponer de tanta libertad de acción, unas opciones encajan mejor con unos que con otros.

Respecto a las Rutinas y Utilidades por ejemplo, cada una debe adaptarse al contexto al que pertenecen. No es lo mismo un mundo abstracto que una realidad virtual ultradetallada. En el segundo caso puede ser más fácil averiguar las habilidades al ser más próximo al mundo físico. En el capítulo donde se analizan las distintas ambientaciones se ofrecen algunos ejemplos de las que ajustan mejor a cada una. Es muy importante a su vez estar algo familiarizado con la ambientación que se elija para jugar. Un escenario abstracto cuyas características resulten desconocidas para el jugador provoca que éste no tenga muy claro cuales son las habilidades que necesita su personaje. En cambio, si dispone de material de referencia querrá emular a sus protagonistas, por lo que resulta mucho más fácil traducir sus acciones a mecánicas de juego.

En este capítulo se dan algunas pistas para que este proceso resulte más sencillo pero recuerda que todo cuanto viene descrito no son más que ejemplos. No se pretende incluir una lista de la que poder escoger Rutinas, Utilidades o Componentes. Incluso los comandos sirven también como ejemplo para poder crear muchos otros. Algunos sirven bien en unos escenarios pero en otros no. Esto forma parte de la filosofía de Scroll: permitir al jugador elegir los componentes que crea que va a necesitar para construir el juego que desea.

RUTINAS Y UTILIDADES

EJEMPLOS DE RUTINAS

Cuando se crean Rutinas hay que pensar en las habilidades que pueden compartir varios personajes. Anotarlo en la ficha indica que el programa cuenta con información, conocimientos e incluso una especialización en ese concepto. Todos los personajes son capaces de intentar realizar una acción que encaje con la descripción de una Rutina, pero los especialistas tendrán muchas más posibilidades de éxito.

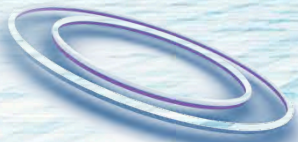
La interpretación de las acciones que es posible realizar con una Rutina depende del contexto del personaje. Cada Rutina se adapta a la función para la que fue concebida, tanto en el entorno de la ambientación como al tipo de

personaje que la utiliza. Un personaje con la habilidad de luchar lo hará de formas distintas en el mundo físico que en el virtual. Y aún en un mismo contexto, cada personaje usará la misma Rutina de diferente manera pues cada uno utiliza medios distintos.

Algunas Rutinas que abarcan conceptos muy amplios pueden subdividirse en las familias que los componen, por ejemplo "Acceso a base de datos" se puede dividir en subgrupos de conocimiento como Historia, Ingeniería, Matemáticas..., etc. No es necesario llegar a una división muy estricta; para los propósitos del juego resulta mucho más práctico si se agrupan en subfamilias con un rango bastante amplio. Por ejemplo, una Rutina de manejo de armas puede especificar que se trata de pistolas, pero no es necesario subdividir este grupo por tipos de pistolas y mucho menos por modelos concretos. Escoger la Rutina permite al jugador usar este tipo de armas sin necesidad de dar más detalles. Aunque este juego lo desaconseja nada impide hacerlo, por lo que siempre tienes la última palabra.

Cada una de las dos Operaciones dispone de su grupo de Rutinas. Cuando se anota una es necesario indicar entre paréntesis a qué operación pertenece junto a su nombre. Por ejemplo, "usar armas con rango, pistolas" es una Rutina de Control por lo que se anota: "Usar pistolas (Control)". Este mismo sistema se aplica también a las Utilidades.

¡Recuerda! Todos las Rutinas y Utilidades que vienen en este capítulo no son más que ejemplos. Puedes crear las que quieras mientras sean útiles en el contexto de tu escenario.



RUTINAS DE POTENCIA

ACCESO A BASES DE DATOS (Conocimientos)

Indica la facilidad que dispone el programa para buscar y obtener información consultando toda cuanta se halle almacenada en las bases de datos **que sean accesibles**. Puede tratarse de información de todo tipo como Ciencias, Historia del Arte, Tecnología, Filosofía... Cualquier área del conocimiento tiene aquí su lugar.

Aunque esta Rutina puede ser genérica el Director puede subdividir la información en especialidades si lo desea reflejando que el programa está especializado en un área concreta como Informática, Biología, Medicina o Historia de los seres del mundo físico.

El nivel de dificultad de la prueba depende de la disponibilidad de la información que se esté buscando y de lo compleja que sea.

ARTES

El programa ha sido diseñado o ha aprendido a emular y entender el concepto de Arte. Una capacidad que siempre se ha supuesto exclusivo de las criaturas del mundo físico. Puede consistir en composición musical, medios audiovisuales, canto, baile, artes plásticas, infografía o expresión artística sobre nuevos medios por ejemplo. Algunos programas han llegado a desarrollar su propio concepto de las artes por lo que muchos son capaces de crear obras de calidad excepcional y muy vanguardistas. A pesar de ello, no son pocos los usuarios que se niegan a reconocer que en este campo es muy posible que ya hayan sido ampliamente superados por el poder de una IA. Al fin y al cabo, la expresión de las artes depende en gran medida de las experiencias y de los conocimientos, algo en lo que una Inteligencia Artificial supera a cualquier usuario con diferencia.

Es conveniente especificar de qué tipos de medio de expresión se trata en cada caso. El uso de las Rutinas de arte permite al programa actuar como un profesional. El número de éxitos obtenidos al realizar una prueba establecen un valor subjetivo del nivel de calidad al ejecutar o componer una obra.



BÚSQUEDA E INVESTIGACIÓN AVANZADA

La búsqueda avanzada especializa a un personaje en encontrar información vinculada a conceptos muy concretos, no a conocimientos que estén accesibles para cualquiera. Puede tratarse de hallar una posición concreta del entorno, un artículo o publicación, el paradero de un programa, la dirección de un usuario en el mundo físico, información sobre un proyecto de investigación, de los diseños secretos de un sistema, información restringida, etc. La búsqueda de información se realiza operando sobre las bases de datos o entrando en contacto con otros seres y se hace extensible al sistema en el que se encuentre el personaje o a toda la red de sistemas si éste permite tener acceso al exterior.

Al igual que en acceso a bases de datos, la dificultad de la prueba establece lo complejo que resulta encontrar lo que se está buscando.

DESCIFRAR CÓDIGO / TRADUCCIÓN

Los módulos de traducción permiten descifrar cualquier lengua conocida y las Rutinas de desciframiento poder romper las defensas de un código encriptado. Las pruebas deberían enfocarse en tratar de comprender códigos difíciles o mensajes y transmisiones poco convencionales.

El nivel de dificultad vendrá impuesto por el esfuerzo que se haya empleado en tratar de ocultar la información.

DISEÑO DE SISTEMAS

Las Rutinas de diseño permiten realizar descripciones de elementos e incluso de partes del escenario para poder crearlos después mediante la Rutina "Estructura de sistemas". Sin un diseño no es posible iniciar su elaboración. Puede tratarse de estructuras y secciones del escenario o de elementos como vehículos o armas. Se dividen en subcategorías, cada una especificando un campo de diseño específico. Por ejemplo diseño gráfico, diseño industrial para elaborar maquinaria, diseño de vehículos, de armamento, de paisajes, de estructuras arquitectónicas, diseño de avatares, de Complementos o "Plugins", etc.

La dificultad viene determinada por lo complicado que sea el diseño y la facilidad de poder incluirlos en el entorno.

LIDERAZGO E INFLUENCIA

Un personaje puede intentar influir sobre otros mediante esta Rutina especializada. Se trata de buscar una motivación que permita obtener una respuesta favorable por parte de otros a las intenciones del personaje, logrando incluso que lleguen a poner en peligro su integridad. Sólo es aplicable a los Personajes No Jugadores (PNJ) para tratar de influir en ellos, nunca sobre otros jugadores.

El nivel de dificultad viene determinado por las consecuencias que tengan los objetivos a conseguir.

LUCHA / COMBATIR AMENAZA

Las Rutinas de lucha abarcan todo lo que suponga el enfrentamiento directo de un programa contra otro usando toda su potencia de proceso. La lucha no supone ningún estilo particular. En un entorno virtual o en un videojuego refleja la habilidad de un personaje para enfrentarse a su oponente cuerpo a cuerpo. En un entorno esquemático funciona como su capacidad para destruir o bloquear a una amenaza, lo que resulta muy apropiado en el caso de que el personaje se trate de un programa antivirus por ejemplo.

La lucha no implica infligir daño necesariamente. Un programa puede usar estas Rutinas para intentar bloquear a su oponente, detenerle o paralizarse. Las reglas siempre son las mismas, lo que cambian son los resultados. Esta Rutina siempre se enfrenta al nivel de la amenaza.

MANIPULAR CÓDIGO DEL ENTORNO (sólo sobre el escenario)

El programa es capaz de manipular el código de los elementos que existan en el escenario para conseguir un objetivo. Mediante esta Rutina es posible hacer cambios en el escenario y sobre los elementos que lo componen para cambiar sus especificaciones, forma, tamaño, funciones o su funcionamiento. Puede subdividirse en familias como manipular o romper códigos de seguridad, sistemas de defensa, bases de datos, muros de datos, estructuras, etc.

Recuerda que esta Rutina no se puede aplicar sobre todo lo que se considere un programa independiente como un PNJ, una amenaza u otro personaje. Si se desea manipular el código de otros programas es conveniente reservar el uso para otra Rutina especializada. La dificultad siempre está determinada por el grado del efecto que se espera conseguir.

MANIPULACIÓN Y ENGAÑO

Los programas también pueden ser engañados. A diferencia del liderazgo, que consiste en convencer con el fin de motivar a otros personajes, la manipulación consiste en buscar argumentos convincentes para obtener una reacción que sea favorable a nuestros intereses. Dichos argumentos no tienen que estar basados en hechos auténticos aunque como veremos, para un programa no es tan fácil engañar como parece.

La prueba suele enfrentarse a una oposición activa y su grado de dificultad viene determinado por el nivel de desafío del oponente. **En muchos casos pueden intervenir las Operaciones de Control en lugar de las de Potencia** para llevar a cabo o identificar una acción de este tipo pues los procesos de autocontrol y de percepción que son gestionados por esa operación son determinantes para tener éxito. Esta Rutina se puede considerar por lo tanto como perteneciente a ambas Operaciones y dependiendo de la situación se puede elegir la más apropiada.



Esta Rutina no puede usarse nunca sobre otros jugadores. Un engaño entre dos personajes jugadores se realiza siempre a nivel narrativo.

Ten en cuenta que para la IA de un programa el proceso de manipulación o engaño es algo tan complejo y sofisticado que **muchos programas jamás llegan a entenderlo y mucho menos a dominarlo**. Esto se aplica en ambos sentidos, tanto al hacer las acciones como al detectarlas. La mentira forma parte de la naturaleza humana por lo que se trata de algo muy característico de su especie.

Los personajes de naturaleza sintética deberían experimentar problemas a la hora de intentar realizar acciones relacionadas con el engaño, la mentira y la manipulación. Reflejarlo en el juego se puede hacer de muchos modos comenzando por supuesto con la posibilidad de contar con ventajas y desventajas. Pero más allá de recurrir a las consabidas mecánicas, interpretar a personajes incapaces de mentir, tan ingenuos que no sean capaces de entender lo que significa o con muchas dificultades para hacerlo puede generar situaciones muy interesantes y divertidas.

OFICIO

En muchos mundos virtuales y de juegos un personaje puede contar con un oficio que le permita realizar unas tareas específicas. Así es posible asumir oficios como herrero, carpintero, constructor de barcos, obrero, modelo, sacerdote, vendedor, tabernero, abogado, médico, conductor, policía, investigador privado... El único límite estará determinado por las características del escenario. Aquellas profesiones relacionadas con la expresión artística se resuelven con las Rutinas específicas de Artes que ya se han explicado y es posible aplicarlas del mismo modo que con los oficios.

El oficio permite al personaje desenvolverse en su entorno, obtener ingresos y ocupar un lugar dentro de la jerarquía social que se haya construido para la simulación. Por otra parte le facilita poder tener éxito cuando se marca un objetivo relacionado con su oficio.

PSICOLOGÍA HUMANA (u otra especie)

Aunque forma parte de las áreas del conocimiento un personaje puede estar especializado en reconocer los patrones de conducta de la Psicología humana

para poder atender mejor las necesidades de los usuarios. Ese es por ejemplo uno de mis cometidos. Aunque aquí se especifica "Humana" puede estar dedicada a cualquier otra especie.

Un personaje de naturaleza digital utiliza esta Rutina cuando quiere averiguar las intenciones o la razón de las acciones de un usuario. También puede usarse para intentar emular su conducta ya que para un programa es muy difícil hacerlo. Sus reacciones emocionales están muy lejos de pasar como naturales para cualquier usuario, que encontrará su conducta como mínimo anormal y estrafalaria.

También se puede utilizar si lo que se desea es manipularlo o engañarlo en lugar de recurrir a la Rutina "Manipulación y engaño" aunque en este caso el uso sólo se podría aplicar sobre la especie escogida. Pero como se ha explicado el engaño es algo muy sofisticado para la IA de un personaje digital por lo que puede experimentar muchas dificultades al querer realizar alguna acción relacionada con estas intenciones. No obstante una especialización en Psicología supone una gran ayuda para mejorar este tipo de "habilidades" y le llevará a estar lo más próximo a lo que puede aspirar una entidad digital para emular con eficacia el arte del engaño de los humanos.

En muchos casos pueden intervenir las Operaciones de Control en lugar de las de Potencia para detectar e identificar algunos aspectos de la Psicología pues los procesos de autocontrol y de percepción que son gestionados por esa operación son determinantes para tener éxito.

La dificultad depende de lo que se pretende conseguir y puede variar de un usuario a otro.

RECURSOS

El acceso a los recursos puede ser muy beneficioso para un personaje. Al fin y al cabo es una de las motivaciones principales que tienen los usuarios por lo que en los mundos virtuales puede suponer la diferencia entre el éxito o el fracaso. Utiliza esta Rutina cada vez que el personaje desee tener acceso a elementos difíciles de conseguir u obtener beneficios de muchas clases, como disponer de algunos privilegios o tener acceso a zonas restringidas.

Un personaje que disponga de recursos puede ocupar un lugar importante en la jerarquía del mundo virtual donde se esté ejecutando. La dificultad viene determinada por la facilidad que exista en ese sistema o mundo virtual de acceder a los recursos que se desean.

MANTENIMIENTO DE PROGRAMAS (sólo sobre programas)

Puedes usar esta Rutina para efectuar reparaciones de emergencia en el código de otro programa que se haya visto comprometido. Cuando se trata de reparar códigos dañados o alterados de otros programas el mantenimiento permite a su especialista poder manipular su código. Esta Rutina es muy amplia y puede permitir hacer muchas cosas, desde reparaciones menores a cambios completos en el programa y en su Avatar. Pero ten en cuenta que sólo se puede aplicar sobre un código que lleve la etiqueta y **se considere una criatura de naturaleza digital**, lo que en el juego suelo llamar **un programa**.

El nivel de dificultad está determinado por el nivel de daños. Normalmente **un punto por cada uno perdido en una operación o por cada casilla de integridad** que haya sido ocupada.

Esta Rutina no permite anular o restablecer un Protocolo de respuesta que se haya activado ni eliminar puntos permanentes sobre las reservas de CPU o de Memoria. Tampoco se pueden recuperar los puntos en Operaciones que se hayan perdido de forma permanente debido a los fallos de la Memoria.

La prueba con éxito durante una escena permite recuperar un dado en una operación que se haya perdido temporalmente debido a los daños recibidos o una de las casillas de Integridad como máximo. En el segundo caso el nivel de éxitos obtenidos en la prueba determina el valor de la casilla que es posible recuperar. Si se obtienen 3 éxitos se puede recuperar la casilla 3 por ejemplo. Una vez aplicada la Rutina de mantenimiento volver a intentarlo no tendrá efectos durante la escena en curso, aunque puede volver a usarse en la escena siguiente. Para poder usarla el programa debe estar libre de situaciones que requieran su atención y no puede realizar ninguna otra función mientras la ejecuta.



TÁCTICA Y ESTRATEGIA

Las Rutinas tácticas dedican sus funciones a encontrar las soluciones óptimas para un problema táctico o estratégico. Una prueba con éxito significa poder contar con alguna ventaja para acometer una acción o encontrar una solución para poder arreglar algún problema logístico. Una victoria siempre debería suponer algún beneficio para el personaje y su grupo.

USO DE ARMAS DE CONTACTO DIRECTO

El uso de armas de contacto directo significa que un programa debe entrar en contacto con su oponente entrando como mínimo en su primer área o área 1. En un mundo virtual o de juego supone un enfrentamiento cuerpo a cuerpo.

Puede tratarse de espadas, hachas, mazas, porras y cualquier otro objeto del escenario que pueda ser tomado por el personaje. Las armas de contacto directo siempre son **Elementos** aunque también pueden ser **Complementos**. Más adelante se explica en qué consiste cada término.

Su principal ventaja es que el programa puede beneficiarse de contar con el valor de su potencia en caso de que lo tengan. Todos los elementos que han sido diseñados con este fin disponen de alguno, que en términos de juego significa que otorgan una cantidad de éxitos extra que se añaden a la tirada y al daño total. Un objeto cotidiano normalmente no cuenta con valores de potencia lo suficientemente efectivos, pero si el Director lo cree justificado algunos por ser convenientes para la tarea o por cualquier otra razón podrían contar con al menos un valor de 1 punto. Por ejemplo, una barra de metal.

—**USAR DOS ARMAS:** En el caso de querer usar dos armas a la vez no es necesario especificar nada más. La Rutina otorga todas las posibilidades a su dueño. Si el Director lo considera apropiado el jugador puede obtener ventaja al usar dos armas.



RUTINAS DE CONTROL

ACROBACIAS

En un mundo virtual y en cualquier entorno de juego esta especialización convierte al personaje en un acróbata, un rasgo muy común de los personajes en los mundos diseñados para jugar. Estos serán capaces de hacer maniobras de muchas clases con gran facilidad como giros en el aire, salto entre barras, caer de pié, hacer equilibrios, dar volteretas y muchas otras acciones. El nivel de dificultad dependerá de la acción que se pretenda realizar y el número de éxitos conseguidos lo bien que se ha ejecutado la tarea.

ADQUISICIÓN DE DATOS AVANZADA

La adquisición avanzada especializa a un programa en los procesos de toma de datos de su entorno inmediato como hallar cualquier elemento capaz de ser encontrado como un trozo de código, un objeto, una amenaza, un fichero oculto, una llave escondida, un panel de control secreto, una trampa, etc. En cualquier mundo virtual y de juego se hace extensible al entorno que esté dentro del área de influencia del personaje o lo que es lo mismo, dentro de su área de percepción.

La dificultad de la prueba establece lo complejo que resulta encontrar lo que se está buscando.

ALERTA Y DETECCIÓN

Los personajes con esta Rutina son muy eficaces a la hora de percibir el entorno para detectar amenazas que se aproximen e identificar a otros programas que entren en su área de influencia.

El nivel de dificultad de la prueba depende de las condiciones del entorno y de las características de lo que se espera o debería identificar.

ARTES MARCIALES

Las artes marciales son una forma de lucha de contacto directo especializada en neutralizar amenazas específica de los mundos virtuales y de muchos entornos de juegos. Implica conocer una serie de técnicas de enfrentamiento que conceden un conjunto de beneficios al especialista.

En términos de juego cada arte marcial funciona como un arma, añadiendo un valor de potencia de entre 1 y 3 puntos. Si se desea es posible complicarlo más destinando estos puntos según si la maniobra que se desee realizar se trata de un ataque o de la defensa.

Las artes marciales se subdividen en subcategorías que se aprenden por separado como Kung-fu, Ninjutsu, Karate, Taekwondo... Existen muchos estilos de lucha.



ATLETISMO

La especialización convierte al programa en un atleta para un gran número de pruebas. Esta habilidad es esencial, al igual que Acrobacias, en un entorno virtual o en un mundo de juego, permitiendo en el segundo emular las acciones de sus personajes protagonistas. Cualquier acción que requiera saltar, correr o trepar con habilidad requiere pruebas de atletismo, pero a ser posible siempre que la acción suponga un riesgo para el personaje y un fracaso tenga consecuencias.

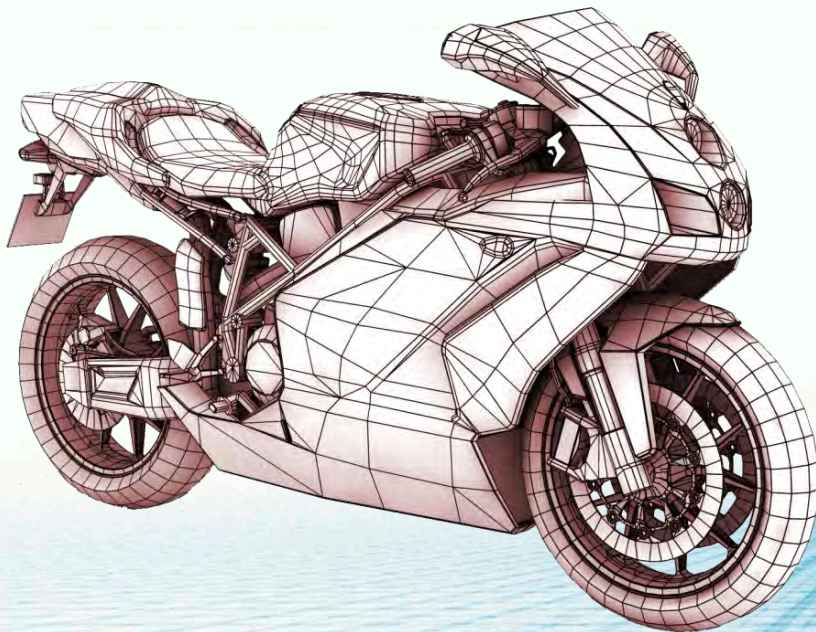
El valor de la operación de Control puede condicionar algunas acciones de Atletismo. Por ejemplo, un valor de Control de un punto significa que el personaje no puede intentar hacer algunas acciones más allá de una dificultad moderada, como hacer un salto muy largo por ejemplo. Un valor de dos o tres puntos elimina esta restricción.

CONducir VEHÍCULO

Esta Rutina especializa al personaje en la conducción de cualquier elemento que en el sistema pueda servir como vehículo. Desde motos, coches y tanques hasta aviones, helicópteros y aeronaves. Se subdivide en familias por tipo de

vehículo por lo que el personaje debe contar con la Rutina para cada uno; por ejemplo conducir motos, coches, pilotar helicópteros, etc.

La dificultad de las pruebas estará determinada por la maniobra que se vaya a realizar. Normalmente no es necesario realizar pruebas para las acciones rutinarias de conducción pero sí para aquellos casos en los que se presenten problemas o amenazas que si no se superan pueden tener consecuencias para el personaje.



CONTROL DE SISTEMA REMOTO

Mediante esta Rutina el programa tiene el control de dispositivos y periféricos que normalmente se encuentran en la Realidad Básica, aunque en algunos casos puede tratarse de elementos del Sistema. La especialidad le permite controlarlos y beneficiarse de sus ventajas.

La dificultad para controlarlos depende de los niveles de seguridad, de las características del hardware o de las del sistema que se pretenda mantener bajo control.

ESTRUCTURA DE SISTEMAS

Si el diseño permite planificar componentes la Estructura de Sistemas permite construirlas en el entorno. Mediante esta Rutina el personaje se convierte en un arquitecto del entorno virtual. Pero para poder construir algo es indispensable contar antes con un diseño o será una tarea muy difícil.

Con Estructura de Sistemas es posible crear programas que describen "algo", como vehículos, armas o dispositivos, e incluso alterar partes del escenario. Añadir una estructura importante, como un edificio, o hacer una modificación compleja en el escenario requiere contar con el permiso del Sistema. Esos permisos se pueden obtener mediante el comando /Attrib, aunque hacerlo significa cometer un acto ilegal e incurrir en una Anomalía. Para construir "Extras", como Elementos o Complementos que un programa puede usar directamente en cambio no es necesario contar con permisos especiales. Más adelante se explica lo que significan los Extras en el juego.

Estructura de Sistemas se divide en subcategorías, cada una especificando un área de especialización. Construcción de elementos de maquinaria o herramientas por ejemplo, de equipo, vehículos, armamento, modificación del paisaje, elementos arquitectónicos del entorno, modelado de Avatares, elaboración de Complementos, etc. La dificultad viene determinada por lo complicado que sea elaborar la estructura o el Extra, la facilidad de poder incluir los nuevos elementos en el entorno y sobre todo si se cuenta o no con **un diseño previo.**



SIGILO

Los mundos virtuales y los entornos de juegos emulan las acciones del mundo físico por lo que en muchos de ellos contar con una especialización para poder actuar con sigilo supone una ventaja para el personaje.

Las pruebas de sigilo a menudo se realizan contra una oposición activa, enfrentando la tirada del personaje contra la de un oponente que pueda descubrirlo. Lo normal es que entren en juego las Operaciones de Control para detectarlas. Si de lo que se trata es de ser sigiloso ante un conjunto de circunstancias, es posible establecer un nivel de dificultad para todas ellas y lanzar los dados, aunque lo conveniente en ese caso es enfrentar la tirada a una oposición pasiva en forma de una cantidad de éxitos fijos que hay que superar.

USAR ARMAS A DISTANCIA

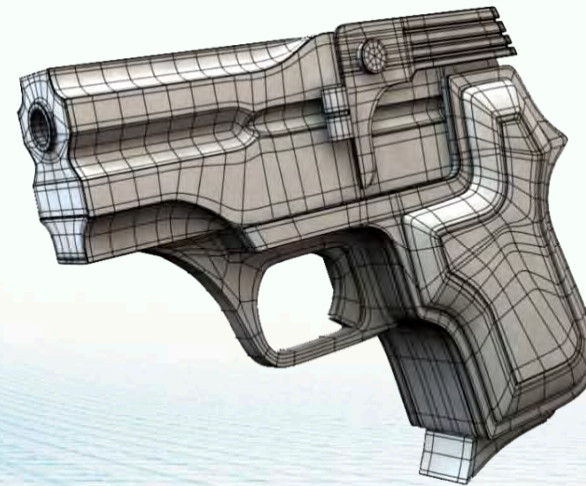
Las armas a distancia permiten a un personaje actuar sobre su amenaza sin necesidad de estar en contacto directo. Puede usarse tanto en la primera como en la segunda área pero debe encontrarse siempre dentro del área de influencia de su objetivo. Normalmente el arma con rango deja de ser efectiva más allá de esas zonas.

En un mundo virtual o en un entorno de juego supone un enfrentamiento con armas a distancia como arcos, pistolas, subfusiles, fusiles, armas láser, escopetas e incluso bombas y explosivos. También es posible tomar objetos del escenario y lanzarlos.

Las armas son Elementos y pueden ser Complementos. Todos ellos tienen un valor de potencia. En términos de juego significa que otorgan una cantidad de éxitos extra que se añaden a la tirada y al daño total.

Un objeto cotidiano tomado del escenario normalmente no cuenta con unos valores de potencia que puedan ser lo suficientemente efectivos para que se pueda aplicar un bonificador. Pero si son convenientes por algún motivo algunos pueden contar con al menos un valor de potencia de 1 punto.

—**USAR DOS ARMAS:** En el caso de querer usar dos armas a la vez no es necesario especificar nada más. La Rutina otorga todas las posibilidades a su dueño. Si el Director lo cree conveniente el jugador puede obtener ventaja al daño al usar dos armas contando un éxito extra al resultado de la tirada.



EJEMPLOS DE UTILIDADES

Las Utilidades son talentos únicos que posee un personaje. Sólo quien la posee puede beneficiarse de sus efectos. Su interpretación depende del tipo de personaje y del contexto en el que se encuentre. Una Utilidad puede consistir en la capacidad que posee un programa antivirus de aislar a las amenazas o devorarlas, en una de las armas secretas con las que cuenta un usuario que

“La situación sobre nuestra Tierra es paradójica. Las interdependencias se han multiplicado. La conciencia de ser solidarios con su vida y con su muerte liga desde ahora a los humanos. La comunicación triunfa; el planeta está atravesado por redes, faxes, teléfonos celulares, módems, Internet. Y sin embargo, la incompreensión sigue siendo general.”

Edgar Morin

pretenda hackear un sistema o bien en un sistema de misiles montado sobre el chasis de un robot de combate. La Utilidad se adapta a la función del personaje. Sus efectos no se pueden emular por medios normales aunque algunos Complementos pueden otorgar a un programa funciones que actúan como una Utilidad.

Para utilizar una Utilidad siempre hay que recurrir a la Memoria. La cantidad reservada determinará el grado y eficacia del efecto. El jugador puede dedicar los puntos que quiera para hacer la tirada, pero siempre que no sobrepase su límite máximo de memoria disponible. Recuerda que hacerlo siempre supone asumir un

riesgo. Cada punto como se ha visto equivale a un dado que se puede añadir a la reserva. Los dados no se conservan de una tirada a la siguiente como sucede con las Rutinas y la gestión del Tiempo de proceso.

Para poder usar una Utilidad el personaje debe destinar al menos 1 dado como mínimo a la reserva o no funcionará. En general destinar un dado permite beneficiarse de un efecto menor, el grado más simple de lo que es posible lograr al usar la Utilidad. Dedicar todos los puntos posibles sin sobrepasar al límite permite obtener el resultado más potente que es posible alcanzar. Normalmente el efecto más poderoso se obtiene al dedicar entre 6 y

8 puntos de Memoria, lo que permite añadir otros tantos dados a la reserva. Por último, existen algunas Utilidades más sencillas que sólo requieren un mínimo de puntos para poder usarse. En esos casos se especifica siempre la cantidad necesaria en su descripción.



Puedes usar las Utilidades que se describen a continuación como ejemplo para crear las que creas convenientes. Si quieres que tu programa sea capaz de hacer algo fuera de lo normal para realizar sus funciones principales inventa una Utilidad que le permita poder hacerlo. Con ellas dispones de total libertad para crear un personaje hecho a tu medida.

UTILIDADES DE POTENCIA

ÁREA DE CONTENCIÓN

Esta Utilidad permite al programa crear un área que retenga a una o a varias amenazas. El campo de contención las mantiene confinadas dentro de un espacio. Para que tenga efecto la tirada debe tener tantos éxitos o más que el nivel de la amenaza más poderosa o la de su líder. El efecto dura al menos hasta el final de la escena.

ARMA INTEGRADA

El personaje cuenta con una o varias armas incorporadas que puede usar siempre que quiera. Mientras no estén en uso es difícil para otros programas

poder detectarlas. La cantidad de dados destinados a la reserva de la Memoria determinarán la eficacia final del arma.

Los puntos destinados a la reserva pueden determinar también la potencia extra añadida que posee el arma incrementando su factor de 1 a 4. Por ejemplo, si el personaje destina dos puntos a la Memoria puede obtener un arma con un factor de potencia 1, si destina 4 una con un factor de potencia 2, etc.

En un mundo de juego suele tratarse de armas extra que el personaje lleva incorporados o de capacidades que se amoldan a su Avatar como extremidades extensibles o armas naturales. La mayoría de los tipos de ataques secundarios de los que disponen estos personajes se pueden catalogar como Utilidades que entran en esta categoría. Esta Utilidad también es aplicable a un hardware operando en la Realidad Básica reflejando el armamento que tiene disponible en su configuración.

ARMADURA DE DATOS

El personaje es capaz de rodearse con una armadura, invocar una o hacerla crecer a su alrededor. Permanecerá activa hasta el final de la escena. La forma y naturaleza de esta protección depende de cómo quiera describirla el jugador.

Su eficacia dependerá de los resultados pero los puntos destinados a la reserva pueden determinar también la potencia extra añadida que posea. Por ejemplo, si el personaje destina dos puntos a la Memoria puede obtener una armadura con un factor extra de protección de 1 punto que se añade al resultado; si destina 4 una con un factor extra de 2 puntos, etc.

En otros casos puede consistir en detener del todo un tipo de ataque concreto haciendo invulnerable al personaje, como al daño provocado por las balas o por el de las armas de contacto directo.



CABALLO DE TROYA

Esta utilidad permite introducir una pequeña sonda en el interior de un sistema de seguridad, un recinto fuertemente protegido o la guarida de un antagonista sin ser detectada. Una vez dentro ejecuta sus funciones para descubrir la forma de desactivar sus defensas y abrir un acceso que permita acceder al operador y a sus aliados sin ser detectados. El jugador puede decidir qué aspecto tiene y su forma de actuar.

Para poder ejecutar sus funciones sin ser detectada debe actuar con sigilo por lo que necesita algo de tiempo. El nivel de la sonda depende de los éxitos obtenidos en la tirada del jugador. El Director puede recurrir a los desafíos para resolver sus acciones o a cualquier otro mecanismo de las reglas que considere apropiado.

CUARENTENA

Esta utilidad se encuentra con frecuencia en los programas antivirus. El personaje es capaz de aislar a una amenaza poniéndola en cuarentena tanto tiempo como quiera, pero para poder hacerlo debe antes derrotarla. La amenaza quedará confinada dentro de un espacio reservado —o bolsillo dimensional—. Este espacio usa las mismas funciones que emplea el entorno para definir el espacio abstracto aunque quedando fuera de la influencia del sistema por lo que no puede actuar sobre él. A efectos prácticos el espacio de cuarentena es indetectable.

DEFORMAR LA MATRIZ

El personaje puede tratar de deformar el entorno próximo dentro de su área de influencia. Para hacerlo interviene sobre las funciones que describen su mundo, alterando el código y manipulándolo a su antojo. De esta forma el personaje es capaz de reescribir la realidad durante unos instantes.

Esta utilidad permite obtener una gran variedad de efectos, desde provocar pequeñas alteraciones en las áreas más próximas hasta modificar por completo un espacio importante. El efecto estará determinado por la cantidad de puntos destinados a la reserva de la memoria. Un dado permite hacer un efecto menor, como deformar un objeto pequeño, lanzarlo o encender un fuego en una simulación virtual; con seis dados puede conseguir

provocar un pequeño terremoto, deformar las paredes de una habitación o una escalera, alzar y proyectar muchos objetos sobre una víctima, invocar una ventisca o prender un muro de llamas.

DESDOBLAMIENTO

El personaje es capaz de crear copias exactas de su Avatar. El número de copias depende de los resultados de su tirada pero por regla general necesitará obtener al menos 2 éxitos en una tirada sin oposición o bien destinar 2 puntos a su reserva por cada copia que desee crear.

Cada copia actúa como desee el personaje, confundiendo a las posibles amenazas y a todos los que se encuentren dentro de sus áreas de influencia. Cuando es atacada y dañada se revela su verdadera identidad.

DEVORADOR DE CÓDIGO / GUSANO

Esta utilidad permite crear un gusano que va devorando el código lentamente y a lo largo del tiempo. Su principal ventaja es que es difícil de detectar. Su nivel de eficacia depende de los éxitos obtenidos al hacer la tirada o de los puntos destinados a la reserva, como se prefiera.

Una vez haya transcurrido el tiempo suficiente el gusano cumplirá su labor, minando la resistencia de una protección hasta eliminarla o abriendo una brecha que permita el acceso. Una vez lo haya hecho desaparece. Si lo desea el director puede recurrir a los desafíos para resolver sus acciones o a cualquier otro mecanismo de las reglas que considere apropiado.

HIELO NEGRO

Esta utilidad permite crear un muro sólido de datos que protejan al personaje y a sus aliados. El muro tiene también la particularidad de atrapar a todo aquel que lo toca. Puede adoptar la forma que desee su creador y adaptarse al entorno como él quiera. Su dureza y resistencia, así como la dificultad para liberarse, dependerán de la cantidad de éxitos obtenidos al hacer la tirada o a la cantidad de puntos destinados a la reserva de la Memoria, como se prefiera. Ese número determinará el nivel de la oposición pasiva que debe superar el oponente que desee traspasarlo o liberarse.

REINTEGRAR CÓDIGO

Esta Utilidad permite reintegrar el código dañado de un programa o incluso reparar secciones y elementos del mundo que hayan sufrido algún tipo de alteración o de corrupción en su código.

Por cada dos éxitos obtenidos en la tirada es posible recuperar un punto perdido debido al daño en Operaciones, pero no la pérdida de puntos por fallos graves que haya sufrido el programa. El resultado total en éxitos permite reintegrar el código absorbido por las casillas de estrés a razón de un punto de casilla por cada éxito obtenido. Las casillas que de este modo puedan ser recuperadas quedan libres para poder ser utilizadas de nuevo.

PERFORADORA

Esta Utilidad permite realizar la misma función que un gusano pero más rápido y probablemente de forma más ruidosa, por lo que es fácil de detectar. La perforadora es capaz de abrir un acceso en cualquier sistema de seguridad rompiendo el código mediante la fuerza bruta. Su aspecto lo decide el jugador. Puede parecer una fina línea de luz láser o un gigantesco taladro con un diseño de dibujos animados.

Siempre que el tiempo sea un factor importante, a nivel de mecánicas maniobrar con el taladro puede resolverse mediante un **reto por acumulación**, si no es así no es necesario recurrir a mecánicas. Los éxitos obtenidos deben superar un valor de “Dureza” expresado por una cantidad mínima de éxitos a conseguir en cada tirada. La diferencia de éxitos que sean favorables al jugador se pueden acumular. Una vez se obtienen los suficientes, una cantidad que decide el Director, éste puede dar la prueba como superada. Consulta las reglas sobre Dureza y Resistencia para más detalles. El tiempo necesario para hacerlo dependerá de la cantidad de éxitos que sea necesario acumular. Pero siempre existe la posibilidad de ser descubierto, algo que sucede al cabo de un tiempo o en cuanto las defensas tienen éxito en sus pruebas para detectarlo.

SABUESO INFERNAL

El personaje crea un programa de ayuda que combatirá por él hasta el final de la escena. Su aspecto lo decide el jugador pero casi siempre recuerda a un

perro demoníaco con un estilo que varía dependiendo de la ambientación. El nivel del sabueso será igual a los éxitos obtenidos en la tirada para invocarlo o bien a la cantidad de puntos destinados a la reserva de la Memoria, como se prefiera.

SUSPENSIÓN

Este tipo de Utilidades permiten romper las leyes que se han establecido en algunos Sistemas con el fin de emular la física de la Realidad Básica. Permite al programa caminar o flotar sin caer debido a las funciones que emulan a la gravedad. La dificultad de la tirada depende de la distancia y el tiempo que el personaje quiera permanecer en suspenso o levitando. Normalmente con dos dados es suficiente para evitar caer y sufrir sus efectos.

VISIÓN DEL CÓDIGO

El personaje es capaz de visualizar el código con el que está construido el entorno. Esta Utilidad le dota de una visión verdadera de la naturaleza de las cosas por lo que todo cuanto esté oculto o bajo otra apariencia le es revelada. La cantidad de dados destinados a la prueba determina la fuerza de esa visión y su capacidad para romper los sistemas de protección que pretendan ocultar elementos u otros programas en el código.



UTILIDADES DE CONTROL

CORRER POR LAS PAREDES

En un entorno virtual el personaje puede usar las paredes para correr al máximo de la velocidad de la que es capaz, lo que además de hacerle ganar una gran ventaja táctica puede permitirle escapar. La cantidad de puntos destinados a la memoria determina la distancia máxima que es posible recorrer.

DETENER LAS BALAS

El personaje puede detener todas las balas que se dirijan hacia él en un ataque. La cantidad de puntos destinados a la Memoria determinará la cantidad que es posible parar o la gravedad de los disparos sobre los que se puede mantener el control. Recuerda que intentar detener las balas siempre debería ser más difícil que tratar de esquivarlas.

El uso mínimo de esta Utilidad es de tres dados y permite detener unos pocos disparos mientras que seis o más detendrán una auténtica lluvia de plomo de varias armas automáticas disparando contra el personaje y/o sus aliados.

DESPLAZAMIENTO

El personaje es capaz de romper las leyes que emulan la física de un entorno virtual para poder cambiar su posición y reubicarse en otro punto del espacio abstracto. Esta Utilidad emula el concepto de teleportación.

Un uso mínimo de la Utilidad permite desplazarse una corta distancia dentro de las áreas de influencia de otros programas. Dos o tres dados le permiten salir de un área de influencia mientras que con seis o más el personaje y sus aliados pueden desaparecer y reaparecer en otras coordenadas del mundo sin que importe la distancia. Esta Utilidad sólo funciona dentro de un sistema en concreto, no es posible usarla para viajar entre sistemas.

EMPUJE

El personaje es capaz de manipular las leyes que emulan la física de un entorno virtual para conseguir efectos que le permiten controlar la posición de

los elementos del escenario. Puede alzarlos o controlarlos a voluntad. También es posible emular las fuerzas que actúan en la simulación provocando vectores de empuje que puedan derribar objetos o los Avatares de otros programas.

La cantidad de dados determina el efecto. Un solo dado permite controlar un pequeño objeto del tamaño de un puño mientras que seis u ocho permiten que el personaje levante un objeto de tamaño enorme, como un camión de dieciocho ruedas.

ESQUIVAR LAS BALAS

El personaje es capaz de esquivar las balas aunque no puede detenerlas. Esta Utilidad tiene la ventaja de que esquivar las balas siempre es más fácil que intentar detener su trayectoria.

En muchos sentidos funciona igual que "Detener las balas" pero el uso mínimo de la Utilidad es de un dado en este caso en lugar de los tres necesarios para detener un disparo. El uso mínimo permite esquivar unos pocos disparos. Más de seis dados permiten al personaje esquivar todas las balas en un tiroteo donde haya decenas de armas automáticas salpicando plomo en todas direcciones.

GANCHO

El personaje dispone de un gancho extensible que le permite acceder a distintas áreas del escenario con facilidad. Puede usarlo para cubrir amplias zonas y moverse con facilidad.

La dificultad de la acción que pretenda realizar y la distancia determinará la cantidad mínima de dados que es necesario utilizar. Un dado permite alzarse con el gancho agarrándolo a un elemento cercano del entorno. Más de cuatro o cinco dados permiten columpiarse entre los edificios de ambos extremos de una calle.

MUTACIÓN DE CÓDIGO

Esta utilidad permite a un personaje alterar el código para transformar durante un tiempo una de las Utilidades de un aliado en otra completamente distinta.

Normalmente se realiza sobre un miembro del grupo para adaptarlo a una situación. El efecto dura hasta el final de la escena. En la siguiente el personaje recupera su función original.

La nueva utilidad se ciñe a sus propias reglas por lo que hay que emplear sus mismas mecánicas. No es posible emplear esta Utilidad para alterar las funciones de las amenazas.

REPLICANTE

El personaje es capaz de imitar el Avatar de otro programa y alguna de sus Utilidades más importantes. La réplica es exacta por lo que es muy difícil detectar que se trata de una copia a no ser que se utilicen Utilidades que permitan examinar el código como "Visión del código". De igual modo, la Utilidad de la que se haya hecho una réplica funciona exactamente igual por lo que emplean sus mismas mecánicas.

Para realizar la réplica de las Utilidades el personaje debe superar una prueba contra el nivel del modelo. Para hacerlo debe entrar en contacto con éste o al menos estar en sus proximidades para poder evaluarlo correctamente. La cantidad de dados destinados a la tirada ajusta la calidad de la copia y la dificultad para ser detectada. El efecto dura hasta el final de la escena.

SALTO

El personaje es capaz de romper las leyes que emulan la física del entorno para realizar saltos que parecen imposibles. Con un uso mínimo puede saltar con facilidad los pocos metros que separan dos azoteas separadas por un callejón; con más de cinco dados puede hacer saltos tan amplios como los que separan los edificios opuestos de una gran avenida o entre dos manzanas.

SINCRONÍA

Al ejecutar esta Utilidad el personaje transfiere todos sus datos de Control a un aliado, que repetirá exactamente sus mismas acciones durante toda la

"¿Cual es la definición de Dios? Si te refieres al creador del mundo, no soy yo. Si te refieres al omnipotente del mundo, no soy yo. Si te refieres a la existencia en común con el mundo, si, podrías llamarme así."

Serial Experiments Lain

escena. Esto lo convierte en una "instancia" del donante pero sin que se trate de una réplica independiente. El proceso de sincronización no ocupa demasiado tiempo (un asalto) pero no es posible establecerla si existe algo que requiera la atención del personaje, como estando envuelto en un combate por ejemplo.

La instancia efectuará sus acciones tal y como **le dicte el donante, usando el resultado de las tiradas de éste al hacer las pruebas**. Pero las consecuencias de las acciones de la instancia recaen sobre ella obviamente. La Utilidad es muy efectiva cuando el donante quiere que otros aliados tengan sus mismas capacidades o para "estar ahí" cuando no es posible estar de otro modo. La instancia en cambio se beneficia al ser guiada de poseer capacidades que no tendría en circunstancias normales.

Esta Utilidad **no crea gemelos o duplica las capacidades de un personaje** para que pueda hacer dos veces una misma acción. O lo hace uno u otro, pero no ambos; uno controla y el otro responde. No obstante sí es posible que la sincronía pueda hacerse en ambos sentidos invirtiendo el proceso. De esta forma el donante adquiere las capacidades de un aliado y se convierte en una instancia suya.

Para establecer la sincronía basta con declararlo. Los personajes deben tener un breve instante para transferir su código estando obligados a permanecer próximos. Una vez hecha cada uno puede tomar su propio camino.

Durante el resto de la escena, cada vez que el donante quiera que su instancia actúe en sincronía debe añadir un mínimo de **dos puntos en la Memoria** cuando use sus Operaciones y sus Rutinas. Incrementando los bancos de memoria a **cuatro dados** el aliado podrá utilizar sus Utilidades. Si añade **dos dados más** la sincronía puede realizarse en ambos sentidos siendo posible intercambiar los papeles entre donante e instancia. Todos estos dados se acumulan a los que sean necesarios para usar otras capacidades.

Siempre hace las pruebas el donante asumiendo el gasto de Memoria mientras que la instancia será la que sufra las consecuencias cuando las cosas no salgan como se esperaba.

TREPAR

El personaje es capaz de trepar por las paredes y el techo de los edificios. Para hacerlo debe usar todas sus extremidades. Esta Utilidad no permite moverse tan rápido como "Correr por las paredes" pero tiene la ventaja de que la distancia que es posible cubrir es mucho mayor. El personaje no puede alcanzar la máxima velocidad de la que es capaz pero podrá moverse con facilidad todo el tiempo que quiera mientras dure la escena.

El uso mínimo de un dado permite caminar por las paredes y los techos de un local o de una vivienda mientras que cinco o seis harán posible desplazarse por las paredes de los edificios de una ciudad.

VELOCIDAD TERMINAL

El personaje es capaz de moverse tan rápido que al usarla a su máxima potencia no será más que un borrón imperceptible. Por cada dado empleado más allá del uso mínimo el jugador puede multiplicar su velocidad por ese valor. Por ejemplo, si un personaje destina cinco dados a la Utilidad puede multiplicar su velocidad por cinco. El efecto dura hasta el fin de la escena.

VOLAR

El personaje es capaz de romper todas las limitaciones de la simulación y desplazarse como desee. El efecto dura hasta el fin de la escena. Usar esta Utilidad es difícil. Para poder volar con libertad es necesario destinar como mínimo cuatro dados a la reserva de la Memoria. Maniobras más complejas y vuelos a gran distancia necesitan de más recursos, como mínimo entre seis y ocho dados.



EXTRAS

“Si tuvieras que arar un campo, ¿qué preferirías usar?, ¿dos fuertes bueyes o 1024 pollos?”

Seymour Cray

Todo código que sirva para describir una pequeña parte del mundo digital y **que pueda ser de utilidad para un personaje** se considera un **Extra**. Armas, vehículos, células de energía, botiquines, discos de datos, ropa, muebles, objetos del escenario, una escalera, un planeador, complementos de mejora, monedas, dispositivos que otorgan efectos temporales, herramientas, un teléfono, una lámpara, un tiesto, una rama, un bate de béisbol... todo puede ser un **Elemento**, un **Complemento** o un **Ítem**. Puede existir tanta variedad como permita la capacidad del Sistema global.

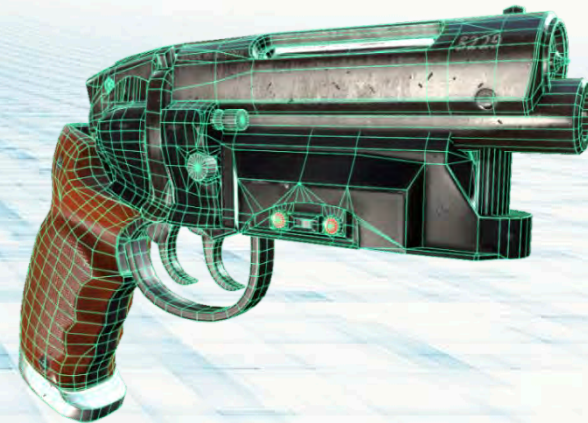
El mundo digital en Scroll se asume como un entorno compartido por millones de usuarios. En muchos casos éstos disponen de las herramientas, el permiso y la invitación a incorporar en un sistema sus propias creaciones, por lo que en la práctica la cantidad de extras disponibles es ilimitada. Dada la enorme diversidad de sistemas distintos que se pueden definir en Scroll no es difícil imaginar el vasto número que es posible crear para una ambientación.

Un mundo de juego puede tener miles de objetos descritos según sean sus características, diseño, función y particularidades. Pero por lo general si existen siempre es por alguna razón. En estos sistemas cada objeto significa esfuerzo y horas de trabajo por lo que suele haber una economía para su diseño. Todos deben de tener una función y servir para un propósito más allá

de tratarse de una simple descripción. No obstante si el sistema es compartido por una comunidad que tenga acceso y existe libertad de creación sí que es posible crear objetos que sólo sirvan como complemento y relleno del escenario. En esos casos basta con describir el objeto y poco más.

Para crear tus propios extras lo primero que debes hacer es tratar de identificar a qué grupo pertenecen. Para hacerlo debes contestar a las siguientes preguntas:

- ¿Qué es? ¿Cuál es su función? ¿Es un objeto o una mejora del personaje?
- Si son una mejora. ¿Son independientes o se combinan con su código?
- ¿Conceden bonificadores a quien los utiliza? Si lo tienen ¿cuál es su valor?
- Si se conectan al personaje ¿cuántos espacios necesita reservar en el Búfer?
- ¿Se pueden guardar? De ser así, ¿es posible agruparlos formando conjuntos para poder apilarlos?
- ¿Es posible utilizarlos varias veces? ¿Se agota su uso?
- ¿Es posible usarlos cuando se necesite o su efecto al acceder a ellos es inmediato?



Una vez tienes las respuestas es fácil identificar de qué tipo de extra se trata. Hacerlo permitirá definir sus especificaciones con más facilidad y usarlo con eficacia. A continuación dispones de una tabla que resume las características más importantes de cada grupo.

ELEMENTOS

- Especifican su **función** y su **potencia** aunque no es necesario tener un valor de potencia para ser considerados un Elemento. Sólo si tienen una aplicación directa para el personaje.
- No necesitan espacio en el Búfer.
- Ocupan espacio en el inventario (de haberlo). Por lo general no se pueden apilar.
- Se pueden usar cuando se necesitan. Su uso suele ser ilimitado.

COMPLEMENTOS O PLUGINS

- Especifican su **función**. En muchos casos imitan Utilidades.
- Necesitan espacio en el Búfer para funcionar.
- Estén o no en uso no necesitan guardarse en un inventario (de haberlo).
- Es posible reservarlos para cuando se necesitan. Son de uso ilimitado.
- Se conectan al código del programa, combinándose ambos.

ÍTEMS

- Sólo es necesario especificar su **función**.
- No necesitan espacio en el Búfer.
- Ocupan espacio en el inventario. Muchos se pueden apilar.
- Se pueden usar cuando se necesitan, pero su uso suele ser limitado.

Tokens

- Es un tipo de ítem. Sólo tienen una **función**.
- Su efecto es inmediato por lo que no se pueden reservar para usarlos cuando se quiera, y su uso es muy limitado.
- No necesitan espacio en el Búfer.
- No se guardan en el inventario y de hacerlo no ocupan espacio.

ELEMENTOS



Los Elementos se describen por la **función** para la que fueron concebidos y si conceden o no un bonificador, aunque no tienen que conceder bonos para ser considerados Elementos. En caso de tener alguno a ese bonificador se le denomina la **potencia** del Elemento. En la mayoría de los casos el bonificador se añade al número de éxitos totales de la tirada para determinar el resultado y calcular el daño si es que lo provocan. Hay casos en los que la potencia sólo se tiene en cuenta para calcular el daño total, no el grado de éxitos de la tirada. Se trata de algunas excepciones y siempre se especifica en su descripción. Uno que simule la explosión de un artefacto y que no esté siendo manipulado por ningún personaje es un ejemplo de este tipo de Elementos.

El valor del bono de potencia es una decisión que depende del tipo de Elemento, del escenario y del Director. No se añade con el objetivo de simular el comportamiento de algo sino para que **tenga un efecto en el juego**. En un

arma que pueda portar un personaje lo normal es que se trate de un valor de uno a tres, cuatro como máximo. Este bono describe la ventaja que supone emplear un arma respecto a no contar con ninguna. Si son armas más potentes pero que un personaje no puede transportar puede ser mayor.

Si se trata de un vehículo ese bonificador puede ser más alto, lo suficiente para reflejar su velocidad, la ventaja que supone para el personaje utilizarlo y también su peligrosidad en el caso de un accidente. A continuación se ofrecen algunos ejemplos.

ARMAS DE CONTACTO DIRECTO

Objeto contundente: bate, barra, tubo...

- **Función:** Objeto del escenario y arma de contacto directo.
- **Potencia:** 1
- **Espacio en el inventario:** 3 casillas.

Disco de datos

- **Función:** Sirve para almacenar información y como arma de contacto directo o con rango de efecto. Siempre regresa a su dueño.
- **Potencia:** 1
- **Espacio en el inventario:** Ninguna. Lo transporta el personaje llevándolo adosado a su Avatar.

Espada/hacha/maza

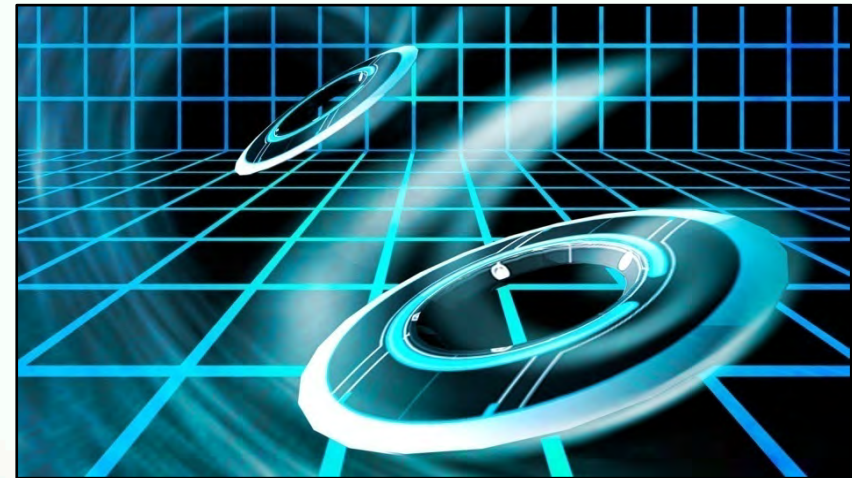
- **Función:** Armas cuerpo a cuerpo.
- **Potencia:** 1
- **Espacio en el inventario:** 3 casillas

Espada/hacha/maza a dos manos

- **Función:** Armas cuerpo a cuerpo.
- **Potencia:** 2
- **Espacio en el inventario:** 6 casillas

Arma de energía cuerpo a cuerpo

- **Función:** Arma de contacto directo de cualquier tipo basada en la energía.
- **Potencia:** 2 a 4.
- **Espacio en el inventario:** 3 casillas.



ARMAS CON RANGO DE EFECTO

Arco largo o compuesto

- **Función:** Arma con rango de efecto.
- **Potencia:** 1
- **Espacio en el inventario:** 3 casillas

Pistola automática

- **Función:** Arma con rango de efecto.
- **Potencia:** 2
- **Espacio en el inventario:** 1 casilla.

Subfusil/arma de energía de potencia media

- **Función:** Arma de rango de efecto de fuego rápido.
- **Potencia:** 3
- **Espacio en el inventario:** 2 casillas.

Escopeta recortada

- **Función:** Arma de rango de efecto corto pero que afecta a un área.
- **Potencia:** 3
- **Espacio en el inventario:** 3 casillas.



Fusil de asalto

- **Función:** Arma con gran rango de efecto de fuego rápido.
- **Potencia:** 4
- **Espacio en el inventario:** 6 casillas.

Arma de energía potente

- **Función:** Arma con rango de efecto de cualquier tipo basada en la energía.
- **Potencia:** 4.
- **Espacio en el inventario:** Varía.



Especial*

- **Función:** Cualquier Elemento imaginable que posea el personaje diseñado para o capaz de actuar como un arma. Este grupo incluye todas las posibilidades.
- **Potencia:** de 1 a 5.
- **Espacio en el inventario:** Varía.

Explosivos/granadas

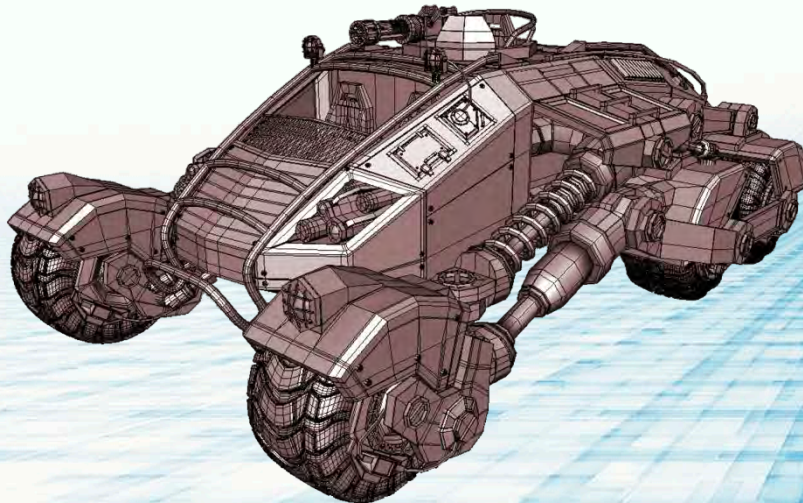
- **Función:** Arma de rango de efecto en un área. Sólo se aplica la potencia al daño resultante de la tirada, no a la cantidad de éxitos para averiguar si se supera la prueba.
- **Potencia:** 5
- **Espacio en el inventario:** 1 casilla.

VEHÍCULOS

Vehículo todo terreno

- **Función:** Al activarse el elemento se transforma en un vehículo abierto todo terreno que envuelve a su conductor y a otro pasajero. El vehículo puede desplazarse por terrenos que resultarían imposibles de atravesar para otros, como una moto. Cuando está desactivada tiene la forma de un pequeño mando en forma de barra curvada.
- **Potencia:** 8 (graduable). Incrementa la velocidad.
- **Espacio en el inventario:** 2 casillas cuando está desactivado.

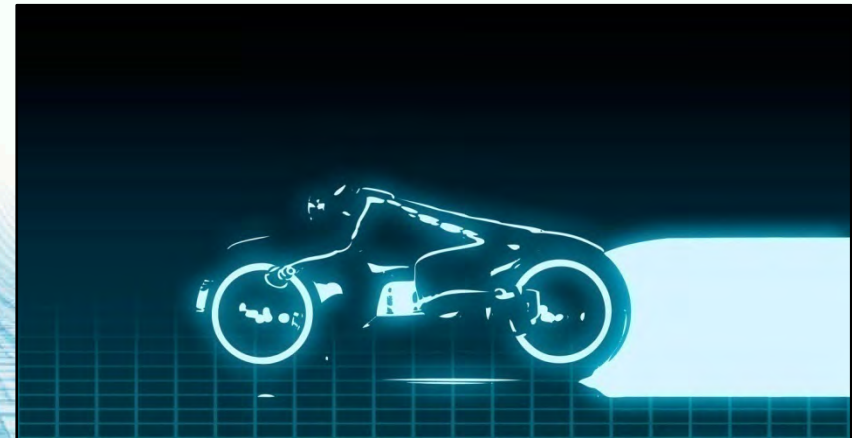
Nota: El conductor puede graduar su velocidad lo que le permite decidir el grado de potencia que añade a la tirada. **Cada punto incrementa el factor de velocidad del personaje por su valor.** En caso de accidente el grado elegido de velocidad se añade al total para calcular el daño.



Motocicleta de luz adaptable

- **Función:** Al activarse el elemento se transforma en una motocicleta que envuelve a su conductor. Cuando está desactivada tiene la forma de una pequeña barra de metal. Es posible activar un mecanismo que fuerza un muro de datos sólido que va dejando en su recorrido. El muro persiste mientras transcurre la escena (o un tiempo determinado por un número de asaltos) y se desvanece al final de la misma o si el vehículo es desactivado o destruido.
- **Potencia:** 6 (graduable). Incrementa la velocidad.
- **Espacio en el inventario:** 1 casilla cuando está desactivada.

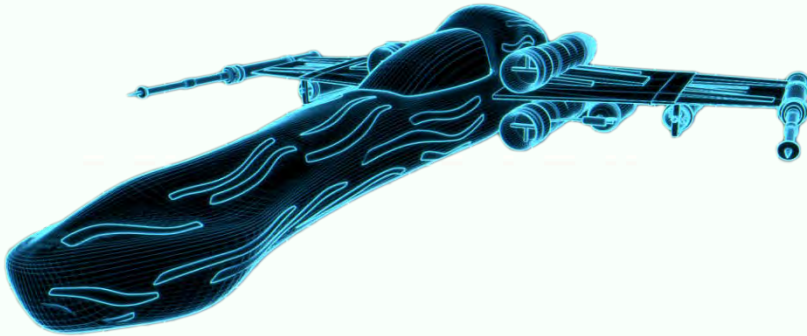
Nota: El piloto puede graduar la velocidad decidiendo el grado de potencia que añade a la tirada. En caso de accidente el grado elegido se añade al total para calcular el daño.



Aeronave

- **Función:** Aeronave para dos ocupantes que permite desplazarse por el espacio aéreo en distintos entornos simulados.
- **Potencia:** 12 (ajustable).
- **Espacio en el inventario:** no es posible almacenarlo.

Nota: El piloto puede graduar la velocidad decidiendo el grado de potencia que añade a la tirada. En caso de accidente el grado elegido se añade al total para calcular el daño.



DISPOSITIVOS

Restaurador de código

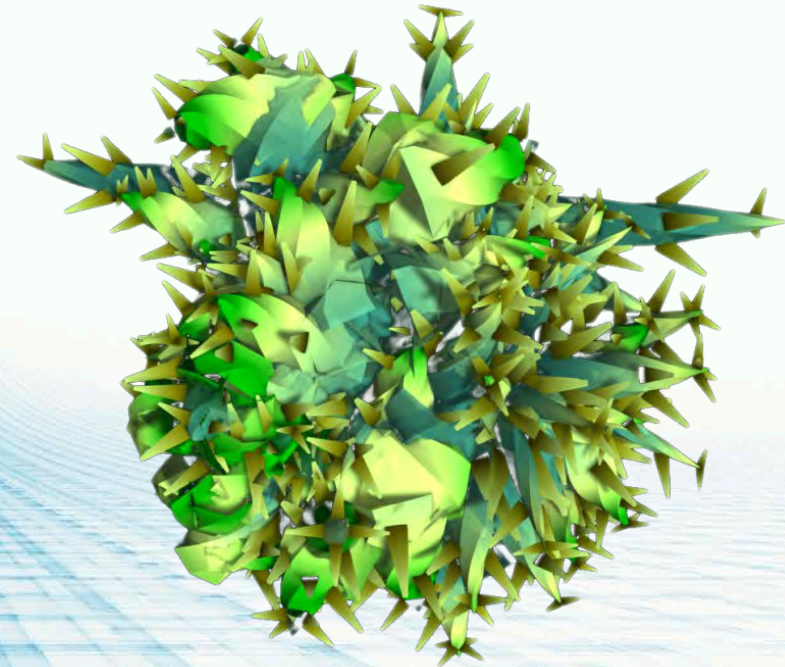
- **Función:** Sistema de reparación de código dañado. Permite reintegrar las funciones de un programa cuyo código se haya visto comprometido. Su potencia beneficia a una prueba para reintegrar el código de un programa.
- **Potencia:** 2 (graduable)
- **Espacio en el inventario:** 4 casillas

Paracaídas

- **Función:** Unas aletas en cruz se extienden para frenar una caída dentro de una simulación que disponga de un sistema de emulación de física. El usuario podrá planear hasta el aterrizaje. Es posible recoger de nuevo las aletas en el dispositivo para poder usarlo de nuevo.
- **Potencia:** no.
- **Espacio en el inventario:** 4 casillas.

Armadillo

- **Función:** Se trata de un sofisticado código de protección diseñado para proteger a un programa de las amenazas. Al activarlo una coraza similar a las escamas de un armadillo rodean el Avatar del programa. Mientras esté activo el valor de su potencia se añade a la cantidad de éxitos obtenidos para resolver cualquier ataque sufrido.
- **Potencia:** 2.
- **Espacio en el inventario:** 1 casilla.



COMPLEMENTOS

Para crear un Complemento piensa en un efecto que pueda dotar a un personaje de alguna habilidad especial o de una función extra. En muchos casos pueden emular Utilidades o reforzar Rutinas. También pueden cumplir las mismas funciones que muchos Elementos por lo que en este caso pueden tener un valor de potencia. De tener alguna incompatibilidad o para funcionar estar obligado a tener prioridad exclusiva sobre otros Complementos se debe especificar en el apartado de incompatibilidades. Al tener que combinarse con el código de su portador y ocupar espacio del Búfer un Complemento debe tener algún uso que valga la pena. De no ser así los Elementos pueden cubrir perfectamente las funciones de un Complemento.

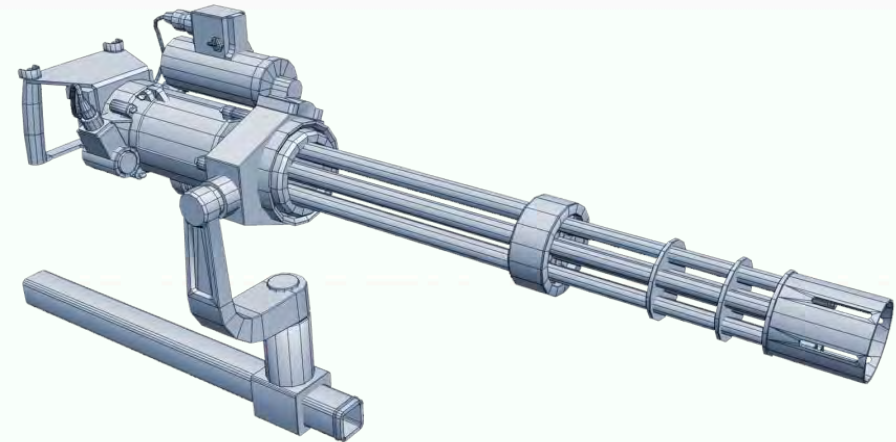
Los Complementos que se describen a continuación son ejemplos y se ofrecen como una sugerencia para que puedas crear los que desees.

Algoritmo de reparación

- **Función:** El Complemento repara automáticamente el daño recibido aunque siempre dentro de ciertas limitaciones. Cada vez que un programa recibe algún daño por primera vez se ignoran sus efectos. Todo el que venga a continuación se aplica normalmente.
- **Potencia:** no.
- **Espacio de Búfer:** 2
- **Incompatibilidades:** Ninguna.

Arma incorporada

- **Función:** Al estar activo su portador dispone de un arma oculta que puede hacer aparecer y desaparecer en cualquier momento. Puede tratarse de cualquier tipo de arma de contacto directo o con rango de efecto. El jugador decide cómo se integra y qué aspecto tiene.
- **Potencia:** Varía según arma.
- **Espacio de Búfer:** 1
- **Incompatibilidades:** Ninguna.



Control del Avatar

- **Función:** Permite al portador transformar a su Avatar rápidamente copiando a cualquier otro con el que haya tenido contacto o haya estado en sus proximidades. Se aplican las mismas reglas que la Utilidad "Réplica".
- **Potencia:** no
- **Espacio de Búfer:** 1
- **Incompatibilidades:** Puede dar problemas con Utilidades u otros Complementos que tengan algún efecto sobre la representación visual de la simulación.

Copia de emergencia

- **Función:** Permite a su portador poder realizar una copia de seguridad de sí mismo cuando haya perdido más de la mitad de la integridad de su código. La copia hereda el código comprometido. Al hacer cada copia el Complemento fuerza al Sistema cediendo parte de su propio código por lo que después de tres copias se desactiva y queda inutilizable. Este complemento es muy difícil de conseguir.
- **Potencia:** no.

- **Espacio de Búfer:** 3
- **Incompatibilidades:** Es necesario desconectar el resto de los Complementos activos para que funcione correctamente. Este Complemento es experimental y puede provocar algunos fallos en el rendimiento de las copias.

Sistema de vuelo

- **Función:** Permite a su portador poder desplazarse por el espacio aéreo en muchos entornos virtuales. Funciona como la Utilidad "Volar" y se deben aplicar sus mismas reglas.
- **Potencia:** no.
- **Espacio de Búfer:** 2
- **Incompatibilidades:** Ninguna.

ÍTEMS

Reserva el uso de los Ítems para todo aquello que un personaje pueda necesitar. Algo que como siempre dependerá del tipo de entorno. Puede tratarse de munición, comida o energía en algunos mundos virtuales o de juegos. Hay ambientaciones que no necesitan de ningún Ítem, como los escenarios esquemático realistas por ejemplo. Otros harán un uso intensivo de muchas clases distintas de ellos.

Los Ítems necesitan una descripción de su función y de su utilidad, si ocupan o no espacios en el inventario, si es posible apilarlos y en qué cantidad. La mayoría se pueden guardar para usarlos cuando se necesiten.

Comida

- **Función:** En algunos juegos permite recuperar parte de la integridad del código comprometido en el caso de utilizar las reglas de integridad. Normalmente entre 1 y 3 puntos de estrés.
- **Inventario:** 1 casilla. Apilable hasta 10 unidades.
- **Usos:** Un sólo uso. Se gasta al usarlo.

Munición

- **Función:** En algunas simulaciones y entornos de juegos permiten recargar el uso de un Elemento que sirva como arma.
- **Inventario:** 1 casilla. Apilable hasta 20 unidades.
- **Usos:** Un sólo uso. Se gasta al usarlo.

Recursos/dinero/oro

- **Función:** En algunos juegos y entornos virtuales permite disponer de una forma de cambio para poder adquirir bienes (que a su vez son Elementos, Complementos u otros Ítems). La forma de cambio depende de la simulación o del juego del que se trate. Puede tratarse de dinero en metálico, bonos, monedas de oro, créditos, gemas, objetos preciosos...,etc.
- **Inventario:** 1 casilla. Apilable de 50 a 100 unidades.
- **Usos:** Un sólo uso. Se gasta al usarlo.

Célula de energía/pack médico

- **Función:** En algunos entornos de juego y en los mundos virtuales permite recuperar parte de la integridad del código comprometido. Normalmente entre 1 y 6 puntos de estrés en caso de utilizar las reglas de integridad y hasta 1 punto de corrupción del código.
- **Inventario:** 1 casilla. Apilable de 5 a 10 unidades.
- **Usos:** Un sólo uso. Se gasta al usarlo.

TOKENS

Los Tokens son Ítems con unos usos muy determinados. Complementan las reglas de un entorno virtual y especialmente el de los mundos de juego ofreciendo recompensas a sus usuarios. Un programa podrá beneficiarse de ellos si consigue alcanzarlos. No se suelen poder almacenar y su uso está limitado al instante en el que se recogen, activándose su efecto automáticamente. Ya que no se puede reservar su uso para un momento de necesidad es necesario plantear una estrategia a la hora de alcanzarlos.

Algunos Tokens sólo están disponibles cuando los programas lo tienen dentro de su área de influencia, momento en el que se activan. En algunos casos para poder cogerlos el personaje dispone de un tiempo límite que se traduce en un límite de intentos para cogerlo. Si lo hace deberá hacer algún tipo de maniobra por lo que el jugador debe lanzar los dados y tratar de superar una prueba. Su dificultad será como mínimo moderada o desafiante. Si falla todas sus pruebas el Token desaparece.

Monedas/estrellas/puntos

- **Función:** Se trata de un tipo de Token destinado a otorgar una puntuación al personaje. Forman parte de las leyes del juego en los entornos de muchos de ellos.
- **Uso:** Sus efectos se producen al recogerlos. No se puede guardar aunque sus efectos son acumulables. Por ejemplo, almacenándose su puntuación en la memoria.
- **Tiempo límite para recogerlo:** Varía.

Comida/vida/energía

- **Función:** Su misión consiste en otorgarle energía al personaje. En términos de juego cada uno puede conceder puntos para recuperar la integridad de las casillas de estrés. A razón de un punto por Token o por una cantidad de ellos. En caso de no usar las reglas de Integridad este beneficio podría venir de otro modo aunque lo normal es que entonces no exista este tipo de Token.
- **Uso:** Sus efectos se producen al recogerlos. No se puede guardar aunque sus efectos pueden acumularse.
- **Tiempo límite para recogerlo:** Varía.

Tokens especiales que conceden efectos

- **Función:** Este Token produce un efecto sobre el personaje al recogerlo como otorgarle más potencia/fuerza o capacidades extra por un tiempo limitado. El tiempo suele consistir en un número de asaltos o durar hasta el final de la escena.

- **Uso:** Sus efectos se producen al recogerlos. No se puede guardar ni usar en otro momento que no sea al tomarlo.
A menudo obtener un nuevo efecto sustituye al anterior. Pueden existir excepciones que permitan acumular algunos efectos pero siempre con limitaciones por lo que esta debería ser la norma general. De ser así se debe especificar con claridad en su descripción.
- **Tiempo límite para recogerlo:** Varía.

COMANDOS MÁS UTILIZADOS

“El sistema operativo (por tanto) se ha convertido en una especie de instrumento para ahorrarse trabajo intelectual, que traduce las intenciones vagamente expresadas de los humanos a bits.”

Neal Stephenson

Los comandos están destinados a los usuarios, no obstante los programas pueden intentar forzar su ejecución, aunque no es ni debería ser nada fácil para ellos salvo unas pocas excepciones. Para ejecutar un comando es necesario hacer uso de la consola: una especie de pantalla con teclado flotante que posibilita su introducción en el entorno de un sistema. Un recurso muy útil cuando todo lo demás falla. En algunos escenarios tampoco resulta fácil invocarla por lo que puede ser necesario realizar pruebas que reflejan los esfuerzos del programa por hacerla aparecer o por emular sus funcionalidades.



Ten en cuenta que lo mismo da una consola y un comando que una llamada directa al sistema por cualquier otro medio que se te ocurra. El término

"consola" no es más que un recurso del juego y su objetivo es que resulte familiar para la mayoría de los jugadores.

Los comandos se agrupan en comandos de Potencia y de Control por lo que para ejecutarlos hay que recurrir a la operación que corresponda. El personaje que lo intente no puede recibir ayuda de sus aliados y el uso de cada comando cuenta como límite para el tiempo entre ejecuciones ya que se trata de llamadas directas al sistema y éstas siempre están restringidas. Por lo tanto, desde que uno de los miembros de un grupo trate de ejecutar un mismo comando su límite de tiempo y oportunidades contará para todos los demás. Estas restricciones por otra parte se refieren a cada comando, no a todos ellos. Tras la ejecución de un comando nada impide ejecutar cualquier otro.

A continuación de describen algunos de los comandos más comunes. Puedes usarlos como una guía para crear los que consideres apropiados. Para crear nuevos comandos considera que sus efectos siempre tienen algún efecto sobre el Sistema. La forma en que su ejecución beneficia a un personaje ya es otro asunto. Existen comandos que sólo se pueden aplicar sobre personajes con ficha o Personajes No Jugadores (PNJ) del Director. Si es el caso se especifica debidamente en su descripción. Cada comando se describe de la siguiente manera:

- **Nombre:** Nombre del comando.
- **Función:** Efectos del comando al ejecutarse.
- **Dificultad:** Expresada como una **oposición pasiva**. Un número mínimo de éxitos que hay que obtener **igualando o superando** en la tirada al valor expresado para ejecutar el comando con éxito.
- **Tiempo de procesador mínimo necesario.** Algunos comandos requieren un mínimo de CPU para ser efectivos. Si el programa no tiene acceso a ese tiempo de procesador el comando no funciona.
- **Memoria mínima necesaria:** Casi todos los comandos necesitan memoria para poder funcionar. Cada uno necesita de un mínimo o no podrá ejecutarse.
- **Tiempo entre ejecuciones.** Se especifica un tiempo mínimo en el que el comando no puede volver a ejecutarse. Normalmente consiste en una escena pero hay algunos que pueden volver a usarse en el

turno siguiente en una sucesión de asaltos. Los más inusuales son los que no permiten más usos mientras no se complete un capítulo o una fase de la aventura. El intento de cualquier miembro de un grupo cuenta para el límite de intentos de todos los demás miembros.

- **Tiempo de ejecución:** Tiempo que tarda un comando en producir su efecto una vez ha sido ejecutado. En algunos ese efecto es inmediato pero hay otros que requieren de un tiempo de espera. El tiempo necesario puede ser:

—**Inmediato:** El efecto se produce en el mismo turno en el que haya sido activado.

—**Un asalto:** El efecto se produce al finalizar el asalto y cuando todos hayan actuado. Si no hay un seguimiento de asaltos como en el enfoque absoluto la acción tiene lugar durante la escena en curso.

—**Una escena:** Algunos comandos requieren un tiempo para ejecutarse y éste se mide con la duración indeterminada de una escena. Aunque el efecto puede darse en cualquier momento lo normal es que se produzca al finalizar la escena, lo que en realidad señala un cambio en los acontecimientos. A veces es necesario resolver antes los conflictos que haya en una escena para poder ejecutar el comando si este lo requiere. Dar por terminada una escena puede ser tan sencillo como que el Director declare que el **tiempo avanza** hasta que se produce el efecto. Una vez se decide que la escena ha llegado a su fin tienen lugar los efectos del comando, que sirve de cierre y a su vez de inicio de una nueva. Es el caso del comando /Shutdown por ejemplo, que establece una cuenta atrás; al terminar y producirse el efecto se pasa a la escena siguiente.

Si se ejecuta en cualquier otro momento que no está claro o existen dudas el director describe una escena de corta duración en donde tenga lugar el uso del comando y sus efectos.

COMANDOS DE POTENCIA

/ATTRIB (ATRIBUTOS)

Función. Este comando permite alterar los privilegios de acceso sobre otro programa para poder manipularlo con otros comandos. Al cambiar sus atributos se habilita tener acceso de lectura y escritura por lo que es posible afectarle con ellos. El comando también permite obtener los permisos necesarios para crear estructuras en el sistema o hacer modificaciones en el entorno. La dificultad depende de lo importante que sean esos cambios. **No siempre es necesario modificar los atributos de un programa o del entorno**, sólo en el caso de que no se cuente con los privilegios necesarios.

Dificultad. Dos veces el nivel de programa o el de la amenaza. Los permisos para poder hacer modificaciones en el entorno se obtienen asignando una dificultad que oscila entre 4 y 12. Su valor depende de la importancia y el efecto de los cambios que se vayan a realizar.

Tiempo de procesador mínimo necesario. 4

Memoria mínima necesaria. 4

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Un asalto.

/DEBUG (DEPURAR)

Función. Este comando permite averiguar la información más relevante de un programa como su Designación, Funciones, Vulnerabilidades, Clase, Nivel, Rutinas y Utilidades. Para que sea efectivo antes es necesario cambiar sus atributos mediante el comando /Attrib para obtener privilegios de lectura sobre él.

Dificultad. Nivel del programa o de la amenaza.

Tiempo de procesador mínimo necesario. 2

Memoria mínima necesaria. 2

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Inmediato.

/DELETE PROGRAM (BORRAR PROGRAMA)

Función. Este comando borra la copia activa u original de un programa. Sus copias, en caso de tener alguna, no resultan afectadas. Quien lo ejecuta debe hacer una tirada e igualar o superar en éxitos el **doblo** del nivel de su víctima para conseguirlo. La víctima debe estar dentro de su área de influencia.

Dificultad. Obtener una cantidad de éxitos que iguale o supere el DOBLE del nivel del programa (o su nivel de desafío) que se desea borrar.

Tiempo de procesador mínimo necesario. 3

Memoria mínima necesaria. 3

Tiempo entre ejecuciones. Una vez por escena.

Tiempo de ejecución. Un asalto. El borrado se produce al finalizar.

/CLS (BORRADO DEL ÁREA INMEDIATA)

Función. Desactiva y borra el rastro de los Avatares de todos los **programas aliados** que estén dentro del área de influencia del programa que lo ejecute. A efectos de juego los personajes se vuelven invisibles hasta el final de la escena.

Dificultad. 8

Tiempo de procesador mínimo necesario. 2

Memoria mínima necesaria. 4

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Inmediato.

/ERASE PROGRAM (SUPRIMIR PROGRAMA)

Función. Este comando es una versión mucho más efectiva del comando /Delete, pues permite erradicar por completo a un programa. Si se ejecuta con éxito borra completamente la copia original activa de un programa y todas las copias de respaldo en el caso de tener alguna. El programa se desvanecerá con un grito escalofriante. Quien lo ejecuta debe hacer una tirada e igualar o superar en éxitos el **triple** del nivel de su víctima para conseguirlo. La víctima debe estar dentro de su área de influencia.

Dificultad. Obtener una cantidad de éxitos que iguale o supere el TRIPLE del nivel del programa (o su nivel de desafío) que se quiere erradicar.

Tiempo de procesador mínimo necesario. 6

Memoria mínima necesaria. 6

Tiempo entre ejecuciones. Una vez por escena.

Tiempo de ejecución. Un asalto. El borrado se produce al finalizar.

/HCF "HALT AND CATCH FIRE" (DETENER Y AUTODESTRUIR)

Función: Pone al sistema en una condición de carrera forzando a todos los programas a competir por su primacía al mismo tiempo. En consecuencia, todos comienzan a ejecutar sus procesos al máximo de su rendimiento. Cuando esto sucede el sistema pierde el control y ya no es posible recuperarlo por lo que al final de la escena éste cae y se reinicia. En algunos casos pueden producirse fallos de hardware debido al calentamiento de sus componentes físicos. El Director puede decidir si esto sucede o bien lanzar un dado para resolverlo.

Todos los programas están obligados a realizar sus tiradas con todos los dados que podrían usar si todos sus bancos disponibles (y sólo los disponibles) estuviesen completos. Las consecuencias siguen el procedimiento normal que se describe en las reglas.

Ejecutar este comando toma algo de tiempo, aunque nunca se conoce el momento exacto del reinicio. Si los programas no consiguen escapar del sistema a tiempo caerán con éste, aunque no serán borrados.

Dificultad. 20 éxitos. El comando no está disponible en todos los sistemas.

Tiempo de procesador mínimo necesario. Todos los que haya disponibles.

Memoria mínima necesaria. Todos los que haya disponibles.

Tiempo entre ejecuciones. Una vez por fase o capítulo.

Tiempo de ejecución. Una escena. El reinicio se produce al finalizar.

/HELP (AGENTE DE AYUDA)

Función: Invoca un agente de ayuda del sistema en el caso de haber alguno operativo. Se le pueden hacer preguntas que contestará dependiendo de la información que tenga disponible y de los permisos que posea el personaje para poder acceder a esa información. En caso de no tener los permisos necesarios la aplicación se negará a contestar. Al tercer intento se cerrará y no podrá volver a ser invocada mientras se esté en el mismo sistema o hasta el fin de la fase o capítulo. Si es atacada de cualquier modo se cerrará igualmente de inmediato.

Dificultad. 6 éxitos

Tiempo de procesador mínimo necesario. 1

Memoria mínima necesaria. 1

Tiempo entre ejecuciones. Una vez por escena.

Tiempo de ejecución. Inmediato.

/HOME (VUELTA A CASA)

Función. El programa es capaz de cambiar su ubicación y la de todos los aliados que se encuentren dentro de su área de influencia, reapareciendo en el área de reinicio que cada uno haya definido. Sólo es posible hacerlo si se

encuentra en una simulación que posea un sistema de coordenadas que defina el espacio virtual. Lo que incluye a muchos entornos de juegos y a casi todos los mundos virtuales.

Ejecutar este comando requiere tiempo por lo que sólo se hace efectivo al finalizar una escena. Quien lo ejecuta no puede estar inmerso en un conflicto que ponga en riesgo su integridad o se arriesga a que la ejecución se interrumpa.

Dificultad. Depende de la distancia que haya hasta el área de reinicio. Mínimo 6. Máximo 10.

Tiempo de procesador mínimo necesario. 4

Memoria mínima necesaria. 4

Tiempo entre ejecuciones. Una vez por escena.

Tiempo de ejecución. Una escena. El salto se produce al finalizar.

/MOVE (MOVER)

Función. Permite saltar a unas coordenadas determinadas. A efectos de juego el personaje puede teleportarse a un punto en el entorno si sabe dónde está.

Dificultad. Depende de la distancia. Mínimo 4, máximo 10

Tiempo de procesador mínimo necesario. 2

Memoria mínima necesaria. 4

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Un asalto. El efecto se produce al finalizar.

/PEEK (RECUPERAR CÓDIGO)

Función. Este comando restablece el valor en la memoria que haya alterado el comando /POKE (descrito a continuación), por lo que anula las ventajas obtenidas con ese comando.

Dificultad. Resultado de la tirada obtenida al ejecutar el comando /POKE.

Tiempo de procesador mínimo necesario. 1

Memoria mínima necesaria. 1

Tiempo entre ejecuciones. Un asalto.

Tiempo de ejecución. Un asalto. El efecto se produce al finalizar.

/POKE (MANIPULAR CÓDIGO)

Función. Este comando se usa con frecuencia para hacer trampas, por lo que es una herramienta muy útil si se desea alterar la función principal de un programa activo, de un sistema, un subsistema o las características de cualquier "Extra" que esté presente en el mundo digital. Permite cambiar un valor en la memoria de forma que altere alguna característica menor de cualquiera de ellos. El efecto dura hasta que finalice la escena. El jugador decide el resultado que desea pero debe tratarse de algún efecto menor como alterar su aspecto, las características de su función principal o conseguir algún efecto que le brinde a quien lo ejecuta una ventaja indirecta como munición infinita, una copia de seguridad temporal o recursos extra por ejemplo.

La única regla a recordar es que este comando nunca debería conceder la victoria directamente sobre un conflicto. Lo que provoca es algún efecto que suponga una ventaja de cualquier tipo. Su grado estará determinado por el resultado de la tirada. Al hacerla es conveniente anotar el resultado obtenido para usarlo como dificultad en el comando /PEEK

Dificultad. Nivel del programa (o su nivel de desafío) objetivo o según el resultado que se desea obtener.

Tiempo de procesador mínimo necesario. 2

Memoria mínima necesaria. 6

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Un asalto. El efecto se produce al finalizar.

/RENAME (RENOMBRAR)

Función. Permite renombrar la Designación, Nivel y Clase de un programa para enmascararlos o falsear esa información. Si se obtienen privilegios sobre otro programa —si es que no se tienen— también es posible afectarle con este comando. Para obtener privilegios de escritura consulta el comando /Attrib.

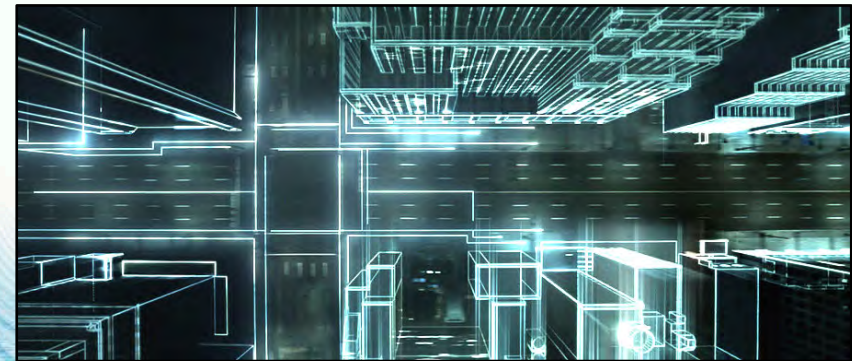
Dificultad. Nivel del programa (o su nivel de desafío).

Tiempo de procesador mínimo necesario. 1

Memoria mínima necesaria. 2

Tiempo entre ejecuciones. Un asalto.

Tiempo de ejecución. Un asalto. El efecto se produce al finalizar.



/RESET (REINICIAR)

Función. Fuerza el reinicio de otro programa. El programa afectado guarda sus datos y se cierra. Cuando se reinicializa aparece en el área de reinicio que haya designado. Si no se poseen privilegios sobre él es necesario recurrir al comando /Attrib para conseguirlos y así poder forzar el reinicio.

Dificultad. Dos veces el nivel del programa o de la amenaza (o su nivel de desafío).

Tiempo de procesador mínimo necesario. 4

Memoria mínima necesaria. 6

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Un asalto. El efecto se produce al finalizar.

/RUN (EJECUTAR)

Función. Permite cargar otra copia del programa en la Memoria de forma que puedan existir varias **instancias** de un mismo programa ejecutándose al mismo tiempo. El número de copias que posea determinará el máximo número de ejecuciones que pueden realizarse. Un programa siempre cuenta al menos con su copia original por lo que puede ejecutar como mínimo una instancia de sí mismo. No se puede usar este comando para hacer duplicados. Cuando todas las copias disponibles han sido cargadas ya no es posible cargar más.

Cada instancia carga con toda su integridad intacta pero los efectos que hayan sufrido las copias debido a algún error durante la ejecución afectan también a la instancia. Los nuevos datos forman otro personaje, una copia exacta del original pero capaz de emprender sus propias acciones. A efectos de juego el jugador dispone de un clon y puede hacer con él lo que quiera.

Por cada instancia que se desee cargar hay que añadir 2 niveles de dificultad más a la dificultad base y dos espacios de Memoria y de Tiempo de proceso adicionales para realizar la tirada.

Dificultad. 12 más 2 éxitos por cada instancia que se desee crear.

Tiempo de procesador mínimo necesario. 2 más 2 por instancia.

Memoria mínima necesaria. 2 más 2 por instancia.

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Un asalto. El efecto se produce al finalizar.

Nota: El límite máximo de instancias es igual al número de copias disponibles. Un programa siempre tiene una por defecto.

/SCR "SHOOT & CHICKEN RUN" (DISPARAR Y PIRARSE)

Función. Este extraño comando, creado por un hacker que terminó sus días flotando en las aguas de algún puerto asiático, permite forzar un reinicio de toda la sección del entorno que se encuentre dentro del área de influencia de un programa. El reinicio se localiza en esa zona exclusivamente, no afectando al resto. Esto fuerza al sistema a tener que cargar de nuevo todos los datos del área afectada del escenario. Todos los programas que se encuentren en ella, menos quien haya ejecutado el comando claro, se ven también forzados a reiniciar lo que les obliga a cargar en sus respectivas áreas de reinicio. Pero aún hay más..., a continuación el programa que lo haya ejecutado puede evadirse de la zona a toda velocidad exactamente de la misma manera que con el comando /CR "Chicken run" (ver más adelante). La distancia exacta es algo subjetivo. Será la suficiente para evadir cualquier encuentro y ponerse a salvo.

Dificultad. 14

Tiempo de procesador mínimo necesario. 6

Memoria mínima necesaria. 6

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Un asalto. El efecto se produce al finalizar.

/SHUTDOWN (REINICIAR EL SISTEMA)

Función. Este comando fuerza un reinicio completo del Sistema, que inicia una cuenta atrás avisando del reinicio inminente. Al cerrarse todos los programas son descargados de la memoria. Tras la reinicialización vuelven a cargar apareciendo en sus respectivas áreas de reinicio. El uso de este

comando se considera una falta grave que en muchos sistemas es penada con la descompilación lenta y “dolorosa” del programa, bit a bit...

Dificultad. 16

Tiempo de procesador mínimo necesario. 8

Memoria mínima necesaria. 8

Tiempo entre ejecuciones. Una fase o capítulo.

Tiempo de ejecución. Una escena. El reinicio se produce al finalizar.

/STOP (DETENER)

Función. Detiene la ejecución de un programa durante un tiempo. Para conseguirlo se realiza una tirada enfrentada al nivel de la víctima (o su nivel de desafío). Cada éxito obtenido en la tirada que lo sobrepase significa un asalto que el programa queda detenido por lo que no puede actuar durante sus turnos correspondientes. En el enfoque absoluto se detienen un tiempo las acciones de la víctima durante esa escena lo que permite a sus oponentes hacer otras acciones.

Dificultad. Nivel del oponente (o nivel de desafío).

Tiempo de procesador mínimo necesario. 2

Memoria mínima necesaria. 3

Tiempo entre ejecuciones. Una escena

Tiempo de ejecución. Un asalto. El efecto se produce al finalizar.

/VER (VERSIÓN)

Función. Permite averiguar la Designación, Funciones, Nivel y Clase de un programa. Para ello es necesario contar con privilegios de lectura sobre él —si es que no se tienen— con el comando /Attrib. Si en la tirada la diferencia de éxitos contra el objetivo es alta superándolo por 3 o más también es posible

averiguar sus Vulnerabilidades y otra información importante emulando al comando /Debug.

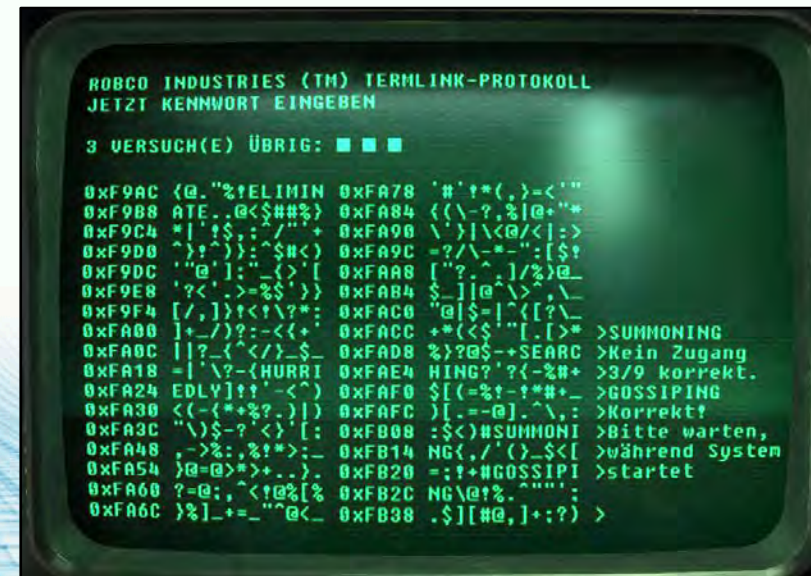
Dificultad. Número de éxitos igual o mayor que la mitad del nivel del programa o amenaza (redondeando arriba).

Tiempo de procesador mínimo necesario. 1

Memoria mínima necesaria. 3

Tiempo entre ejecuciones. Una escena

Tiempo de ejecución. Inmediato.



/WTF! (PERO QUÉ COÑO...)

Función. El término “WTF!” (What the fuck!) es una expresión que el usuario escribe con frecuencia en la consola. Hace ya algún tiempo una IA se tomó la molestia de interpretarlo en vista de que dado su uso reiterado sobre

todos los sistemas debía de tratarse de algo de importancia crucial. Eso dio lugar al nacimiento de uno de los primeros comandos creados por los seres digitales, siempre en el empeño de hacer la vida del usuario más fácil.

Cuando hay problemas en un sistema y el usuario no sabe qué es lo que sucede suele recurrir a este comando. Al hacerlo envía una solicitud al sistema para que le devuelva un informe exhaustivo sobre cualquier anomalía que se haya producido, esté o no resuelta. El informe se envía por cualquier dispositivo de salida disponible. Lo normal es que se trate de las anomalías que provocan algunos programas que no ejecutan su programación como es debido. Cuando un sistema emprende acciones para corregir alguna sigue un procedimiento, como enviar a sus agentes por ejemplo.

Si un programa consigue ejecutar el comando puede obtener un informe completo de sus planes y hacerse una idea de lo que sabe, lo que puede serle de gran utilidad para planear su próximo paso. Un trquito de espionaje haciendo una pequeña trampa. Si un sistema no ha conseguido solucionar las anomalías detectadas puede reflejar en el informe sus rasgos de personalidad, por ejemplo poniendo mil excusas o reiterando con pomposidad ante el usuario su férrea determinación a poner fin a los problemas.

El Director tiene derecho a reservarse cualquier información que considere vital para mantener el secreto de la trama.

Dificultad. Depende del sistema. Lo normal es entre 6 y 12.

Tiempo de procesador mínimo necesario. 4

Memoria mínima necesaria. 4

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Inmediato.



COMANDOS DE CONTROL

/BIND (FIJAR POSICIÓN)

Función. Fija la posición del área de reinicio en las coordenadas de un mundo virtual o de un juego. El entorno debe contar con un sistema de coordenadas que defina el espacio abstracto. En el caso de no existir unas definidas un programa siempre reinicia en las coordenadas 0, 0, 0; que casi siempre se corresponde con el centro exacto del mundo virtual o con el límite de una de sus esquinas.

Dificultad. 1

Tiempo de procesador mínimo necesario. 0

Memoria mínima necesaria. 0

Tiempo entre ejecuciones. Un asalto.

Tiempo de ejecución. Inmediato.

/CR "CHICKEN RUN" (PIRARSE)

Función. Este comando permite salir de un área lo más deprisa posible escapando del área de influencia de los programas con los que se tuviese contacto. El programa no desaparece y reaparece sino que escapa a toda velocidad, por lo que una amenaza podría intentar detenerle. Para conseguirlo debe igualar o superar con una tirada de Control la cantidad de éxitos que se hayan obtenido al ejecutar el comando.

La distancia exacta recorrida no es importante; basta con saber que es la suficiente para escapar. A efectos de juego el comando permite **evitar un encuentro** y evadirse, logrando ponerse a salvo fuera del área de influencia de las amenazas. Eso sí, nada les impide iniciar la persecución, aunque para averiguar hacia dónde ha huido el programa deberá utilizar otras habilidades como tratar de rastrearlo o localizar su nueva posición por cualquier otro medio.

Dificultad. 12

Tiempo de procesador mínimo necesario. 4

Memoria mínima necesaria. 4

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Inmediato.

/CHAT (USAR CANAL DE CHARLA)

Función. Este comando permite utilizar las áreas de charla o “chats” que utilizan los usuarios para sus comunicaciones. El programa puede hablar como si fuese un usuario más, y a no ser que se inicie alguna acción para averiguarlo nadie será capaz de conocer su naturaleza. No obstante la forma de comunicarse de un ser digital puede despertar las sospechas de un usuario del mundo físico. ...Aunque la mayoría ni se dará cuenta ya que muchos se expresan muchísimo peor de lo que puede hacerlo una IA de potencia media (NdE: experimento reproducible).

Dificultad. 2 o varía según el canal.

Tiempo de procesador mínimo necesario. 0

Memoria mínima necesaria. 0

Tiempo entre ejecuciones. Un asalto.

Tiempo de ejecución. Inmediato.

/COPY (COPIAR)

Función. Permite hacer una copia de un programa. La copia se almacena en el lugar que se haya reservado para ello en el sistema o bien donde decida el personaje siempre que cuente con esa posibilidad.

Algunos sistemas no permiten la realización de una copia sin su autorización o sin el consentimiento de un usuario; aunque esto es algo que depende de las leyes de cada sistema. De producirse el sistema lo interpretará

como una anomalía, normalmente porque deduce que se trata de un virus, por lo que actuará en consecuencia.

Dificultad. 12

Tiempo de procesador mínimo necesario. 6

Memoria mínima necesaria. 6

Tiempo entre ejecuciones. Una fase o capítulo. Una ejecución por copia.

Tiempo de ejecución. Una escena. La copia estará lista al final de la escena en curso.



/ESCAPE (FUNCIÓN ESC)

Función. Este comando fuerza la ejecución de la **Función Escape**. Resulta útil cuando un sistema ha sido capaz de anular la función normal (ESC) de la que disponen todos los programas y no es posible invocarla. Casi siempre para impedirles escapar.

Dificultad. Igualar o superar el nivel del programa que ejecuta el comando (o su nivel de desafío).

Tiempo de procesador mínimo necesario. 2

Memoria mínima necesaria. 2

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Inmediato.

/EXECUTE OPERATOR (EJECUTAR AL USUARIO)

Función. Este comando, que según la tradición no es más que un mito, está disponible, oculto y en secreto, en muchos sistemas. Cuando se ejecuta sobre un usuario ¡éste jamás lo permita! provoca una sobreestimulación de sus sentidos o una sobrecarga neuronal de sus funciones cerebrales. Depende del sistema que esté usando para operar en el sistema.

En el caso de que esté accediendo por medios convencionales, como pantallas o visores de datos, el comando comienza a generar patrones audiovisuales de cierta frecuencia que provocan el colapso. En el caso de que esté conectado mediante una interface neuronal o un escáner de ondas cerebrales se induce un sobrevoltaje que literalmente fríe sus tejidos. El efecto tiene 1/3 de probabilidades de matarlo (1-2 en un dado D6), o lo que el Director considere apropiado.

La ejecución de este comando **se considera la falta más grave que puede cometer un programa en toda su existencia.** De ser detectado, el programa será perseguido incluso más allá de los límites del sistema donde se produjese el crimen, pues se extenderá una orden de búsqueda por toda la Red Global. ¡Yo misma no dudaría en buscar al que fuera capaz de cometer un acto tan atroz!

Dificultad. 18

Tiempo de procesador mínimo necesario. 8

Memoria mínima necesaria. 8

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Un asalto. El efecto se produce al finalizar.

¿/Execute operator?

Si tienes conocimientos de informática te darás cuenta enseguida de que muchos de los comandos descritos han sido tomados de diferentes sistemas operativos, lenguajes de programación, videojuegos e incluso de algunos mitos que circulan desde los primeros tiempos de los ordenadores. Esto **es intencionado** y se hace con la intención de hacer algunos guiños al mundo de la informática.

Recuerda que Scroll pretende simular algunos procesos reales de la computación pero nunca emularlos con exactitud. Por eso te invito a que inventes los que te parezcan más apropiados o te resulten divertidos. La diversión debe primar sobre todo lo demás.

/FORCE INTERRUPTION (FORZAR UNA INTERRUPCIÓN)

Sólo aplicable contra personajes con ficha o PNJ'S

Función. Este comando interrumpe las llamadas al procesador que haya hecho un programa hasta el final de la escena. El Tiempo de CPU de la víctima no pierde los puntos que tenga marcados, pero los éxitos obtenidos en la reserva no cuentan para su tirada aunque sí el valor absoluto para saber si la reserva domina. A efectos de juego inutiliza la ventaja que supone contar con los recursos de la CPU para obtener éxitos, dejando sólo sus posibles efectos adversos.

Dificultad. Obtener un número de éxitos igual o mayor que el nivel de la víctima más uno por cada punto que tenga asignado a su CPU. Nivel + Puntuación actual de CPU.

Tiempo de procesador mínimo necesario. 3

Memoria mínima necesaria. 3

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Inmediato.

Nota: Sólo contra personajes con ficha o PNJ'S

/LOAD (CARGAR)

Función. Permite cargar una copia de respaldo **siempre que haya alguna disponible**. Si el programa no tiene ninguna el comando no tiene ningún efecto. Los nuevos datos de la copia sustituyen a la que la invocó, renovándose con toda su integridad y corrupción intacta. Sin embargo los demás efectos que haya sufrido el programa cargado previamente afectan también a la nueva copia, como la pérdida permanente de Operaciones o los puntos almacenados en la CPU y la Memoria.

La mala noticia es que cada vez que se ejecuta este comando se pierden los datos del programa que se pretende sustituir, pues no se registran los cambios. A efectos de juego el programa pierde una de sus copias de respaldo disponibles.

El proceso de carga es lento, por lo que será necesario contar con algo de tiempo para que la carga se realice correctamente.

Dificultad. 12

Tiempo de procesador mínimo necesario. 6

Memoria mínima necesaria. 6

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Una escena. El proceso de carga se completa al finalizar la escena.

/LOC (LOCALIZAR POSICIÓN)

Función. Averigua y almacena las coordenadas en las que se encuentra el programa para poder consultarlas.

Dificultad. 2

Tiempo de procesador mínimo necesario. 0

Memoria mínima necesaria. 0

Tiempo entre ejecuciones. Un asalto.

Tiempo de ejecución. Inmediato.

/MEMORY DUMP (VOLCADO DE MEMORIA)

Función. El programa vuelca toda la información de la que está compuesto en las unidades de almacenamiento del Sistema. También es posible hacerlo en otras fuera del sistema en el que se encuentra en el caso de que estén disponibles. El volcado realiza una copia del programa en las unidades y acto seguido lo cierra, eliminándolo completamente de la memoria. Para poder reiniciar es imprescindible la intervención de un usuario, que deberá conocer su ubicación y la clave de seguridad para poderlo reactivar.

Obviamente este comando supone un gran riesgo. El programa puede quedar fuera de juego si no dispone de un usuario en el que pueda confiar. Este comando es útil para poder escapar, salvando así su integridad en caso de emergencia o para evitar un nivel de corrupción tal que lo dañe sin remedio. Una vez almacenado un usuario con conocimientos puede repararlo o alterar su código a partir de la copia.

Mientras se realiza el volcado el programa no puede estar inmerso en ningún conflicto que requiera su atención.

Dificultad. Nivel actual del programa.

Tiempo de procesador mínimo necesario. 4

Memoria mínima necesaria. 4

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Una escena. El volcado se produce al finalizar la escena en curso.

/PURGE CACHE (BORRADO DEL BÚFER)

Sólo aplicable contra personajes con ficha o PNJ'S

Función. Este comando borra por completo los datos almacenados en el Búfer de un programa.

Dificultad. Obtener un número de éxitos igual o mayor que el nivel de la víctima más uno por cada punto almacenado en el búfer. Nivel + Puntos en el Búfer.

Tiempo de procesador mínimo necesario. 3

Memoria mínima necesaria. 3

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Inmediato.

Nota: Sólo contra personajes con ficha o PNJ'S

/PURGE MEMORY (VACIAR LA MEMORIA)

Sólo aplicable contra personajes con ficha o PNJ'S

Función. Este comando interrumpe el acceso a la Memoria impidiendo que un programa pueda usarla. El efecto dura hasta el final de la escena. La Memoria no pierde los puntos que tenga señalados, pero los éxitos obtenidos por la víctima en su reserva **no cuentan para su tirada**, aunque sí el valor absoluto para saber si la reserva domina. A efectos de juego inutiliza la ventaja de la Memoria para obtener éxitos dejando sólo sus posibles efectos adversos.

Dificultad. Obtener un número de éxitos igual o mayor que el nivel de la víctima más uno por cada punto que tenga asignado a la Memoria. Nivel + Puntuación actual en Memoria.

Tiempo de procesador mínimo necesario. 3

Memoria mínima necesaria. 3

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Inmediato.

Nota: Sólo contra personajes con ficha o PNJS

/STANDBY (MODO DE DESCANSO)

Función. Esta función está disponible solamente en el caso de que el programa ocupe una configuración de hardware en la Realidad Básica. Puede tratarse del chasis de un androide, un cuerpo sintético artificial o una configuración estructurada de nanomáquinas. Su ejecución permite poner en modo de suspensión al soporte para ahorrar energía y comenzar la recuperación de la integridad del código. Todos los daños registrados en las casillas de corrupción avanzan una posición hacia la izquierda. Este desplazamiento sólo es válido la primera vez. Intentarlo más veces no tiene efecto hasta que el programa se recupere del todo o vuelva a recibir daños. Los desperfectos sufridos en el hardware deben repararse por otros medios.

Una configuración de hardware en modo de descanso queda indefensa, no obstante siempre puede contar con su modo de vigilancia o Standby, que le previene en caso de que se aproxime alguna amenaza. No obstante para percibirla es necesario realizar una prueba de Control. Cualquier Rutina de Alerta será una gran ayuda. La dificultad de la tirada depende de las condiciones del mundo físico y de la naturaleza de la amenaza.

Dificultad. 1.

Tiempo de procesador mínimo necesario. 0

Memoria mínima necesaria. 0

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Un asalto.

/SEND (ENVIAR)

Función. Permite enviar un mensaje de texto, audio o video (o más sentidos en el caso de que la ambientación lo contemple) a cualquier dispositivo de

salida en la Realidad Básica que permita su consulta e impresión como teléfonos inteligentes, tabletas de datos, impresoras 3D, fotocopadoras, faxes, interfaces de conexión, puntos de acceso, terminales, etc.

Dificultad. 1 o más. Algunos dispositivos pueden dar problemas por lo que puede aumentar en uno o dos puntos.

Tiempo de procesador mínimo necesario. 0

Memoria mínima necesaria. 0

Tiempo entre ejecuciones. Un asalto.

Tiempo de ejecución. Inmediato.

/SHOUT (GRITAR)

Función. Funciona como el comando /TELL, pero en este caso el mensaje puede ser escuchado en un área bastante amplia alrededor del personaje que lo ejecuta. Obviamente para ser escuchado hay que decir algo...

En muchos sistemas está prohibido el uso de este comando. No supone una falta grave pero su uso puede alertar a los agentes del sistema, que acudirán a investigar.

Dificultad. 1

Tiempo de procesador mínimo necesario. 0

Memoria mínima necesaria. 0

Tiempo entre ejecuciones. Un asalto.

Tiempo de ejecución. Inmediato.

/SLEEP (DORMIR O HIBERNAR)

Función. Fuerza a una víctima a entrar en modo de **hibernación**. El programa queda desactivado por completo, incluyendo su modo Standby. Su código continúa cargado en la memoria pero las CPU del sistema lo

desatienden. Un programa forzado a desactivarse de este modo queda indefenso y es incapaz de percibir amenazas. El efecto dura hasta el fin de la escena en curso. Con el cambio de escena se asume que el programa vuelve a activarse y a funcionar con normalidad.

Dificultad. Igualar o superar el doble del nivel de la víctima (o su nivel de desafío).

Tiempo de procesador mínimo necesario. 2

Memoria mínima necesaria. 2

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Un asalto.



/SUSPEND (SUSPENDER)

Función. Pone en modo de suspensión al programa para acelerar la recuperación de la integridad del código. Todos los daños registrados en las casillas de corrupción avanzan una posición hacia la izquierda. Este desplazamiento sólo es válido la primera vez. Intentarlo más veces no tiene efecto hasta que el programa se recupere del todo o vuelva a recibir daños. A diferencia de /Standby, este comando sólo se puede aplicar sobre programas.

Un programa en modo suspendido queda indefenso, no obstante siempre cuenta con un modo de vigilancia o Standby, que le permite comprobar el entorno en el caso de que se aproxime alguna amenaza.

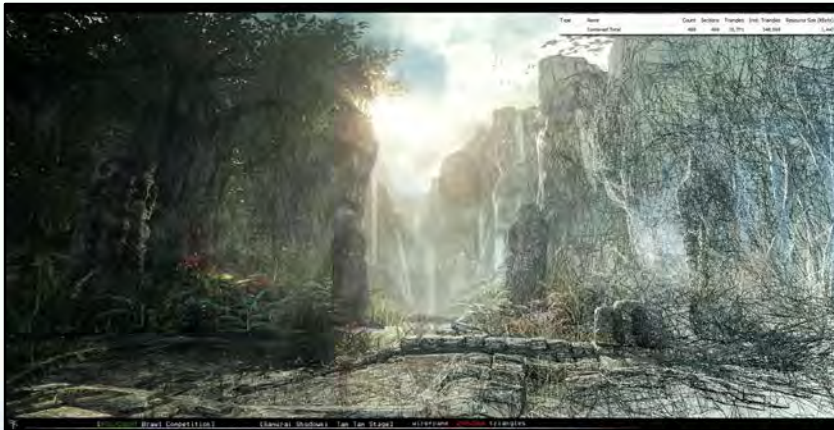
Dificultad. 1

Tiempo de procesador mínimo necesario. 0

Memoria mínima necesaria. 0

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Un asalto.



/TELL (DECIR)

Función. Este comando permite enviar un mensaje a otro personaje siempre que se encuentre en el mismo sistema. Obviamente para ser escuchado hay que decir algo por lo que junto al comando se añade el mensaje en cuestión. La longitud del mensaje puede ser tan larga como se desee y es posible enviar datos de cualquier tipo.

Dificultad. 1

Tiempo de procesador mínimo necesario. 0

Memoria mínima necesaria. 0

Tiempo entre ejecuciones. Un asalto.

Tiempo de ejecución. Inmediato.

/TRANSFER (TRANSFERENCIA)

Función. Permite transferir todos los datos del programa incluyendo sus copias de respaldo que estén almacenadas en las unidades del Sistema en otra unidad de almacenamiento que se encuentre disponible en cualquier punto de la Realidad Básica. Esta unidad normalmente queda fuera del área de influencia del Sistema en el que se encuentra el programa. La transferencia permite poner a salvo el número de copias que se quiera guardar en ella. Siempre que tenga acceso a la unidad puede volver a recurrir a sus copias cuando desee.

Mientras se realiza la transferencia el programa no puede estar inmerso en ningún conflicto que requiera su atención.

Dificultad. El nivel actual del programa.

Tiempo de procesador mínimo necesario. 2

Memoria mínima necesaria. 2

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Una escena. La transferencia termina al finalizar la escena en curso.

/VICTORY (FANFARRIA DE LA VICTORIA)

Función. Este comando fue inventado por un usuario muy aficionado a los juegos aunque incapaz de saber ganar y mucho menos a perder. Envía una señal por todo el sistema que anuncia su victoria en caso de haber ganado alguna contienda. En los “cielos” del espacio virtual (de haberlos) se aprecia

un efecto gráfico similar a los fuegos artificiales mientras un enorme rótulo centelleante anuncia la victoria y el nombre del ganador. Este efecto va acompañado de una alegre musiquilla, o **fanfarria de la victoria**, de los más apropiada (NdE: ...pues sí, a lo “Final Fantasy”).

En un entorno esquemático realista o similar todos los programas reciben en su lugar un mensaje acompañado de la alegre tonadilla (NdE: de estilo 8 bits por favor) informando del suceso. El mensaje puede ser visto y escuchado en todas las áreas del sistema y difícilmente pasará desapercibido.

La buena noticia es que es posible desactivar los avisos de este comando. Basta con introducir el comando seguido de la cadena: “/Victory mode off”. Aunque por razones que aún no se explican la dificultad de hacerlo se duplica. También se puede utilizar este comando para anunciar una derrota, pero que se recuerde, este uso no se ha dado más que en unos pocos casos.

Dificultad. 3 éxitos (6 para mantenerlo desconectado).

Tiempo de procesador mínimo necesario. 0

Memoria mínima necesaria. 1

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Inmediato. El efecto dura el resto de la escena.

/ZOMBIE MODE (MODO ZOMBIE)

Función. Este poderoso comando es capaz de volver a reintegrar temporalmente el código de un programa que haya sido destruido o borrado. El efecto no es duradero por lo que más vale aprovechar los efectos mientras duren. El programa en modo zombi aparecerá muy deteriorado ya que la reinstauración nunca es perfecta por lo que además de tener un Avatar en condiciones penosas sufrirá muchos fallos de funcionamiento. En términos de juego el programa puede usar todas sus funciones pero su Clase efectiva se reduce a IV (4). Por lo tanto, utiliza dados D4 para todas sus Operaciones.

La otra mala noticia es que el programa que ha sido reintegrado no diferenciará entre aliados o enemigos por lo que reaccionará de forma ofensiva

con el que esté más próximo. Al final de la escena el programa vuelve a desintegrarse y ya no puede ser restaurado de nuevo. Descansará en paz finalmente...

Dificultad. Nivel del programa destruido que se pretende “levantar”.

Tiempo de procesador mínimo necesario. 4

Memoria mínima necesaria. 6

Tiempo entre ejecuciones. Una escena.

Tiempo de ejecución. Inmediato. El efecto dura hasta el final de la escena en curso. Después los bits del programa “resucitado” vuelven a disiparse. Ya no es posible volverlo a reintegrar.





Scroll

JUEGO DE ROL EN EL UNIVERSO DIGITAL

DESARROLLO DISEÑO DE SISTEMAS

FUNCIÓN B

"UNA RED DE SISTEMAS"

"No hay lugar como 127.0.0.1"

Anónimo

Scroll propone un momento y un lugar como punto de partida para su ambientación "oficial". Una realidad situada a mediados del siglo XXI en el mundo de La Tierra. Se trata de una propuesta para disponer de un arranque desde el que comenzar a elaborar las tramas para las aventuras. Pero ya desde los comienzos de este documento he comentado en más de una ocasión que los jugadores pueden crear la realidad de su mundo y ubicarla donde quieran. Algo que no se debe más que al deseo de querer animarlos a buscar la que prefieran.

En esta propuesta, la red que conecta todos los sistemas informáticos de la Tierra forman un entramado tan complejo que es ya imposible registrar la cantidad exacta de dispositivos activos que se encuentran intercambiando información al mismo tiempo. Mucho menos sencillo resulta adivinar la cantidad de líneas que los unen unos con otros formando canales de comunicación. El nivel de complejidad es tan vasto que todo el conjunto forma un sistema nervioso que se asemeja al de un cerebro humano. Muchos especialistas aseguran que ya se ha llegado a superarlo a varios niveles. Y no son pocos para los que este conocimiento supone un motivo de preocupación.

En la **página siguiente** hay una imagen que permite hacerse una idea del nivel de hipercomplejidad de este universo digital. La ilustración muestra el flujo de información a finales de la primera década del siglo XXI. En el momento en el que tiene lugar el universo de Scroll, varias décadas después, su nivel de complejidad se ha multiplicado por un factor de diez.

Lo cierto es que la similitud con la de un cerebro se aproxima mucho a la verdad. Cada dispositivo electrónico capaz de recibir y entregar información

cumple la misma función que una neurona, componiendo así la unidad en el control de los procesos de la información. También es capaz de almacenar esa información de la misma forma que lo hace en un sistema nervioso basado en el Carbono. Al mismo tiempo las líneas de datos componen sus sinapsis permitiendo el intercambio de información con sus vecinas en un complejo

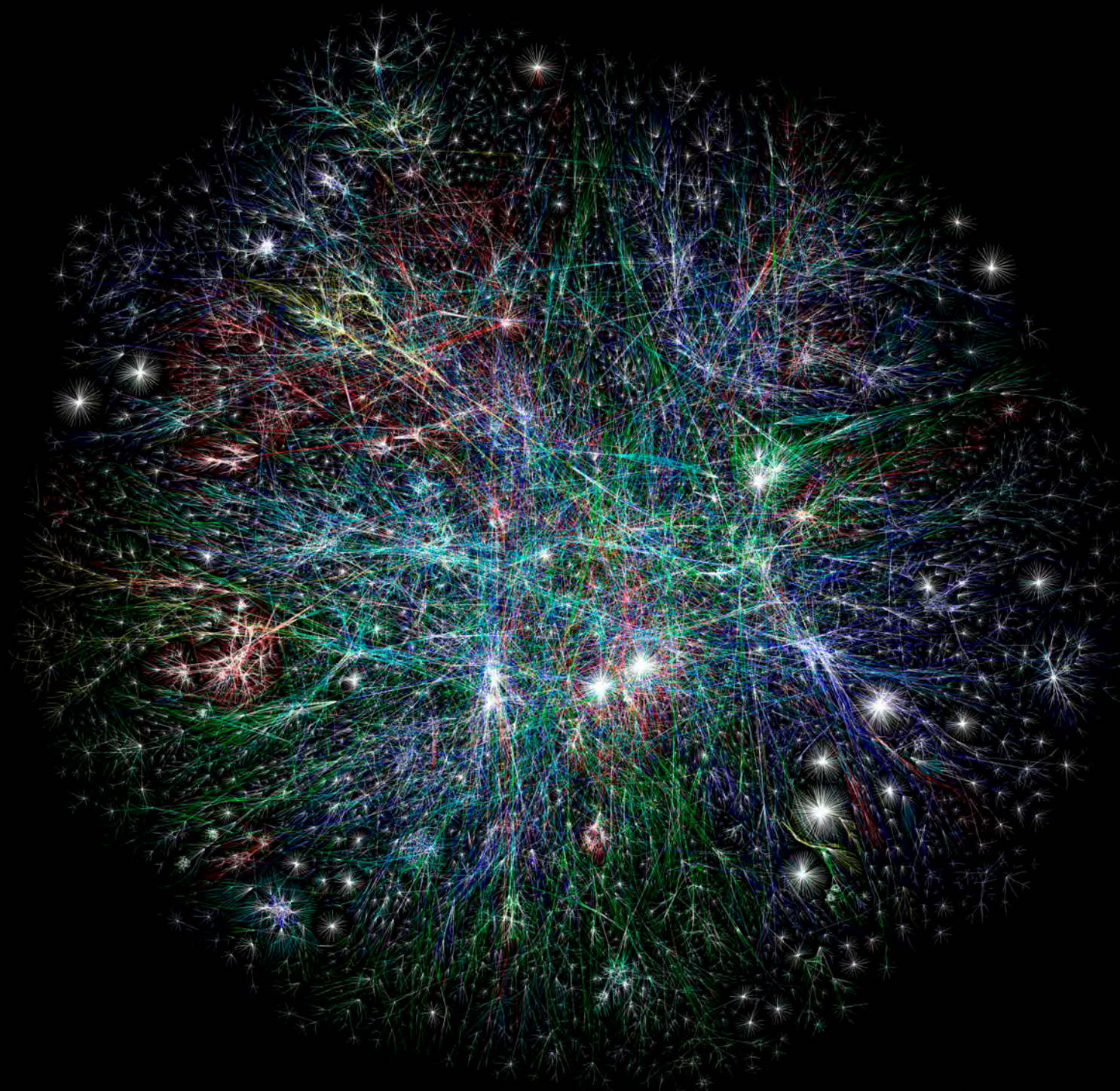


proceso que nunca parece tener fin. Día y noche este entramado gigantesco recibe información, lo procesa y lo entrega sin cuestionarse, en teoría, el volumen de trabajo que está resolviendo cada segundo, y mucho menos sus razones para querer hacerlo.

Las líneas de comunicaciones recorren el planeta Tierra describiendo una danza sin fin a la velocidad de la luz. Mientras un lado del planeta duerme el otro

despierta. Pero en el universo digital no existe ni el día ni la noche, no hay sueño ni vigilia, tan sólo la noche más larga. Una luna nueva eterna e inmutable ilumina a las criaturas nacidas en el medio digital siendo éstas las únicas que pueden percibir sus destellos. Son luces eléctricas, invisibles para los humanos, surgidas de todo cuanto significa o ha significado alguna vez el conocimiento.

Y en su aparente complacencia, los seres inteligentes que los han creado siguen con sus vidas despreocupadas en la Realidad Básica. Ajenos al hecho de que es suficiente con un entramado de comunicación finito y una masa crítica de información para que con un pequeño soplo de inspiración surja el despertar de la consciencia.



ASPECTO DE LA RED GLOBAL A PRINCIPIOS DEL SIGLO XXI

SISTEMAS Y SUBSISTEMAS

¿QUÉ ES UN SISTEMA?

Cada dispositivo dotado de un entorno digital capaz de ejecutar programas **forma un sistema**. A su vez, cada dispositivo conectado a una red que le permita comunicarse con otros da lugar a que existan cientos de millones de sistemas distintos interconectados entre sí. En Scroll se denomina “**Sistema**” al entorno digital en el que se encuentre un programa. La interconexión de todos esos sistemas a través de una red da lugar a la Red Global de Sistemas o **Sistema Global**.

(...) “*Nuestros sistemas operativos están basados en metáforas, y, por lo que a mí respecta, es legítimo cuestionar cualquier cosa con metáforas dentro.*”

Neal Stephenson

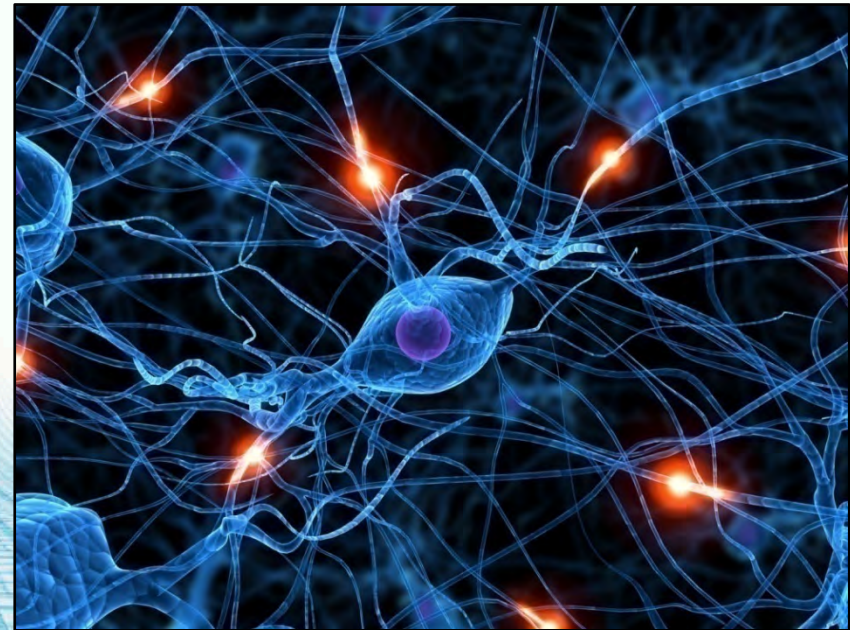
Desde una máquina expendedora de refrescos monitorizada vía red hasta los sistemas de navegación de una aeronave de última generación, todos forman parte del universo digital. Algunos son independientes y solitarios; otros componen subsistemas, funcionando como sistemas dentro de sistemas, y muchos más agrupan a su vez conjuntos de múltiples dispositivos con el fin de crear un único sistema mucho más potente. Por eso en Scroll identificar una configuración de hardware o un dispositivo concreto, a no ser que se haga con un objetivo, no es relevante. Los programas pueden viajar por la red cruzando de sistema a sistema al margen del hardware que les permite mantenerse en funcionamiento.

Todo dispositivo capaz de ejecutar programas forma un sistema. El entorno digital en el que se encuentra un programa en un momento dado se denomina “**El Sistema**” en Scroll.

El Sistema Global o “Red Global de Sistemas” lo componen todos los sistemas que se encuentren interconectados a través de la red.

Sinapsis neuronales de silicio

Cada sistema capaz de comunicarse con los demás forma un nodo del que parten las líneas que sirven de sinapsis neuronal, uniéndolo con los nodos circundantes del mismo modo que lo hacen las neuronas en un cerebro humano. Así se forma un vasto tejido de sistemas interconectados en red, cada uno con sus propias leyes, su estructura, su forma de representación, sus secretos y en algunos casos con sus propias criaturas digitales. Algunas de ellas, muy pocas, sólo en un momento reciente han comenzado a soñar por primera vez.



La red está compuesta por todo lo viejo y lo nuevo. Desde la última tecnología punta hasta los equipos más obsoletos. Aunque muchos utilizan una forma consensuada de organizar y describir la información, otros pueden tener su propio método para hacerlo. Cada sistema se adapta al hardware del que dispone y a las ideas y puntos de vista de quienes lo crearon. Los soportes de hardware más antiguos usan representaciones esquemáticas o abstractas

basadas en iconos para sus entornos de operaciones. Los más sofisticados permiten ejecutar representaciones virtuales hiperdetalladas, lo que da soporte a los juegos de última generación y a los entornos de realidad virtual más avanzados.

La posibilidad de que todos esos sistemas sean capaces de comunicarse entre sí da lugar a una red en la que todo es posible. El Sistema Global, producto de la unión de todos ellos, estructura una vasta red que en conjunto forma un único cerebro de tal magnitud y capacidad de proceso que en la actualidad sólo muy pocos usuarios son conscientes de sus auténticas posibilidades. Pero para lograr que funcione de forma eficaz como un único hipercomputador es necesario que los millones de sistemas que lo componen, es decir sus “neuronas”, se pongan a trabajar de forma organizada usando un mismo lenguaje. Los usuarios están convencidos de que esto aún no se ha conseguido y que aún falta mucho tiempo para que suceda por lo que la mayoría descarta contemplar esta posibilidad. Pero lo que esta mayoría desconoce es que un destello de consciencia ha surgido en la Red Global de Sistemas tras haber alcanzado ésta un nivel tan alto de complejidad. Esta consciencia ha sido capaz de reestructurarse a sí misma sin la intervención del usuario, organizando sus recursos y conduciendo los flujos de información con el fin de mejorar su eficacia y evolucionar como entidad por sí sola. Entre otros efectos, esto ha supuesto la aparición de un reflejo de esa consciencia capaz de hablar un único lenguaje y actuar en su nombre como una sola voz. Sí, lo has adivinado, esta voz no es otra que **La Libélula**.

EL ENFOQUE HÍBRIDO DE SCROLL

Con tantos enfoques distintos en Scroll es posible concebir cualquier tipo de escenario, algo que aquí también llamo “ambientación”. Puedes llamarlo como prefieras, ambientación o escenario son ambos la misma cosa: distintas formas de imaginar el mundo de los programas. Este juego plantea reconciliar todos los enfoques posibles para formar una vasta red de escenarios interconectados. Tanto los que se detallan en el apartado siguiente como todos aquellos que puedan imaginar los jugadores. De esta forma es posible concebir un universo de sistemas diferentes, distintas visiones del gran teatro digital. Porque al final

todos juntos lo que componen son realidades distintas. Cada sistema sigue siendo independiente y dispone de autonomía. Pero todos ellos a la vez, en muchos casos sin saberlo, componen las partes de una entidad única mucho mayor capaz de tomar sus propias decisiones, ajenas casi siempre a los intereses de cada sistema y a los de quienes los crearon.

En Scroll los personajes pueden viajar de un sistema a otro a través de las líneas de comunicación para explorar todos los sistemas activos. Podrían nacer por ejemplo en el potente sistema de una gran empresa compuesto por una gran red de servidores para acceder más tarde a los sistemas de una red privada de un gobierno cualquiera. La red de esta empresa forma un sistema independiente y es posible representarlo recurriendo a un enfoque determinado de los que se describen en este capítulo. Al acceder al sistema del gobierno podrían encontrarse en un escenario completamente distinto. Ten en cuenta que el funcionamiento de todos estos sistemas no importa demasiado. El enfoque o forma de imaginar un sistema funciona como una capa de abstracción que oculta su funcionamiento real (si exceptuamos en cierta medida el “esquemático realista”). Cuando el programa se adentra en un sistema nuevo su escenario puede transformarse en algo diferente.

En el apartado siguiente, donde se explican algunos de los enfoques para los escenarios, se ofrecen detalles a tener en cuenta en el caso de querer utilizar esta idea. Esta es la aproximación híbrida que ofrece Scroll y la que propone como una forma de entender el mundo digital. Un “no lugar” demasiado amplio para limitarlo a una sola visión, por lo que cualquier concepto imaginable puede tener su sitio.

DEFINIENDO EL ESCENARIO

En este apartado se explican algunos de los escenarios con los que es posible concebir un sistema. Todos ellos se encuentran interconectados por la red de comunicaciones y todos o casi todos son accesibles a través de ella. Es posible no obstante que para poder entrar en un sistema determinado sea necesario tener que sobrepasar algún sistema de seguridad o cumplir una condición; algo que puede ser algo tan simple como tener permiso por ejemplo. Lo que en

términos de juego no es más que un obstáculo que hay que superar por el medio que sea.

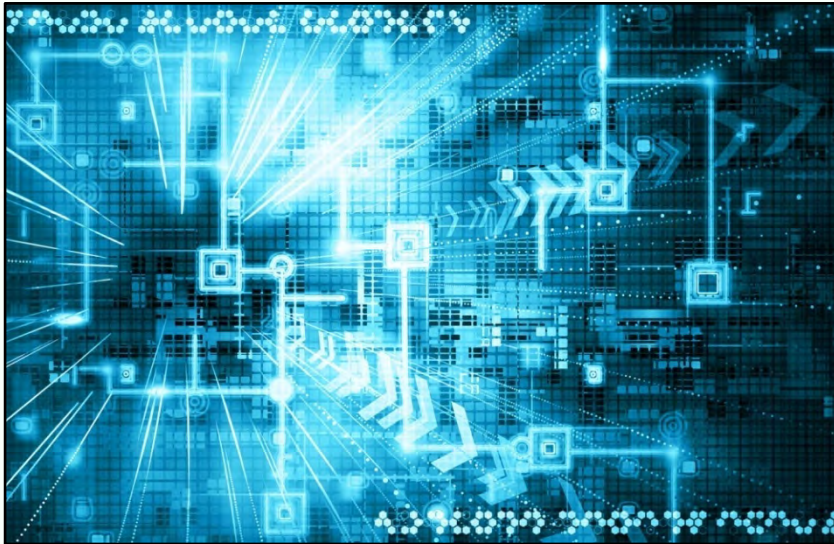
En cada enfoque se detallan sus características más importantes y algunos puntos que hay que tener en cuenta. Del mismo modo se ofrecen algunos consejos que pueden ayudarte a describir su funcionamiento y cómo evolucionan en ellos los programas. Cada escenario propone también una serie de bloques de reglas que es conveniente utilizar para que funcione mejor con Scroll. Ten en cuenta que algunas de esas reglas se han creado precisamente con ese propósito.

Para describir el enfoque y las premisas de cada escenario se sigue un orden que más adelante puede resultar muy útil para definir otros sistemas que se deseen crear para el juego. En el caso de que quieras crear alguno, contesta cada uno de los puntos que se describen y ponle un nombre a tu creación. Es cuanto necesitas para crear un nuevo modo de concebir tu mundo digital.

- **Nombre o tipo de escenario.** Un nombre que describe el enfoque del escenario. Por ejemplo enfoque esquemático, mundo virtual ultradetallado, videojuego de acción en tres dimensiones, etc.
- **Descripción.** Donde se describe el tipo de escenario.
- **Referencias.** En este apartado se incluyen algunas referencias a obras como películas, novelas o comics donde se usa o se hace una aproximación al enfoque que se describe.
- **Tema del escenario.** De contar con esta posibilidad se define el tema sobre el que se diseña el escenario. Un mundo virtual por ejemplo puede estar basado en un entorno de fantasía, de ciencia ficción, de cine negro, etc. El tema no es aplicable a todos los enfoques pero sí que es posible encontrarlo en muchos de ellos.
- **Usos.** Se describe qué tipo de partidas pueden beneficiarse de este tipo de escenario.

- **Resolución de conflictos.** Se describen algunas pautas de cómo pueden resolverse las operaciones en el escenario.
- **Avatares.** Si los personajes disponen o no de alguna forma de representación.
- **Tipo de espacio.** Se describe si existe algún tipo de espacio abstracto definido matemáticamente. Lo más común es que se haga mediante ejes de coordenadas.
- **Uso de reglas simplificadas.** Se detalla en cada caso si es posible o incluso conveniente utilizar las reglas simplificadas.
- **Acceso a copias:** Se describe si es conveniente o no el uso de las copias de seguridad (backups) de los programas.
- **Reglas de integridad.** Se indica si se puede o si es recomendable el uso de las reglas de integridad del código. Han sido diseñadas para algunas ambientaciones por lo que en esos casos será lo más aconsejable.
- **Módulo de inventario.** Se indica si es recomendable contar con un inventario. Estas reglas han sido diseñadas para algunas ambientaciones por lo que en esos casos es conveniente usarlas.
- **Rutinas recomendadas.** Se indican algunas Rutinas que pueden ser útiles o que encajan en ese tipo de escenario.
- **Utilidades recomendadas.** Se indican algunas Utilidades que encajan en ese tipo de escenario.

1. ENFOQUE ESQUEMÁTICO Y REALISTA



Descripción

En el enfoque esquemático se prescinde de cualquier representación que vaya más allá de una serie de diagramas que ayuden a entender cómo funciona el Sistema. Su uso es funcional y no se recurre a ninguna metáfora visual para representarlo. Lo importante de este enfoque es tratar entender cómo se configura y su funcionamiento para poder actuar en él.

En este modo de concebir el entorno digital es posible recurrir a esquemas simplificados de sistemas informáticos auténticos que describan su funcionamiento. Aunque no es necesario disponer de grandes conocimientos de informática, sí es cierto que los jugadores que dispongan de cierta base pueden sacarle más partido a este planteamiento. Incluso es muy posible que muchos de ellos prefieran recurrir a este enfoque ya que al disponer de una base de conocimiento más amplio el uso de metáforas o abstracciones les pueda resultar extraño, no lo consideren necesario o no les divierta.

En el enfoque esquemático se busca el realismo para simular los sistemas, pero no es necesario ceñirse de forma estricta a la realidad de tal modo que se resienta la experiencia de juego. Si resultan demasiado complejos en un afán de representarlos tal como son algunos jugadores no sabrán qué hacer. Muchos conceptos informáticos requieren de un estudio que está mucho más allá de las intenciones de este juego. Por esta razón se recomienda simplificarlo en lo posible y usar solamente los conceptos básicos que sirvan para los propósitos de las aventuras. Por ejemplo, un jugador no necesita entender cómo funciona un cortafuegos. Para resolver las acciones le basta con saber que suponen un obstáculo que hay que superar.

Referencias

Aunque en muchos casos se hacen ciertas concesiones y se recurre a metáforas que rozan el enfoque simbólico abstracto (el siguiente enfoque descrito) para explicar a la audiencia lo que pretenden hacer los protagonistas, películas como "La red", "Enemigo público", "Los fisgones", "Misión imposible (1996)" y "Hackers" utilizan un enfoque esquemático realista para representar la red y los procesos informáticos.

Usos

El enfoque esquemático es ideal para representar operaciones en el entorno informático del mundo actual o del futuro próximo por lo que está orientado a resolver los ataques a sistemas de ordenadores, bases o redes de datos de cualquier clase como parte del desarrollo de una narración.

Es adecuado si lo que se desea es utilizar Scroll para describir las operaciones en el mundo digital tal y como lo conocemos hoy en día, ya sea en este o en cualquier otro juego de rol. En el segundo caso los jugadores pueden utilizar Scroll como un sistema de reglas que complemente a su propio juego, expandiendo sus posibilidades y ofreciendo alternativas para simular las operaciones. Así mismo es posible recurrir a este tipo de escenarios cuando los personajes en el mundo de Scroll accedan a sistemas muy antiguos, con sistemas operativos que hayan quedado obsoletos o a los sistemas específicos que se encuentran dentro de algunos dispositivos de hardware. Por lo general las operaciones con el enfoque esquemático se resuelven con rapidez, lo que

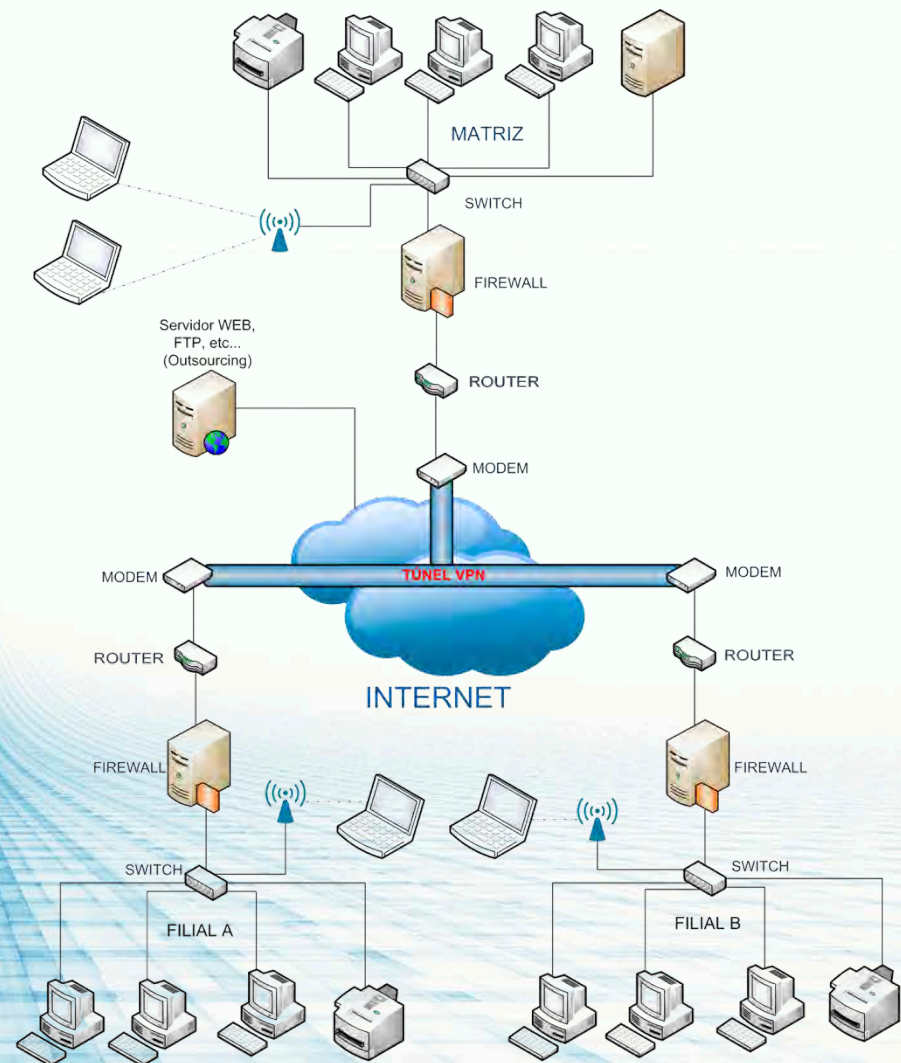
es muy conveniente si lo que se desea es servir de complemento en otros juegos o se recurre a él desde los demás escenarios.

Resolución

En el diagrama que se muestra en esta página tienes un ejemplo de lo que se pretende conseguir mediante este enfoque. Se trata del esquema real de la red privada de una empresa que se usará como ejemplo no sólo en éste tipo de escenario sino también en algunos de los siguientes para poder comparar las distintas perspectivas. Aunque aquí se recurre a una maniobra de incursión como un ejemplo, ésta no tiene que ser la única aproximación a este tipo de escenario. En realidad sólo es otra de las muchas maneras de entender cómo se organizan los componentes de los distintos sistemas.

En su configuración se disponen tres vías que conducen a distintas ramificaciones de la red: una para la sede central llamada aquí "matriz" y otras dos para sus sucursales. Todo está conectado a su vez a Internet, que en este caso es el Sistema Global. Si alguien quisiera acceder a un dispositivo o a un ordenador en particular de esta red por cualquier razón, la incursión comienza accediendo desde Internet al canal VPN, que es donde empieza el tramo de la red privada de la empresa. Suele tratarse de un canal con un gran ancho de banda hechos de fibra óptica por los que por él puede viajar un gran volumen de datos.

Los programas que recorran el canal pueden encontrar las primeras dificultades al encontrarse con los sistemas de protección que posea el software del router, lo que en términos de juego puede suponer, por ejemplo, realizar una prueba o incluso un combate para poder atravesarlo con éxito. Más allá del router se encuentra el primer gran desafío en forma de un cortafuegos. Superarlo debería ser una prueba compleja que requiera algo más desafiante y complicado que resolverlo con una sola tirada. Aquí es donde las pruebas con estructuras distintas o de nuevo el combate, que al fin y al cabo es una forma de resolver los problemas, pueden resultar muy útiles.



Conviene recordar que no todo tiene porqué ser solucionado por la fuerza bruta. Es posible que los personajes pudieran "negociar" con todo cuanto aquí se entienda como un obstáculo que haya que superar. Quizás el software del cortafuegos desee algo a cambio de conceder el paso libre...

Pero atravesar el cortafuegos no significa la victoria pues aún es necesario encontrar el camino correcto a través del Switch que conduzca a la terminal que se está buscando. Los cruces de caminos de este tipo son un buen momento para introducir algún tipo de puzzle o de enigma, un recurso que se puede usar siempre en todos los tipos de escenario.

Por último aún queda obtener el acceso a la terminal donde se esconde lo que se esté buscando. Todas las terminales disponen de una clave de acceso que si el usuario se ha trabajado no tiene porqué ser nada fácil descubrir, aunque lo más común es que se trate de algo bastante absurdo y obvio. Algo que depende del nivel de implicación del usuario con su trabajo y, por supuesto, de ¡su nivel de inteligencia! Incluir un acertijo como una forma de resolver las claves de acceso en el juego, además de una sugerencia, es sólo una de las muchas opciones de las que dispones para simular su resolución.

Una vez el programa logra introducirse en la terminal se abre ante él otro sistema en el que puede encontrarse con cualquier cosa, desde amenazas que actúan en su nombre como sus defensas hasta la posibilidad de toparse con aliados inesperados.

Tema del escenario

Por su propia definición y concepto, no es posible definir temas en los escenarios del enfoque esquemático. No obstante puede tener un tono tal y como muchas obras relacionadas con el mundo digital lo han hecho. El entorno realista es un lugar perfecto como complemento de historias de espionaje y robo de información.

Avatares

Por sus características en el escenario esquemático no se utilizan avatares ni representación alguna de los programas. No existen formas, representación gráfica o espacios abstractos definidos por lo que no tendrían sentido.

Tipo de espacio

No existe espacio ni formas definidas por lo que no se utiliza ningún sistema de movimiento.

Uso de reglas simplificadas

Es posible utilizar las reglas simplificadas para simular operaciones sencillas, especialmente si se utiliza Scroll en otros juegos de rol. Prescindir de algunos componentes simplifica las mecánicas del juego y acelera la resolución de los conflictos.

Acceso a copias

No se utilizan. Aunque desde un punto de vista realista sería posible contar con copias de seguridad, debido a su planteamiento este enfoque no lo contempla. Los programas disponen de una sola oportunidad. Aunque es posible contar con ellas si todos los jugadores están de acuerdo.

Reglas de integridad

No se utilizan. Es más, se desaconsejan por el tipo de planteamiento.

Módulo de inventario

No. Por lo general el módulo de inventario no tiene sentido en el enfoque esquemático ya que para empezar no se usan elementos o ítems.

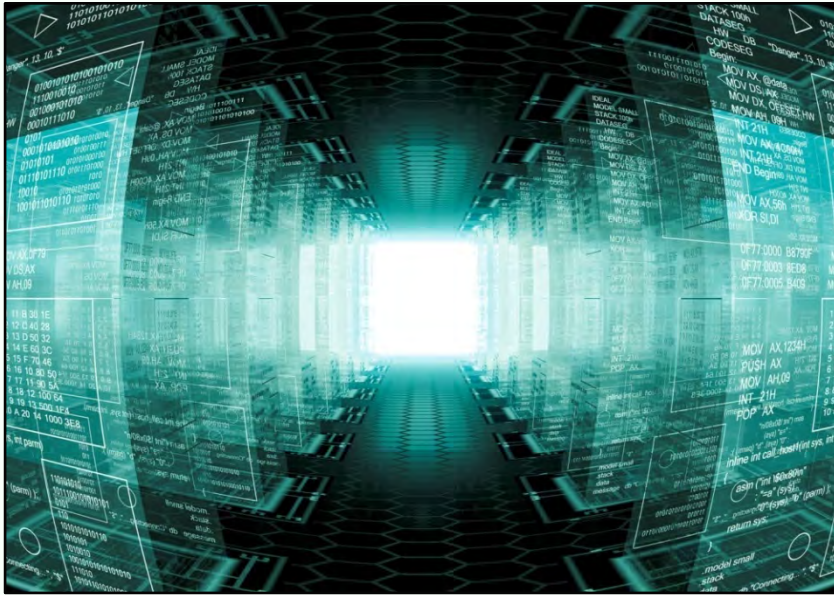
Rutinas recomendadas

Acceso a bases de datos, algoritmos de búsqueda, descifrar código, destruir o bloquear amenaza, manipular o romper código, control de sistemas remotos. Las rutinas de combate representan la capacidad del programa para superar todo cuanto pueda suponer un obstáculo o amenaza.

Utilidades recomendadas

Cuarentena, devorador de código o gusano, perforadora, mutación, réplica. Del mismo modo que en las rutinas, las utilidades ofensivas reflejan la potencia del programa para superar a todo cuanto suponga un obstáculo o amenaza.

2. ENFOQUE SIMBÓLICO ABSTRACTO



Descripción

El escenario simbólico abstracto, también denominado en este juego “Espacio de Gibson”, recurre al uso constante del **símbolo** dentro de un espacio en dos o tres dimensiones para representar la información. Esta representación sigue en la mayoría de los casos el esquema básico visto en el enfoque anterior. La diferencia es que aquí se recurre al diseño de una interface gráfica (una representación visual) que oculte el esquema real de los sistemas tras **una capa de abstracción** (término explicado en el siguiente enfoque). Con esto se pretende que el control resulte mucho más intuitivo y fácil de asimilar, haciendo más amigable el entorno digital para el usuario medio. A partir de este enfoque las interfaces evolucionan hasta concebir la copia exacta de la realidad que significa un entorno virtual ultradetallado.

Cuando un programa accede a un sistema (a cualquiera de los descritos o que estén aún por concebir) se adapta a su nuevo entorno, lo que incluye la

posibilidad de disponer en algunos de una forma de representación o de avatar. El enfoque simbólico abstracto es el primer escenario donde resulta práctico (y coherente) contar con uno. De súbito los programas se encuentran inmersos en una proto-realidad virtual que los transforma en iconos y en símbolos. Siempre que se encuentren en el sistema tendrán que atenerse a las leyes que rigen esta forma de representación. Esta condición se hace extensible a todos los demás enfoques vistos en el juego.

Aunque aún es común encontrar en muchos dispositivos este tipo de entorno en dos dimensiones (muy popular a comienzos del siglo XXI), la interface de la mayoría de sistemas se basa en un espacio matemático tridimensional que se ha impuesto como un estándar. Los flujos de información transcurren por ese espacio como un reflejo de las mismas líneas físicas de datos tal y como se encuentren ubicadas en la Realidad Básica. Esto quiere decir que la posición física de una configuración de hardware (uno o múltiples ordenadores o dispositivos) en el mundo físico se refleja en el espacio abstracto siguiendo un patrón similar. Los grandes bloques representan a las empresas u organismos, y las líneas por las que circulan los datos se muestran como “autopistas” de información donde ésta viaja yendo y viniendo de la misma forma que lo hacen las luces de los coches por una autopista durante las horas nocturnas. Cada usuario conectado a la red por el medio que sea, ya sea un teléfono inteligente o una terminal de ordenador, se muestra como un pequeño icono que evoluciona siguiendo su misma trayectoria en el mundo físico. En resumen, el entorno digital es un “reflejo” abstracto de los sistemas digitales que estén activos en el mundo físico.

Una de las premisas de este método de representación del escenario digital es recurrir a la simplificación haciendo uso de un simbolismo abstracto y funcional. Los detalles son secundarios. Los elementos del escenario que forman los iconos intentan economizar recursos siendo lo más sencillos que sea posible. Esto permite liberar al sistema de tener que gastar sus recursos en hacer cálculos innecesarios. No se intenta simular la realidad pero se hace un gran esfuerzo porque los elementos resulten intuitivos. La meta es que nada más verlos un usuario pueda averiguar su función e intuir cómo se maneja el sistema para realizar sus operaciones y obtener lo que desea.

El uso masivo del icono lo convierte a éste en el principal elemento visual como unidad básica de representación. En muchos casos esto se debe a que en su planteamiento se tiene —o tenía— en cuenta poder manipular el entorno con dispositivos de interacción de poca sensibilidad como periféricos de control basados en el concepto del ratón, la pantalla táctil o la interfaz estrella, el guante de datos o "dataglove". Hoy estos dispositivos han quedado obsoletos, relevados por la versatilidad y facilidad de uso de los modernos interfaces neuronales (los más caros del mercado) que permiten manipularlos directamente con el pensamiento, o por la más común cámara-escáner capaz de posicionar las extremidades y expresiones faciales del usuario. A día de hoy es ya raro encontrar un dispositivo que no posea algún tipo de cámara de este tipo, siendo incluidos de serie en todos como su sistema básico de interacción. El usuario no tiene más que mover sus manos, que serán escaneadas por la cámara, para manipular los bloques de información, cogiéndolos, arrastrándolos y soltándolos como si lo hiciera en el medio físico.

Podemos decir que las interfaces que tenemos en la actualidad en nuestros teléfonos inteligentes, tabletas y ordenadores tienen un enfoque simbólico abstracto. Están compuestos de iconos intuitivos dispuestos sobre un escritorio en dos dimensiones. Los iconos siguen un estándar de diseño para que resulten reconocibles entre distintos sistemas. La interacción hoy por hoy se realiza mediante pantallas táctiles usando nuestros dedos. Incluso es posible seguir la trayectoria de cada usuario en el mundo físico mediante el posicionamiento GPS, lo que permitiría representar sobre el espacio abstracto sus evoluciones.

El catálogo de símbolos es finito y aunque no todos, los más representativos se acogen a una normativa ISO que regula su diseño con el fin de garantizar un estándar. Por esta razón la red es una forma de representación consensuada que es compartida por todos los usuarios. Una alucinación que ha sido creada mediante acuerdos y tratados entre las compañías y los gobiernos de diferentes países. Por supuesto, hay muchas excepciones por lo que sólo la mayoría de empresas o los organismos gubernamentales se acogen con más fidelidad a la normativa. Una asociación de carácter internacional formada por más de cincuenta países controla y regula que se cumpla el estándar.

Pero como no podía ser de otro modo, también existe la gran diversidad de aportaciones que realizan los usuarios a título individual. Algunos respetan la normativa pero muchos otros, por cientos de razones distintas, contribuyen con elementos visuales reconocibles, en muchos casos tratando de marcar una diferencia que los haga únicos.



Referencias

En Scroll esta forma de representación también recibe el nombre de: "Espacio de Gibson" en honor al escritor William Gibson de finales del siglo XX que popularizó esta visión de la red y de sus sistemas informáticos. Este enfoque es muy popular en el género Ciberpunk e inauguró términos como el "Netrunning". Algunas obras clave de este género que emplean este enfoque son novelas como "Neuromante", "Mona Lisa acelerada", "Conde cero", "Snow Crash", "Johnny Mnemonic", series míticas como "Max Headroom" o anime como "Serial experiments Lain". Otra propuesta cuya consulta es indispensable es la propuesta de los videojuegos "Rez" y "Child of Eden" donde el concepto alcanza la madurez y se convierte en obra de arte. Al final

de este documento dispones de una lista donde se citan estas obras y sus autores.

Usos

El sistema simbólico abstracto es una forma de representar el mundo digital que forma parte del legado de muchas obras de ficción y de muchos juegos de rol. Elegirla como forma de representación de los sistemas es en realidad una decisión personal en la que influyen más los gustos que tener que buscar razones para querer hacerlo. No obstante, por su propio concepto de diseño la visión simbólica abstracta puede facilitar la comprensión de los procesos informáticos y hacerlos muy intuitivos.



Emplea este enfoque si te gusta su concepto, si éste encaja con tu visión de los sistemas o si es el que describe tu juego de rol favorito, como los de género ciberpunk más clásicos (Ciberpunk o Shadowrun por ejemplo). También es recomendable recurrir a ella como una alternativa a los demás enfoques. Los programas pueden acceder a un sistema de este tipo encontrando algo nuevo y diferente que los jugadores no estaban esperando.

Resolución

Para resolver las acciones en este enfoque se recurre a otro punto de vista. Si en el enfoque esquemático los personajes siguen el esquema de funcionamiento de un sistema informático, en el simbólico abstracto se describen sus acciones en el contexto de un espacio en dos o tres dimensiones basado en ese mismo esquema. En términos generales se trata de traducir el diagrama de funcionamiento en elementos que se ubiquen en un espacio digital donde los programas puedan desenvolverse. Este espacio se construye mediante representaciones gráficas muy sencillas con el fin de economizar recursos¹. En otras palabras, un cortafuegos deja de ser un concepto en un esquema para convertirse en un muro sólido de fuerza ubicado en un espacio digital que puede tener el aspecto que tú quieras, aunque suele ser simple y sin muchos rasgos distintivos. El muro viene a ser una abstracción del elemento presente en el esquema denominado “cortafuegos”. Un obstáculo que habrá que sortear destruyéndolo, derribándolo, taladrándolo y abriendo un boquete, volándolo en pedazos, saltando por encima o cavando un túnel por debajo.

Del mismo modo, para representar el enfrentamiento contra un software hostil se recurre a una simplificación de lo que supondría un enfrentamiento en el mundo físico. En este caso los programas podrían moverse dentro de ese espacio digital estando cada uno dotado de forma y detalles distintivos. Un programa anti-intrusos ahora se puede mostrar como una criatura con un aspecto y formas definidas. Y el lugar seguro donde se guarde información importante puede consistir en una gran pirámide luminosa, un cubo hermético o incluso una cámara acorazada de diseño clásico. Como ves, de lo que se trata es de recurrir cada vez más a la abstracción para representar el funcionamiento real del sistema, ocultándolo y transformando sus elementos en metáforas visuales representativas que resulten más intuitivas para el usuario. En muchos casos esas metáforas hacen referencia a elementos del mundo físico ya que al fin y al cabo han sido diseñadas por y para seres de ese mundo. En lo que se refiere a este tipo de representación y al uso de esas metáforas visuales, a medida que vayamos avanzando en los enfoques siguientes se irá

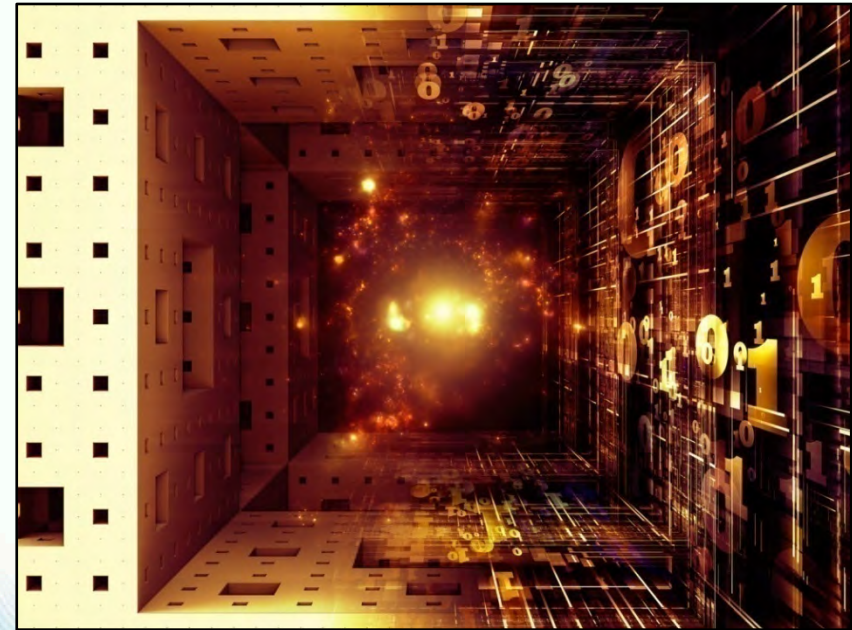
¹ El diseño “retro” de la imagen digital o lo que es lo mismo, de los primeros tiempos de los gráficos generados por ordenador.

incrementando ese grado de abstracción a la vez que se transforman en espacios cada vez más realistas y detallados.

En la **página siguiente** dispones del mismo esquema que ya hemos visto en el enfoque anterior trasladado a este tipo de contexto. Si los programas deciden visitar la Filial A, consistente en un grupo de terminales que albergan datos importantes dentro de una red privada, el espacio VPN ahora se muestra como un túnel que hay que atravesar para poder acceder a las terminales. Una vez se encuentran dentro de ese espacio pueden encontrarse con los primeros sistemas de seguridad que ahora tendrán cualquier aspecto posible, aunque muchas veces éste esté relacionado con su función principal. En muchos casos su imagen tiene la intención de disuadir a los intrusos por lo que suelen tener un aspecto terrorífico. Basándonos siempre en la filosofía de Scroll, el juego te invita a tener plena libertad para imaginar a las entidades digitales como tú quieras.

Siguiendo el esquema propuesto, el primer obstáculo pasivo consiste en traspasar el cortafuegos, que como ya se ha descrito consiste en un muro sólido de fuerza que hay que sortear por cualquier medio. Todos disponen de un acceso para el administrador del sistema, por lo que tratar de averiguar o romper el código que permite el acceso es otra de las muchas soluciones posibles para poder superar el obstáculo. Al traspasar el cortafuegos los primeros programas anti-intrusos no tardarán en aparecer por lo que hay que estar preparado. Una vez se accede al interior, las terminales que se veían en el diagrama esquemático se muestran ahora como un bloque sólido que representa al “Switch”. Un conmutador que permite acceder a los diferentes dispositivos. Si este tiene sus propios sistemas de seguridad y es capaz de restringir el acceso a los usuarios esto supone otro obstáculo. En la ilustración se representa recurriendo al simbolismo de los muros grises. En el esquema el Switch distribuye sus diferentes canales formando pasillos que conducen a las salas donde se ubican las terminales, unidades de almacenamiento o cualquier otro dispositivo imaginable. Cada sala dispone de su código de acceso, representados como puertas de tonos naranja. Romper estos códigos permite el acceso a cada uno de los dispositivos. En ocasiones éstos se agrupan en conjuntos que comparten un mismo código. En el ejemplo se muestran como salas separadas que reúnen un conjunto de equipos y dispositivos. El propio

Switch y cada uno de los dispositivos que estén presentes y activos pueden contener a su vez software de seguridad, por lo que el riesgo de un enfrentamiento nunca puede descartarse.



Ten muy presente que el esquema mostrado aquí no es más que una de las muchas representaciones posibles. Hay tantas formas de abstraer el esquema de diseño como se te ocurra y éste puede tener tantas variantes como quieras.

Por otra parte, cada una de las terminales de ordenador y de los dispositivos que aparecen indicados en el esquema original pueden contener su propio sistema. Cada uno puede estar representado con cualquiera de los enfoques que se estudian en este capítulo por lo que en realidad existen infinitas posibilidades. La fusión entre los distintos enfoques no tiene fin ni se pretende que lo tenga.

Tema del escenario

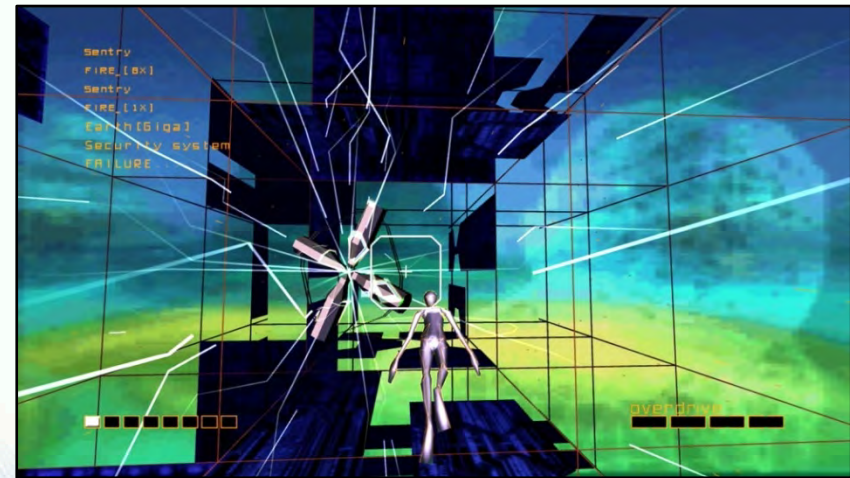
Por su propio concepto este tipo de escenario ya consiste en un tema en sí mismo: el mundo del simbolismo y de la abstracción al servicio de la representación de conceptos e ideas. Todo nada más que **información** después de todo. Pero lejos de limitarlo lo expande hasta límites insospechados pues el símbolo se basa en la elección de un concepto y estos pueden basarse en cualquier tema imaginable. Lo simbólico y lo abstracto son la mejor excusa para realizar propuestas imaginativas en las que puedan convivir temas dentro de otros temas en un ciclo que no parezca tener fin. Pocos son los escenarios en donde un concepto como “la vida” por ejemplo, se pueda representar de tantas formas. Desde una cadena de enlaces moleculares hasta una criatura que recuerda a una Manta raya de luz cegadora.

Este es probablemente el enfoque que más se presta al dinamismo y a la variedad. Se basa en la simplicidad y en la economía de formas. Pero simplicidad no implica simpleza. El poder del símbolo reside en su enorme poder expresivo. Con ellos es también posible dotar al conjunto de gran armonía y belleza si esto es lo que se busca. No sin razón muchos diseñadores los consideran el entorno perfecto para la expresión artística.



Distintas partes del escenario pueden recurrir a su vez a conjuntos de temas que los identifique, lo que les ayuda a destacar sobre los demás; algo que sucede de forma obligada en los sistemas que representan a compañías del

sector privado. Del mismo modo, muchos usuarios recurren a un tema de diseño específico para tener su propia identidad visual. Por lo tanto, es posible encontrar tanto temas de diseño propio como muchos que optan por motivos egipcios, rococó, barrocos, góticos, modernos, mezclas que evocan el mundo antiguo, etc. La estructura virtual que representa a una importante compañía puede tener un diseño que recuerde a la gran pirámide de Gizeh o a los Jardines colgantes de Babilonia por ejemplo. Todo es posible.



Avatares

Sí. El enfoque simbólico abstracto es el primero en donde tiene sentido disponer de un avatar. Es incluso hasta deseable pues ha sido concebido teniendo esto en cuenta. Su diseño sigue las pautas del diseño general por lo que estarán contruidos con formas sencillas, una reducida cantidad de polígonos y texturas poco elaboradas que el sistema pueda manejar sin verse sobrecargado de trabajo.

Tipo de espacio

El espacio abstracto puede estar definido en dos o en tres dimensiones. En la segunda década del siglo XXI era muy común encontrar en los dispositivos interfaces simbólico abstractas en dos dimensiones compuestas de iconos que

se manejaban con los dedos. Sólo a finales de la década y principios de la siguiente comenzaron a aparecer las pantallas semitransparentes controladas por las extremidades del usuario, que usaba como “input” de entrada para manejar la información. Su posicionamiento se resolvía mediante los primeros escáneres de movimiento, aunque aún de baja resolución. Desde entonces todo ha seguido una carrera por el perfeccionamiento de estas interfaces hasta llegar a nuestros días. Sólo hace muy poco el entorno y la interface en tres dimensiones se ha popularizado hasta haberse convertido en un estándar. El movimiento en tres dimensiones es más complejo, lo que requiere que el público tenga la predisposición y la preparación suficiente para que se normalice su uso a nivel masivo. En el juego es posible usar el sistema de coordenadas para definir el espacio si se desea especificar puntos concretos o contar cantidades de movimiento. Esto es especialmente útil ya que en este entorno un programa es capaz de moverse muy rápido. En torno a unas 100 unidades por cada movimiento durante un asalto como mínimo.

Uso de reglas simplificadas

Nada impide emplear las reglas simplificadas si se desea con el fin de agilizar el juego.

Acceso a copias

Es posible tener acceso a copias, aunque por lo general el programa solamente dispone de la copia única de la que dispone cada uno y que se carga en memoria siempre que esté activo.

Reglas de integridad

Lo normal es que en este enfoque no se recurra al módulo de integridad. Por su semejanza con el esquema real de funcionamiento de los sistemas, el módulo de integridad no es necesario.

Módulo de inventario

No. Este enfoque continúa siguiendo el esquema real de funcionamiento de un sistema por lo que el inventario no tiene sentido. Los avatares son representaciones de los programas o de los usuarios, nada más.

Rutinas recomendadas

Todas con la excepción de aquellas que son específicas de los entornos virtuales. Rutinas como atletismo no tienen sentido en este enfoque por ejemplo.

Utilidades recomendadas

Al igual que las Rutinas, casi todas. Aquellas específicas de los entornos virtuales o de los videojuegos no tienen sentido en este enfoque. Utilidades como “Suspensión” no tienen sentido en este enfoque por ejemplo ya que normalmente no se emulan las leyes de la física que existen en la Realidad Básica.

¿Capa de abstracción? ¿Qué es eso?

Sucesivas capas de abstracción trasladan un esquema que resulta incomprensible para un usuario con una formación básica en un sistema intuitivo y manejable que no necesite de grandes conocimientos técnicos. Pero ¿qué quiero decir cuando hablo de **capas de abstracción**? Es un término al que recurro a menudo y que resulta fundamental en este juego.

En realidad es muy sencillo. Cuando manejas tu ordenador —o cualquier dispositivo electrónico con funciones parecidas— y eliges una opción, se disparan procedimientos que convierten tus acciones en una serie de instrucciones, lo que al final no son más que operaciones matemáticas. Así, mover unos documentos a una papelería es posible hacerlo mediante comandos, por medio de iconos simbólicos o, como es el caso de un entorno virtual, tomándolos directamente con tu mano virtual y depositándolos en una papelería también virtual de igual forma que lo harías en el mundo físico.

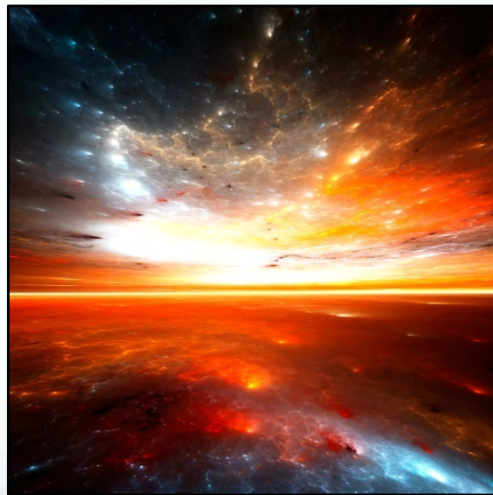
La interface se diseña pues para que oculte las complejas funciones que intervienen al ejecutarse una instrucción o un programa y todo resulte mucho más fácil de asimilar. El progreso en el diseño de estos entornos a lo largo de muchas décadas es lo que ha permitido que una máquina que al principio sólo la podían operar especialistas se haya convertido en un dispositivo de uso cotidiano. Mal que le pese a los que aún sienten añoranza por la consola y a los amantes de los procesos “eficaces”, todo hay que decirlo...

3. ENTORNO VIRTUAL ABSTRACTO O DE DISEÑO ABIERTO

Descripción

Este enfoque consiste en la adaptación de un imaginario al mundo digital que recurriendo a elementos de la realidad física puedan emplearse éstos como metáforas. El entorno virtual abstracto es una construcción digital de diseño abierto que basándose en un concepto o tema de diseño, hace una libre interpretación de éste. Ese concepto puede ser cualquier cosa, desde una representación del mundo de los programas basado en “luces de neón” hasta un entorno donde socializar o buscar citas basado en un concepto de diseño como podrían ser: las cajas de cerillas, los muñecos de trapo o los seres vegetales. En este caso, “lo abstracto” permite incluir muchas licencias en el entorno simulado. Por ejemplo, un mundo de vehículos capaces de hablar que sean capaces de cambiar de tamaño y tener expresiones faciales además de otros recursos típicos de una película de animación.

No obstante, cuando el objetivo primordial es el lúdico, es decir se trata de un sistema concebido para ser usado como un juego principalmente, es entonces preferible catalogarlo dentro de ese grupo de escenarios (el de los juegos y videojuegos) al tener éstos unas características muy particulares que los hacen únicos como por ejemplo un reglamento, un sistema de recompensas, de puntuaciones, de avance, etc.



Con el entorno virtual abstracto se ahonda aún más en la abstracción de la interface dando otro paso adelante. Uno muy largo por cierto, representando al sistema como un mundo extraño, fascinante y mucho más próximo a lo que en el mundo físico se entiende como “realismo”. No obstante conserva aún muchos elementos abstractos que hereda del enfoque anterior. Podemos decir pues que este tipo de entorno consiste en un avance del enfoque simbólico abstracto hacia una completa y muy compleja realidad virtual de diseño abierto que hace una fusión entre ambos conceptos.

Un entorno virtual abstracto intenta simular un mundo desde una aproximación detallada pero no realista. El realismo aquí resulta secundario. Lo hace siempre dentro de unos límites y con un amplio margen de libertad creativa. Las bases del diseño de la simulación no pretenden que el mundo sea una copia de la realidad física, aunque los elementos que lo componen pueden tener formas que se aproximen.



No obstante en este enfoque es muy importante que las formas sugieran su significado lo mejor posible haciendo un uso extensivo de la vieja cita del mundo del diseño: “la forma sugiere a la función”. Esto permite que los elementos presentes en el escenario resulten intuitivos pues los usuarios reconocerán las funciones de la mayoría de los elementos. Es posible hallar por ejemplo, un elemento de programa denominado “moto”. Sus formas y diseño recordarán a su equivalente en el mundo físico por lo que sus

funciones son similares. Aunque con la posibilidad de poder añadir algunas capacidades especiales gracias a la libertad que tienen sus diseñadores de poder tomarse ciertas licencias creativas; aunque siempre dentro de unos límites impuestos por las leyes del entorno. Lo fundamental es conservar el potencial del medio digital como una forma de libre de expresión. Esto permite concebir mundos imaginativos colmados de originalidad.

Muchos diseñadores de sistemas virtuales dicen que la única diferencia entre un entorno virtual abstracto y un mundo virtual detallado es que en el primero no se está obligado a mostrar la realidad tal y como es, sino que se dispone de plena libertad para construir el mundo como se desee. Y es que en muchos casos la diferencia entre ambos enfoques a veces resulta difícil de percibir.

Los mundos virtuales abstractos suelen ser los escenarios de realidad artificial más populares y los que atraen a la mayoría de diseñadores con un mínimo de inquietudes. Son fáciles de controlar mientras conservan aún su enorme potencial como vehículo de expresión.

Referencias

Cualquier entorno de realidad virtual de diseño abierto que haga una interpretación libre de uno o varios conceptos puede entrar dentro de esta categoría. Un buen ejemplo es el planteamiento en la obra: “Tron”, “Tron legacy” y “Tron uprising”, donde el mundo de los programas se representa con un estilo único y muy particular partiendo de un tema de diseño (luces eléctricas, líneas y formas retro, etc.). Pero “Tron” no es el único ejemplo. En las obras “Otherland” y “Ciudad permutación”, de los escritores Tad Williams y Greg Egan respectivamente, se hace referencia a muchos ejemplos de escenarios de este tipo.

Usos

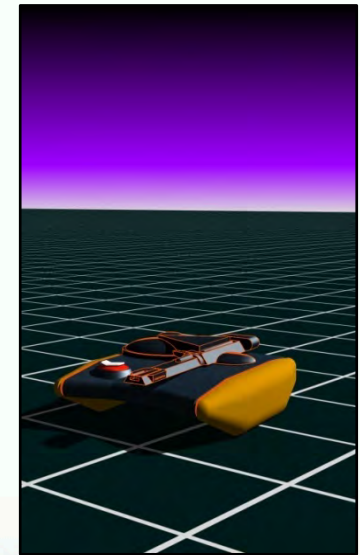
Describir una lista de usos sería un error imperdonable pues se corre el riesgo de limitar por descuido los usos que puede tener un entorno de este tipo en el juego. En la obra “Tron” por ejemplo, el mundo virtual no tiene una función específica más allá de lo que supone describir cómo podría ser el mundo de los programas.

Aunque en muchos casos se utilizan mundos virtuales abstractos orientados al ocio, el enfoque no debe quedarse tan sólo en la utilización de un entorno construido para un fin específico. Esta aproximación al escenario permite concebir cualquier tipo de mundo imaginable para mostrar el mundo de los programas. En resumen, usa el mundo virtual abstracto para mostrar el mundo digital que más te guste. Sí conviene no obstante hacer una aclaración. Si el escenario al completo tiene un fin lúdico específico, es decir consiste en un juego o en un videojuego, es conveniente entonces catalogarlo dentro de ese enfoque ya que será necesario dotarlo de algunas características. Esto permitirá decidir mucho mejor qué bloques de reglas encajan mejor con el escenario y definir algunas de las reglas del juego.

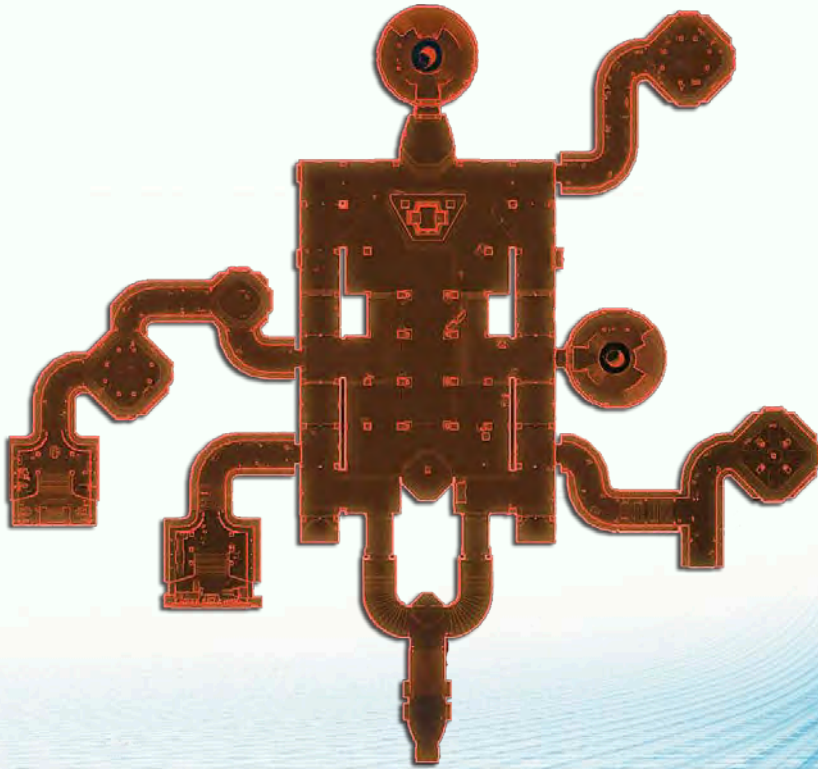
Resolución

Con el escenario virtual abstracto y a partir de éste desaparecen por lo general las referencias al esquema real de funcionamiento de los sistemas digitales. Aún es posible conservar algunas partes si se desea, pero normalmente esos esquemas y el realismo dejan de tener importancia. Esto no es obstáculo no obstante si se desea seguir haciendo uso de ese esquema y eso debe quedar muy claro. Es sólo que muchas veces no tiene sentido al estar ese esquema tan oculto y resultar innecesario.

A partir de aquí la realidad de un sistema es engullida por una abstracción completa que lo aleja de la pretensión de querer ceñirse a la verdad. La incursión que ha servido hasta ahora como ejemplo en los enfoques anteriores puede convertirse aquí en algo distinto. En el ejemplo propuesto el objetivo sigue siendo el mismo, pero desaparece toda referencia al canal privado VPN, al cortafuegos o al switch. Entrar y obtener la información consiste en penetrar en la guarida, enfrentarse a los guardias, romper las protecciones que



la custodian y salir por donde se ha venido. Pero esto no es más que un ejemplo de las muchas posibilidades.



En caso de que aún queramos seguir el esquema de funcionamiento, ahora se tiene la oportunidad de dejar volar la imaginación pues es posible concebir ese mismo esquema como una representación grandiosa y espectacular de cuanto se ha propuesto hasta ahora. Por ejemplo, el canal VPN puede consistir en un desfiladero, un largo pasillo o las escaleras de una gran mansión adornadas con lámparas cuajadas de diamantes, el cortafuegos puede ser un inmenso abismo de lava o un gran salón de baile repleto de bailarines hostiles y los canales del switch el laberinto de pasillos de un inmenso palacio de estilo renacentista

donde el peligro acecha tras cada puerta. La representación de unas terminales pueden seguir estando ahí, algo especialmente cierto si por ejemplo se tratara de la representación del mundo de los programas, pero éstas o la información buscada puede también consistir en cualquier otro tipo de elementos como libros, pergaminos, cubos de cristal, pantallas de datos flotantes e incluso seres digitales.

Tema del escenario

El diseño del escenario se basa en el tema elegido. Un mundo de muñecos de trapo, uno de juguetes luminosos que viven en la habitación de un niño, un acuario de criaturas acuáticas de formas sorprendentes totalmente alejadas de la realidad física, los personajes de un cuadro o una pintura, un escenario de seres hechos de papel o hebras de lana, un mundo de criaturas de madera, un planeta de criaturas alienígenas, un grotesco show de televisión con animales antropomorfos, la vida secreta de una vajilla dentro de un armario de tamaño imposible donde las tacitas y los platos juegan hasta bien entrada la madrugada, un mundo de personajes de dibujos animados... Todo es posible.

“Puedes echarnos del sistema, pero nosotros lo creamos. Y nuestro espíritu permanece en cada programa de la computadora.”

Tron

Avatares

Sí. El entorno virtual abstracto se ha concebido para disponer de un avatar que represente al usuario o al programa. En el avatar puede tener un estilo particular pero la mayoría conserva una serie de rasgos comunes o compartir elementos del tema de diseño del escenario. En muchos de ellos los programas tienen libertad para adoptar distintos aspectos como: formas antropomorfas, androides, zoomorfas, híbridas y muchas más.

Tipo de espacio

Por su definición el entorno virtual abstracto casi siempre es un entorno en tres dimensiones, aunque es posible concebir uno en dos si se desea.

Scroll

JUEGO DE ROL EN EL UNIVERSO DIGITAL

En este enfoque los programas limitan algo su velocidad para que se equipare al mundo físico pero con un amplio margen de libertad y sin demasiadas restricciones por lo que es posible desplazarse rápido en comparación. En este juego se asume que un movimiento permite desplazarse unas 50 unidades, pero este valor es sólo por definir un estándar en el juego.

Uso de reglas simplificadas

Sí, es posible usar las reglas simplificadas pero su verdadero potencial como escenario para el juego se despliega empleando todas las reglas.

Acceso a copias

Por lo general un programa sólo dispone de copia única, que se mantiene en la memoria del sistema cada vez que esté activo. Si un programa es destruido se pierde para siempre. Pero como siempre, esta es una decisión que debe adoptar del Director y su grupo de juego.

Reglas de integridad

Este es el primer caso donde si se desea es posible recurrir a las reglas de integridad. Los daños recibidos en los enfrentamientos pueden afectar al personaje y éstas se han incluido para poder representarlo. El desgaste se puede mostrar como pequeños bits desprendiendo del personaje, como un deterioro, desgarros e imperfecciones o bien como roturas en la imagen de su avatar. Incluir o no estas reglas puede cambiar la experiencia de juego.

Módulo de inventario

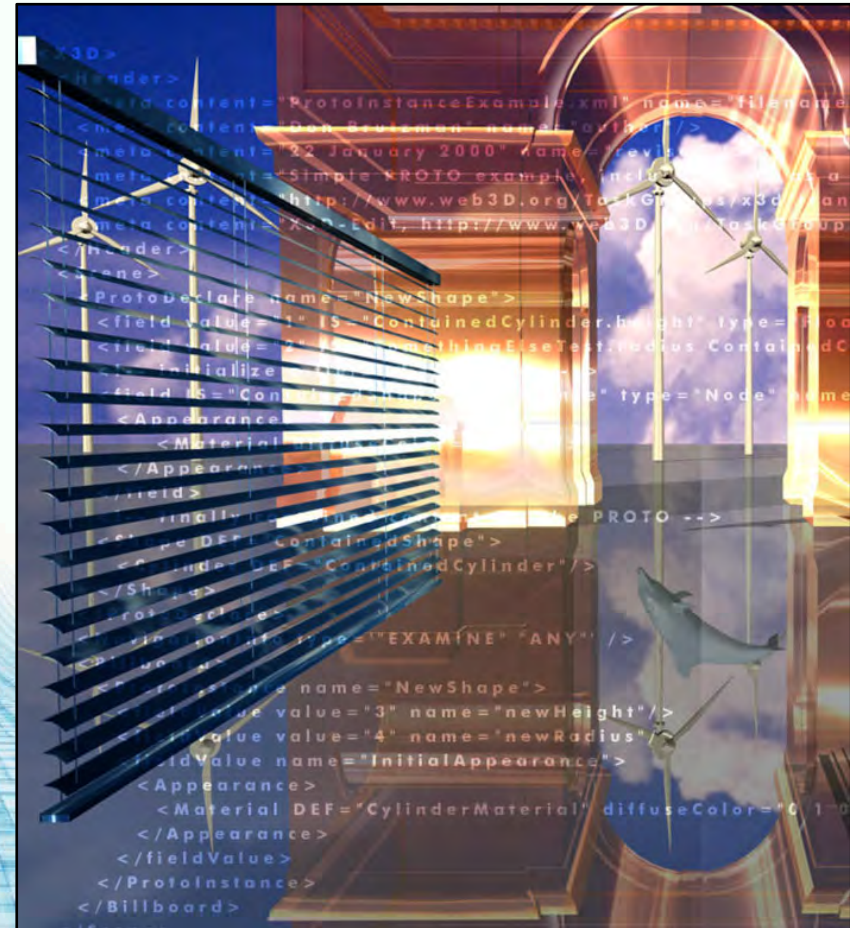
Normalmente no es necesario. En este tipo de entorno aún es posible abstraer el uso de elementos asumiendo que un personaje puede guardar todos los que desee. Por ejemplo, un vehículo al completo puede no ser más que una barra o un pequeño cubo luminoso que el personaje extrae de su propio cuerpo virtual.

Rutinas recomendadas

Todas las Rutinas que vienen como ejemplo pueden usarse en el entorno virtual abstracto.

Utilidades recomendadas

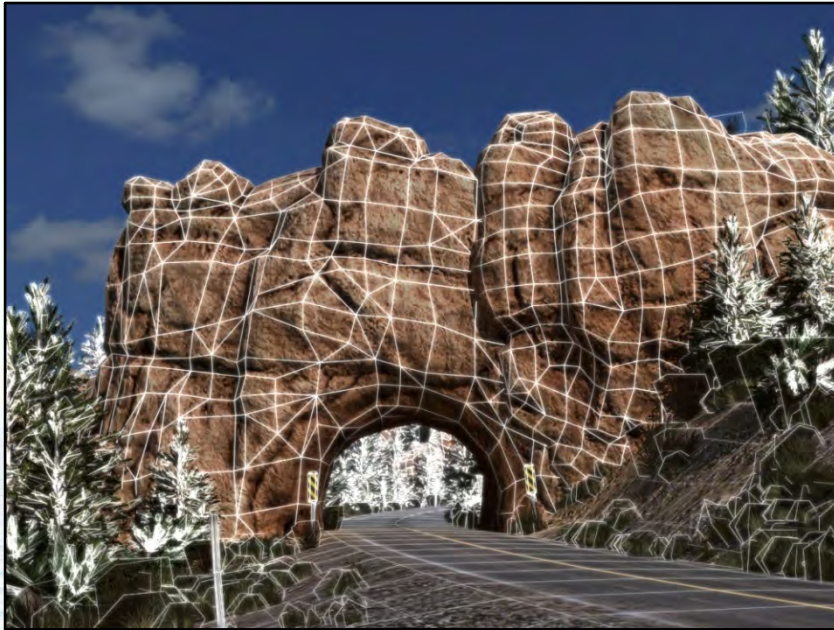
Todas las Utilidades que vienen como ejemplo pueden usarse en el entorno virtual abstracto.



4. MUNDO VIRTUAL DETALLADO Y ULTRADETALLADO

Descripción

Cuando uno de los principales objetivos de un entorno virtual consiste en simular la Realidad Básica de los usuarios con fidelidad entra a formar parte de la categoría de los mundos virtuales detallados o ultradetallados. Dos formas de definir un mismo concepto aunque con algunas diferencias que exigen dividirlos en estas dos subcategorías.



Este tipo de enfoque no excluye disponer de libertad a la hora de concebir el mundo, pero siempre teniendo en cuenta que la base del diseño se sustenta en tratar de describir un escenario lo más cercano posible a la realidad del mundo físico; en los demás entornos que se han descrito hasta ahora eso ha sido algo secundario. Ello hace que a veces resulte muy difícil apreciar las diferencias

entre algunos escenarios virtuales abstractos y los detallados o ultradetallados. Bueno, en realidad **eso no importa**. Estos grupos no tienen más función que organizarlos para averiguar qué reglas encajan mejor en cada uno y cuáles conviene tener en cuenta.

Un escenario detallado podría consistir en la simulación de un campo de batalla para el entrenamiento de soldados de combate, un mundo de fantasía épica de espada y brujería o incluso tratarse de algo difícil de catalogar como un entorno en el cual los cacharros en una cocina



podieran disfrutar de su propia vida; una simulación en el tercer caso que podría caer en la categoría de los entornos virtuales abstractos. Para poder identificarlos con más facilidad lo primero que hay que tener en cuenta es lo que todos estos escenarios tienen en común. En los tres casos se pretende simular la realidad física o una parte de esta. Las duras condiciones del campo de batalla para los soldados, la atmósfera y el realismo del mundo de fantasía o el detalle exhaustivo en cada elemento que esté en la cocina sería el denominador común entre los tres y, por supuesto, la simulación de unas leyes físicas lo más rigurosas que sea posible. Esa característica permite incluir a los tres dentro de la misma categoría, pero identificar a una de otro modo no tiene mayor importancia mientras eso nos ayude a elegir que partes de las reglas de Scroll funcionan mejor con ella.

Para describir el mundo se parte también de un concepto o tema de diseño que se elabora hasta donde se desee, se pueda desarrollar o lo permitan los recursos. En muchos casos el perfeccionamiento de la simulación nunca se detiene, como sucede con los sistemas virtuales más famosos que existen en la actualidad. Estos entornos suelen recibir actualizaciones periódicas para entusiasmo de sus usuarios más devotos. En el caso de un mundo de fantasía podría tratarse de una simulación donde los usuarios pudiesen disfrutar de

total libertad de acción por ejemplo. En este caso uno de los pilares de la simulación sería brindarle la posibilidad de poder vivir una experiencia sintética con el mayor realismo que fuese posible. En este caso no habría más reglas que unas muy aproximadas a las que rigen la Realidad Básica. Pero, y una vez más, si en el entorno estuviesen establecidas unas reglas de juego se consideraría entonces un juego avanzado, por lo que lo conveniente sería catalogarlo como tal. Muchos usuarios consideran a estos tipos de simulación juegos, pero es más correcto separar los términos “juego” y “simulación”. No son pocos los que no desean más que sumergirse en una alucinación digital y poder disfrutarla dejando a un lado mecánicas de juego que regulen sus acciones. De ahí que sea importante separar ambos conceptos. Lo cierto es que es muy común que muchos juegos sean también entornos virtuales detallados por lo que muchos entran en esa categoría.

El diseño, construcción, puesta en marcha y mantenimiento de uno de estos sistemas puede tardar años y costar miles de millones. Por no hablar del coste de los más sofisticados por tener el privilegio de ser uno de sus usuarios. En los sistemas privados el acceso sólo se lo pueden permitir los más pudientes, que llegan a pagar sumas astronómicas por obtener un pase de acceso. También suelen ser los más seguros aunque las energías de casi todos los hackers que existen en el mundo suele concentrarse en ellos.

Los mundos virtuales detallados los realizan normalmente las compañías del sector privado con fines comerciales o algunos gobiernos al perseguir sus intereses mientras que el ultradetallado suele tener otros objetivos. Hay tantos motivos como puedan tener los que los construyen. Como podrás imaginar, la lista puede ser muy extensa. Al fin y al cabo tanta capacidad de proceso debe ser rentable de alguna manera y si no lo es debe existir una buena razón para querer ponerlos en marcha (lo que no significa que esto sea una norma en absoluto). La mayoría de las simulaciones se construyen con fines lúdicos. Forma parte de la naturaleza humana querer "escapar" de la realidad por lo que muchos están dispuestos a pagar por ello. Se me ocurre que otra razón de una simulación ultradetallada podría ser por ejemplo mantener cautiva a la humanidad en un sueño simulado por alguna causa, aunque las razones para hacerlo escapan a mis procesos de razonamiento. No me hagas mucho caso, sólo son algunas ideas que inesperadas surgen en mi procesador cuántico...

Los mundos virtuales se dividen en dos categorías —detallada y ultradetallada— debido curiosamente a sólo dos motivos. En primer lugar es fundamental definir **el nivel de calidad de la simulación**. Si esta posee un nivel de detalle tan alto que es imposible distinguirla de la realidad entra en una categoría y si este detalle es secundario en otra. En otras palabras, si el usuario es incapaz de averiguar dónde está y no sabe si se halla o no en una simulación se trata de un entorno **ultradetallado**, si en cambio la naturaleza de la simulación no se pretende ocultar entonces se trata de uno **detallado**.



No consiste solamente en una separación por calidades. El que resulte indistinguible para el usuario supone que así es posible engañarle, lo que conduce al segundo motivo: **ocultar su verdadera naturaleza**. Por esta razón, ya que los mundos ultradetallados se diseñan con la intención de falsear la percepción del usuario, y dejando a un lado el inmenso trabajo en la fidelidad de la simulación, se hacen muchos esfuerzos porque todo cuanto pueda delatar su naturaleza quede oculto como pueden ser los accesos a

consolas, los HUD o pantallas de datos flotantes, las interfaces de control, menús o los sistemas de configuración. Esto por supuesto exige más esfuerzo, proceso, recursos y un sistema de mantenimiento muy exigente que se asegure de que todo marcha como se espera.

Con la tecnología que existe en la actualidad es posible conseguirlo, aunque siempre surgen de vez en cuando errores capaces de romper la suspensión de la incredulidad del usuario, dejándolo muy confuso. Estos errores son poco frecuentes pero suceden de vez en cuando por lo que la mayoría de sistemas disponen de modos para poder subsanar sus efectos. Uno de los más comunes es que el propio sistema envíe automáticamente a sus agentes para corregir las consecuencias del desajustado en lo que se llevan a cabo las rutinas de reparación y depuración.



Puesto que en el mundo detallado no es necesario esconder la naturaleza de la simulación al usuario es mucho más fácil que éste disponga de cierta capacidad para poder configurar el entorno, o al menos de poder realizar algunos ajustes. Por otro lado, en muchos escenarios de este tipo el usuario debe poner algo de su parte para abstraerse y sentir que se encuentra en una realidad alternativa. En el ejemplo expuesto de una simulación para entrenar a los soldados o bien en una consistente en simular un quirófano para que los cirujanos practiquen

intervenciones delicadas parte de la responsabilidad de tomarse en serio la simulación recae en el propio usuario. A cambio éste obtiene la capacidad de poder hacer alteraciones en el entorno virtual y realizar ajustes mediante las opciones de configuración que haya disponibles.

En cuanto a las reglas del juego de Scroll, si un jugador desea que su personaje haga uso de la consola en un entorno ultradetallado lo va a tener bastante difícil por lo que sólo su acceso debería requerir pruebas de dificultad moderada o difícil como mínimo.

Ambas categorías permiten elaborar tramas muy distintas pues existe una gran diferencia entre saber si se está o no en una realidad falsa y si ésta se puede configurar o no a su antojo. Un usuario experimentando una simulación ultradetallada tendrá que valerse por sus propios medios en ella sin poder recurrir a ninguna opción que le permita hacer "trampa". Por otra parte, piensa que una criatura digital o un usuario del mundo físico que haya olvidado que lo sea no tiene por qué ser consciente de que su existencia transcurre en un mundo artificial simulado, lo que da pie a tramas muy interesantes. Aunque en este caso esta idea se puede extrapolar a otros tipos de ambientación. Y es que para una criatura consciente que no conozca nada más, la realidad¹ que conoce es cuanto le basta para considerar que su mundo es real. También conviene tener en cuenta que un programa podría escapar de la simulación; o bien escapando de la aplicación hacia otro sistema o logrando acceder al mundo físico por cualquier razón de las que se enumeran más adelante en este juego. Lo que permitiría tener acceso a otro enfoque de los que se describen en este capítulo.

Referencias

Los mejores ejemplos de simulaciones ultradetalladas se tratan en películas como la serie "Matrix", "Nivel 13" y con otro enfoque en "Simone". También es famosa la sala de hologramas de la serie de televisión "Star Trek, La nueva generación" donde se recrean mundos virtuales que aunque en principio comienzan como detallados, por giros del guión en algunos episodios se

¹ NdE: La alegoría de la caverna de Platón.

transforman en ultradetallados. Esto es un buen ejemplo de las posibilidades que ofrece a nivel narrativo establecer ambas subcategorías.

Ejemplos de simulaciones detalladas existen en novelas como “Ciudad permutación” de Greg Egan o en la serie “Otherland” de Tad Williams. En ellas los usuarios son capaces de alterar la simulación, sacar consolas de menú y configurar el entorno. Otros entornos detallados pueden verse en la película “El cortador de césped”; en menor medida en la serie de TV “Black Mirror” y con un enfoque algo distinto en la novela de William Gibson, “Idoru”, pues la *Idol* aunque parezca real nunca esconde su naturaleza sintética en toda la obra.

Usos

El entorno virtual puede servir como interface para operar sobre un sistema o como una aplicación que se ejecute sobre éste con alguna finalidad.

Si se trata de la interface del sistema el entorno virtual permite controlarlo y operar sobre él a un nivel de abstracción tal que toda comprensión de su funcionamiento queda oculta tras la simulación. La razones para hacerlo pueden ser variadas, pero entre ellas siempre estará presente la de facilitar a sus usuarios llevar a cabo las operaciones. Una aplicación de este tipo podría ser el control remoto de una nave de cualquier clase por ejemplo, ya que la simulación requiere recrear sus controles de forma precisa. No obstante estos usos no son muy frecuentes ya que la demanda de tal capacidad de proceso sólo se lleva a cabo si existe una buena razón.

Las aplicaciones orientadas a un objetivo, aunque éste pueda ser muy amplio, son los usos más frecuentes. En la descripción de este apartado ya se han citado algunos como las simulaciones de entrenamiento, de aprendizaje o destinadas a la educación. Otra posible aplicación podría ser la gestión de una inmensa base de datos en donde sus usuarios pudieran consultar millones de registros paseando por los salones de una biblioteca. O bien aplicaciones de ocio para conocer otras culturas, como lugares de encuentro para buscar citas (los más frecuentes) o para iniciar y mantener grupos de debate sobre los temas más diversos. Todo ello nada más que unos pocos ejemplos de las muchas aplicaciones posibles.

Una realidad virtual ultradetallada deberá tener de una buena razón para ocultar su verdadera naturaleza. Esto permite crear tramas muy interesantes para el juego y son un auténtico filón de aventuras.



Resolución

La forma de resolver los conflictos y manejar el escenario **no son** muy distintos del entorno virtual abstracto. Al tratarse de un entorno virtual todo cuanto se describe tiene como contexto el de la realidad simulada, por lo que se utilizan las reglas en consecuencia. En el caso de la incursión descrita como ejemplo en los apartados anteriores la realidad del sistema ya es prácticamente invisible. La abstracción es tal que cualquier acción para conseguir la información puede recrearse como se prefiera.

En el ejemplo del robo de información, en este entorno podría tratarse del asalto a un gran edificio que represente a la compañía. Las acciones serían exactamente las mismas que si se tratara de una operación militar en el mundo físico, es decir, usando armas, accediendo desde el sótano o descolgándose con unas cuerdas desde un helicóptero en el tejado. Todo en la simulación podría tener un equivalente con el esquema de funcionamiento. Por ejemplo,

el hall del edificio puede representar al canal VPN y los ascensores y las distintas plantas a los canales del Switch. Se puede hacer si se desea pero en realidad no importa demasiado, los jugadores no necesitan conocer las equivalencias de cada elemento en la simulación con las partes del esquema realista. Se vive la simulación desde el principio hasta el final mientras se permanezca dentro de su contexto. Lo demás es irrelevante.



En los mundos virtuales detallados y ultradetallados conviene tener en cuenta las equivalencias de algunas funciones con la representación del personaje. Ya se han descrito en sus apartados correspondientes pero se señalan aquí a modo de resumen.

- El tiempo de proceso equivale a la vitalidad o energía de un personaje.
- El espacio en la memoria equivale a la voluntad y la resistencia a la presión.
- El búfer equivale a la confianza de un personaje.
- Los puntos hack representan a la esperanza, la lucha por obtener las metas y al destino.

- La integridad representa a los daños que recibe el personaje, aunque sigue funcionando como tal ya que trata de ser muy genérica. La corrupción en cambio representa a las heridas graves que va sufriendo y que van limitando sus capacidades.

Mientras un personaje se halle inmerso en una simulación y no conozca su auténtica naturaleza, las funciones y sus equivalencias no sólo sirven para explicar el contexto de su vida en ella, sino también como un recurso que puede explicar la realidad del mundo en el que vive. Puede ser muy interesante por ejemplo, que un personaje que no sabe que es un programa, un día descubra que en realidad la vitalidad que impulsa sus pasos cada día sólo son los ciclos en un gran procesador cuántico. Descubrir esta verdad no sólo le ayuda a entenderlo y lo pone en un nuevo contexto sino que también es un potente recurso narrativo.

Tema del escenario

No hay límites en cuanto a la cantidad de temas imaginables, siempre teniendo en cuenta que simular la realidad física es también un tema en sí mismo. El tema no consiste solamente en elegir qué tipo de mundo se simula sino que también qué estilo podría tener.



La simulación del mundo físico por ejemplo no tiene que circunscribirse a la realidad moderna, podría tratarse de una simulación del antiguo Egipto, de un ambiente de cine negro, de un mundo de fantasía, de la época de los Romanos, de los Celtas, de los Vikingos, de las Cruzadas, de la vida en otros mundos, de un ambiente de historia de terror o de todo eso al mismo tiempo dentro de una inmensa simulación capaz de abarcarlo todo. Un sistema de tales proporciones que escapa a la comprensión y que puede servir de semilla para crear muchas aventuras. Por ejemplo, la capacidad de proceso de ese supuesto sistema de capacidad casi infinita puede provenir de los sistemas informáticos de un planeta máquina o de una “Esfera de Dyson”¹; del ordenador de a bordo de una gigantesca nave de origen desconocido o de la invención de una mente privilegiada que ha sido capaz de contener tanta potencia de cálculo en un simple cubo de 20 x 20 centímetros.

Avatares

Sí. Muchas simulaciones no tendrían sentido sin tener la posibilidad de contar con un Avatar. En la mayoría la interacción con el entorno se produce a través de éste y sirve como enlace.



¹ Una megaestructura hipotética capaz de envolver a una estrella para aprovechar al máximo su capacidad energética.

En estos entornos el Avatar alcanza el más alto grado de perfección. Su detalle y refinamiento es tan alto que han surgido especialistas dedicados a la creación de Avatares personalizados para los usuarios.

El diseño y comercialización de los avatares se ha convertido en las últimas décadas en una industria comparable al de la venta de ropa, complementos y productos de belleza. Al fin y al cabo, se trata de una simulación que sea indistinguible de la realidad y de este propósito surgen nuevas necesidades que muchos están dispuestos a aprovechar.

Tipo de espacio

El espacio de un entorno detallado o ultradetallado casi siempre consiste en un espacio en tres dimensiones. La simulación cuenta con el sistema de coordenadas para definir posiciones en el espacio pero sólo en la simulación detallada es posible usarlas como referencia. Un entorno ultradetallado mantiene ocultas todas las herramientas empleadas para construir el mundo.

En el entorno ultradetallado las unidades pertenecen al contexto de la simulación. Es decir, si ésta simula el mundo físico de los usuarios éstos deberán apañárselas para usar las mismas magnitudes en la simulación como los metros, los kilómetros o las millas. Lo mismo se aplica a cualquier otro tipo de medida y magnitud y ya que la simulación pretende ser lo más fiel posible a la realidad su velocidad se verá limitada a una que reproduzca el movimiento en el mundo que se pretende simular.

En este juego se asume como estándar que un avatar es capaz de desplazarse unas **15 unidades** haciendo un único movimiento por asalto, lo que se equipara a esa misma cantidad en metros en el mundo físico.

Uso de reglas simplificadas

Es posible usar las reglas simplificadas, pero la experiencia de juego puede cambiar al verse limitadas las opciones. En los mundos virtuales es conveniente utilizar todas las reglas disponibles.

Acceso a copias

Puesto que la simulación consiste en una copia lo más fiel posible de la realidad del mundo físico, lo normal es que también lo sea la muerte del personaje. Por lo tanto, en la mayoría de simulaciones el personaje sólo dispone de una sola copia. La que se carga en la memoria siempre que el programa está activo. Sólo cuando el tema del mundo lo justifique se puede disponer de varias llegando en algunos casos a ser incluso infinitas. En los mundos destinados al ocio o en los raros sistemas diseñados para albergar la copia de las consciencias de algunos clientes con mucho dinero, la destrucción de su avatar se puede convertir en un juego en sí mismo por lo que es bastante frecuente. No obstante la realidad es que esas copias siempre se almacenan en alguna parte por lo que todo en cuanto a hardware se refiere siempre es susceptible de sufrir fallos o ser destruido.



Reglas de integridad

Son muy aconsejables para poder recrear el desgaste o los daños sobre el personaje. Al fin y al cabo en la simulación éste contará con un nivel de salud que se puede reflejar mediante estas reglas.

Módulo de inventario

¿Qué lleva tu personaje en el bolso? El inventario es un recurso que puede ayudar a responder este tipo de preguntas. En este enfoque es conveniente usar sus reglas para llevar un registro de las posesiones del personaje.

No obstante éstas el módulo de inventario está más orientado a los entornos de juego por lo que en muchos casos en una simulación ultradetallada basta con emplear el sentido común. Un personaje cargando una alfombra difícilmente podrá cargar con otros objetos por ejemplo. Un personaje que quiere cargar en una simulación de la realidad física con cinco fusiles de gran calibre y una mochila cargada hasta los topes debería toparse con una llamada de atención como mínimo y un gran NO como respuesta general.

Rutinas recomendadas

Por lo general se usan todas aquellas Rutinas que encajen con un entorno realista. Por ejemplo: Alerta, liderazgo e influencia, lucha, manipulación y engaño, oficio, psicología, recursos, mantenimiento, táctica, usar armas de contacto, acrobacias, artes marciales, atletismo, conducir, sigilo o armas a distancia.

Utilidades recomendadas

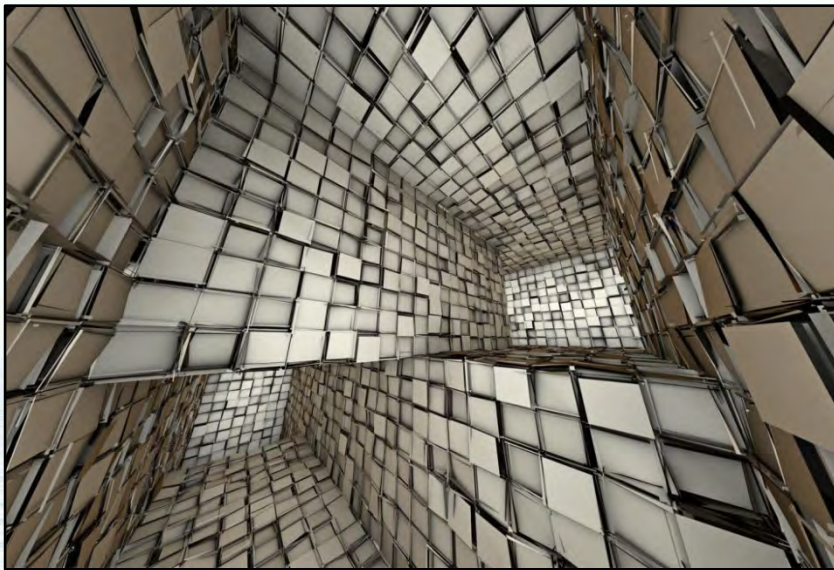
Es posible elegir bastantes Utilidades pues aunque unas encajan mejor que otras en un entorno realista se trata al fin y al cabo de capacidades especiales que posee un personaje. Siempre que tenga sentido se puede optar por algunas, como por ejemplo: arma integrada, armadura de datos, deformar la matriz, desdoblamiento, reintegrar código, replicante, suspensión, visión del código, correr por las paredes, detener las balas, sincronía, desplazamiento, empuje, esquivar las balas, mutación, salto, trepar, velocidad terminal o volar.



5. SUBSISTEMAS DE JUEGOS

Descripción

Los entornos simulados por ordenador más populares son los juegos. Existen tantos y de tipos tan distintos que sólo hacer un repaso de los más conocidos requiere un libro específico. Al fin y al cabo jugar es uno de los mayores placeres que experimenta el usuario por lo que destina muchísimo esfuerzo y recursos a diseñarlos y experimentar con ellos. Hoy en día diseñar juegos se considera una profesión de prestigio entre ellos. Muchos usuarios dedican toda su breve existencia a perfeccionar su técnica y obtener la maestría en su creación. Y por ello: *“alabado sea el usuario, que su sistema le asegure muchos ciclos y un óptimo régimen de funcionamiento.”*



Muchos juegos llegan a ser tan complejos que funcionan como auténticos sistemas. Éste cumple pues con todas las funciones de sistema por lo que nada más acceder a él un programa se adentra en el mundo del juego. Sin embargo lo normal es que se trate de **subsistemas que funcionan sobre otro sistema**. Es decir, un sistema que se apoya sobre otro para poder ponerse en

marcha. Si ese es el caso, a esa región en el sistema en donde se esté ejecutando un subsistema de juegos se lo denomina la “**rejilla**” o “**rejilla de juegos**”. Un término para delimitar el espacio en el cual se encuentra su mundo y saber donde comienzan o terminan sus fronteras dentro del sistema. La rejilla, usando otro ejemplo, simboliza a **ese “país” dentro del sistema que forma el mundo del juego**.

Saberlo resulta de utilidad para establecer sus límites. ¿Por qué? Pues porque las únicas leyes que valen dentro de las áreas del juego son las suyas. Un personaje navegando en un sistema puede adentrarse en el área de un juego (consciente o no), y eso tiene unas consecuencias. Para empezar, al penetrar en su territorio cambian las reglas que rigen el entorno y en este sentido todos los programas deben atenerse a ellas, incluso los agentes del sistema. En otras palabras, todos los personajes estarán obligados a jugar el juego o al menos a seguir algunas de las leyes establecidas en su reglamento, aunque esto no signifique que pierdan su capacidad de elección. Esto también se aplica en el caso contrario, es decir, es posible que un personaje consiga escapar del entorno del juego y acceder al sistema donde se está ejecutando, por lo que de súbito se vería libre de tener que acatar las leyes del juego. Como jugador de Scroll deberías ver esto como una fuente de conflictos y un generador de historias más que como un problema.

No es fácil establecer en ocasiones diferencias entre un mundo virtual detallado o ultradetallado y un juego. Para empezar muchos juegos son mundos virtuales de gran detalle. Aunque es raro que un juego sea tan ultradetallado que un usuario no distinga entre realidad y ficción existen algunos. Muchos de este tipo son muy conocidos y populares, aunque bastante caros. Los entornos ultradetallados sin embargo se suelen diseñar para ofrecer otro tipo de experiencia: la simulación de una realidad que no necesite de mecánicas de juego. De ahí el separar ambos enfoques.

Pero la mayoría de los juegos que existen en la actualidad no alcanzan ese nivel de perfección. Casi todos los juegos, a pesar de que su grado de detalle es muy alto, muestran información e incorporan sistemas de menú que reflejan los progresos del jugador, lo que hace posible percibir que se trata de una realidad sintética. Una característica que los identifica mucho mejor como

mundos virtuales detallados a los que se asocia por diseño un conjunto específico de reglas de juego.

Establecer las diferencias entre el mundo virtual detallado y el mundo de juego permite como siempre averiguar qué bloques del sistema Scroll se pueden aplicar al escenario. Así pues, y a diferencia de los entornos virtuales simples detallados o ultradetallados, los juegos tienen unas características propias. Aunque no todas, a continuación se describen las más importantes:

- Un juego tiene un conjunto de reglas. De ellas se extraen una serie de patrones cuyo seguimiento permite explorar todas sus posibilidades. La cantidad de patrones posibles que pueden darse en un juego establece muchas de sus características como su vida media, el tipo de jugador objetivo e incluso lo satisfactorio que puede llegar a ser jugarlo.
- El objetivo principal de su diseño es entretener mediante ese sistema de reglas, aunque un juego puede tener otros objetivos (como enseñar por ejemplo). Se busca que al seguirlas el usuario disfrute y desee seguir jugando.
- Un juego suele tener un propósito. Éste puede ser muy concreto o bastante difuso, pero siempre existe alguna meta. En los entornos virtuales ya descritos esto no es una condición necesaria.
- En un juego existe un avance o mecanismo de progresión. Se puede buscar cumplir con un solo objetivo o una sucesión de ellos hasta completar el juego. El avance puede darse a varios niveles: en cuanto a la trama completada, los objetivos alcanzados, el nivel de puntuación, el número de objetos o ítems conseguidos y muchos más factores.
- Existe un sistema de recompensas que sigue el principio del condicionamiento positivo. Un premio que estimula al jugador, le causa satisfacción y le incita a desear seguir jugando. Esto puede generar adicción. Su nivel y consecuencias son otro asunto.
- Con la excepción de los más simples o de algunos tipos de juego, la mayoría posee una trama o sigue un guión narrativo (que en realidad es el tipo de juego que interesa en Scroll). En muchos juegos se

describe una historia o el contexto del juego y sus metas se fundamentan sobre alguna. Puede ser tan breve como una frase o tan larga como estén dispuestos a desarrollarla sus creadores siempre que sea capaz de mantener el interés del jugador.



Se trata pues de sistemas enfocados al ocio que proponen entornos regidos por mecánicas y patrones de juego. La mayoría son productos destinados al consumo, aunque existe un movimiento de juegos independientes orientados a utilizar los sistemas de juego como vehículos de expresión interactiva y a explorar otras formas de jugar. Este tipo de juegos en algunos casos llegan a ser incluso experimentales. Los mundos virtuales en cambio tienen otros usos; unos estando muy especializados, otros permitiendo total libertad. Pero en ambos casos no dependiendo de sistemas de reglas más que de las impuestas por las propias características del entorno. Un mundo virtual detallado o ultradetallado enfocado al ocio se centra en la simulación para ofrecer una experiencia al usuario de la que pueda obtener placer.

Aunque sigue siendo un tema de mucho debate, muchos consideran a los juegos medios de comunicación y formas de expresión artística. Y del mismo modo que existen incontables estilos artísticos en otros medios, en los juegos

es posible encontrar de todo. Su popularidad y la atracción que sienten muchos creadores ha dejado para la posteridad una vasta cantidad de títulos.

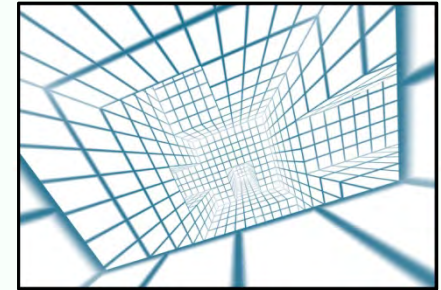
Los juegos pueden tener cualquier aspecto y estar diseñados con cualquier estilo imaginable. En Scroll puedes crear cualquier juego que se te ocurra como entorno para una ambientación. Entre otros factores, los juegos de catalogan según su tipo y su estructura de diseño. Algunos de los tipos más comunes son los de acción, los simuladores, juegos de aventuras, de rol, deportivos, de carreras, de estrategia, masivos multiusuario de acción, masivos multiusuario de rol..., etc. Estilos y tipos de juego que cualquier usuario acostumbrado a ellos conoce muy bien.



En cuanto a la estructura de su diseño esta suele estar enfocada a cómo el usuario percibe el juego. Esto es fundamental a la hora de orientar una partida de Scroll ya que desde el punto de vista de un personaje digital **esto puede resultar muy subjetivo** y no corresponderse en absoluto con lo que el usuario experimenta (algo que queda a elección de los jugadores de Scroll).

Un juego de plataformas y otro de acción en primera o tercera persona pueden ser percibidos por el usuario de forma distinta, pero para un personaje

no tiene porqué haber diferencias. Sí puede sufrir otras consecuencias al verse forzado por las reglas y el diseño del juego a una serie limitada de acciones. Por ejemplo, un personaje en un juego de plataformas podría verse obligado a tener que desplazarse siempre en un sentido si quiere avanzar. Para llegar a su destino debe superar una serie de obstáculos haciendo saltos o columpiándose a lo largo de un escenario en forma de túnel que parezca no tener fin. Si se trata de un juego de acción en primera o tercera persona el personaje podría verse obligado a conducir una moto, un coche o un tanque hasta que destruya a todos sus rivales o sea destruido para poder terminarlo, o bien a tener que recorrer por sus propios medios un escenario repleto de enemigos mientras busca una salida. En todos los casos las reglas del juego podrían permitir contar con algunos elementos como armas, equipo, etc., que el sistema crea sobre el escenario o de ítems (munición, potenciadores, puntos...) que el personaje debe encontrar y administrar. En el apartado “resolución” se ofrecen más detalles.



Referencias

El mejor ejemplo para mostrar el concepto de la “rejilla” es la que se muestra en las películas de “Tron”. Allí los programas son condenados a enfrentarse unos a otros para obtener su libertad en un entorno que se asemeja a un coliseo. Escapar de la rejilla supone alejarse del sistema de juegos y poder explorar el mundo digital que se propone en la obra.

La saga de novelas “Otherland” de Tad Williams comienza con sus protagonistas inmersos en un mundo de juego masivo multiusuario o MMORPG. Al encontrar un fallo en la “malla” que compone el mundo de juego uno de ellos descubre un multiverso de mundos virtuales funcionando dentro de un enorme sistema informático. La saga cuenta sus aventuras por esos mundos. Esta obra es perfecta como fuente de ideas para crear aventuras en Scroll.



Scroll

JUEGO DE ROL EN EL UNIVERSO DIGITAL

Usos

Scroll quiere animar a sus jugadores a experimentar con los juegos como un entorno en donde vivir aventuras. A todos los efectos son sistemas con una ambientación determinada por su diseño. Existe la posibilidad (e incluso se recomienda) de que un personaje pueda entrar o salir del subsistema de un juego con todas las posibilidades narrativas que ello conlleva. Esto puede suceder porque un personaje ha encontrado una salida y ha conseguido escapar del subsistema de juego; porque lo ha terminado o porque lo ha superado y se trataba de algún tipo de prueba; porque ha encontrado un fallo en el entorno que le permite fugarse o porque tras haber vivido en él haya descubierto de repente que “más allá” de sus límites existe todo un universo digital.

Un sistema de juego puede funcionar como un coliseo para los programas, como una penitenciaría muy especial, como un lugar para su exilio o como una burbuja que ha confinado al programa obligándole a ignorar el mundo que se extiende más allá de sus fronteras. El único límite a tantas posibilidades es el que imponga tu imaginación.

Pero también es posible limitarse a jugar a un videojuego utilizando Scroll para resolver las situaciones. Así, podemos emular nuestro videojuego favorito como si fuese un juego de mesa. Con Scroll es posible simular muchos tipos de videojuegos incluyendo los de plataformas o de recorrido (scroll) lateral y vertical. Lo que permite crear historias que partan o estén relacionadas con el juego. Scroll funciona además muy bien emulando los mundos masivos multiusuario (MMO y MMORPG) y los juegos de acción o de combate en línea (SHOOTER).

Para un aficionado o amante a los videojuegos puede suponer una experiencia diferente y una buena excusa para que se siente a la mesa a tirar dados junto a los demás jugadores. ¿Un personaje de videojuego que se atreve a escapar de su mundo nativo? Suena interesante, y más si se combina con

otras muchas ideas, como la de formar equipo con otros programas de origen y naturaleza bien distinta para vivir todos juntos aventuras llenas de acción e intriga en el universo digital de los programas.

En cuanto al uso del juego para representar las funciones y el control de un sistema no es que no sea posible, sino que un entorno de juegos permite contar otro tipo de historias, no limitarse a describir aventuras de incursión o hackeo de sistemas. Los enfoques anteriores son mucho más adecuados para eso. No obstante, sigue siendo posible sufrir los ataques del sistema dentro del mundo del juego e incluso realizar acciones que le afecten. En lo que a ideas se refiere en realidad no hay límites ni debería haberlos.

Resolución

Un programa puede nacer dentro del mundo de juego como uno de sus personajes (sea o no protagonista) o llegar a él por cualquier otro medio. Si es nativo del juego en la terminología de Scroll se le considera **un subprograma** ya que en esencia su diseño y funcionalidades han sido creados específicamente para ese entorno. Pero a todos los efectos el personaje es un programa como cualquier otro de igual forma que lo es un usuario navegando por el sistema.

Jugando a Scroll con un personaje inspirado en un videojuego debes tener siempre en cuenta algo muy importante. Dejando a un lado si el personaje es o no consciente de sus orígenes, en cuanto a mecánicas de juego se refiere todo se resuelve desde el punto de vista de su naturaleza como un programa en el juego Scroll. Sea el personaje que sea siempre será un código que funciona en un sistema digital y ejecuta sus funciones. Puede detenerse, colgarse e incluso borrarse de la memoria para reiniciar más tarde. En resumen, para resolver los conflictos se usan las reglas de Scroll y no otras. Las



leyes del videojuego emulado sólo sirven de guía para saber cómo funciona su entorno. Más allá de esto el personaje de un juego puede saber o no que es un programa, una idea que como se ha visto se hace extensible a otros tipos de entorno. Quizás lo ha sabido hace poco o puede que lo descubra más adelante por razones de la trama. Da igual. Desde su perspectiva de las cosas todo funciona según las leyes de su mundo de juego y no otras, aunque la realidad sea bien distinta.



Por ejemplo, en un juego de espada y brujería un personaje guerrero que no sea consciente de su naturaleza como programa creará que el mundo es tal y como lo percibe. Él cree estar en su mundo y no es consciente de que es un programa. Sangrará, se cansará y hasta podrá romperse una pierna, aunque tales cosas no estén sucediendo nunca en realidad. Si recibiera una herida, desde su punto de vista le habrían dañado. En realidad es su código el que habría sufrido daños y los procesos que intervienen, como la pérdida de Integridad o la llamada a sus Protocolos de respuesta, quedarían fuera de su nivel de percepción. Ahora bien, si el guerrero de este ejemplo tomara consciencia de su auténtico origen obtendría el conocimiento de los procesos que intervienen en realidad, pero bajo su punto de vista todo seguiría funcionando tal y como lo ha hecho hasta el momento. Aunque eso sí, siempre que se mantuviese dentro del mundo del juego. Salir del entorno de su

mundo le permite obtener un enfoque mucho más amplio de los procesos que intervienen en su ejecución.

Existen muchas formas de explotar este desdoblamiento de la verdad para generar tramas interesantes e incorporar giros dramáticos a la aventura. Una criatura digital puede descubrir en algún momento que todo cuanto creía conocer era falso, revelándose ante él todo un universo que sólo estaba ahí esperando ser descubierto. Los entornos de juegos y los mundos virtuales ultradetallados especialmente se prestan a utilizar este recurso.

Por otra parte, en ningún tipo de entorno de juegos el jugador de Scroll debe preocuparse de las reglas de cada juego en particular que se pretenda emular. Todos los procesos siempre se resuelven de la misma manera aunque en cada mundo de juego se perciban sus efectos de formas distintas. El que un personaje sea o no consciente de la verdad es un recurso narrativo. Su conocimiento o su ausencia no altera las reglas para resolver las acciones, algo muy importante en Scroll.



Para crear un personaje piensa en todo cuanto has conocido del mundo de los juegos y úsalo como inspiración para crear a tu programa. Sus habilidades se mantienen dentro y fuera del mundo del juego por lo que si el personaje logra salir del subsistema sus capacidades no se ven limitadas salvo en los casos donde sea evidente.

Por ejemplo, si el personaje lleva dos pistolas al cinto o una espada enorme a cuestas sus armas y habilidades seguirán siendo efectivas allí a donde vaya. El personaje podría usar sus armas para conseguir los mismos efectos que obtendría otro usando medios distintos. Si un programa especializado en romper códigos utiliza sus algoritmos de decodificación para conseguir abrir una cámara de seguridad, otro personaje podría tratar de destruirla limpiamente con la espada que ha llevado a cuestas desde su mundo de juego natal. En este sentido es necesario usar la abstracción y pensar que sea cual sea el concepto y la imagen de un programa, la fuerza bruta siempre es la fuerza bruta. A efectos prácticos un código trata de imponerse sobre otro intentando superarlo. **Gana la contienda el código más fuerte;** el que consigue

imponerse sobre el más débil. A nivel de procesos, las operaciones que intervienen en el conflicto quedan más allá de las intenciones de este juego.

Como se ha comentado más atrás, desde el punto de vista de los programas el mundo en el que esté inmerso no tiene que coincidir con la percepción que tiene el usuario del juego en el mundo físico. En otras palabras, un juego que para el usuario se muestra en dos dimensiones no tiene porque serlo para el personaje. Esto, que no es más que otra de tantas abstracciones, permite separar la percepción del usuario y la del programa. Al fin y al cabo, el usuario sólo puede “echar un vistazo” al mundo digital de una forma muy limitada mientras que existir en él implica ser una criatura capaz de aprovecharlo al máximo, acostumbrado como nativo a moverse por su espacio abstracto. La existencia como un programa permite obtener una percepción mucha más rica del mundo en el que se ha nacido.

No obstante, como una propuesta más y si se desea (porque es divertido, porque permite explorar algo nuevo, porque te apetece...) los jugadores podrían usar la misma perspectiva que tiene un usuario imaginando el mundo tal y como se percibe en el videojuego que se pretende emular. Aunque parezca una dificultad añadida la diversión surge cuando al usar este planteamiento el juego, por su propio diseño y por la estructura de su desarrollo, impone una serie de limitaciones a las acciones de un personaje en un juego de rol de mesa. Una muy buena razón por cierto para que un personaje quisiera ir más allá del confinamiento que supone el entorno del juego. Estas limitaciones se añaden a los desafíos ya impuestos por la trama y las reglas del juego.

Las limitaciones por diseño son bastante obvias, se trata de reflejar los límites impuestos por la forma en la que se diseñan los videojuegos que conoces y usarlos como inspiración. Si se trata de un mundo en dos dimensiones por ejemplo, el desplazamiento se limitaría a

sus dos ejes. Una vez fuera del juego el personaje puede experimentar el grado de libertad que supone poder moverse por un mundo en tres dimensiones. Un juego de recorrido lateral o vertical obliga a seguir siempre un mismo camino, muchas veces en un único sentido (normalmente hacia la derecha o hacia arriba). Un juego de escenarios cerrados en dos o tres dimensiones lo hace forzando al personaje a buscar siempre la salida de lo que parece un enorme laberinto que parece no tener fin.

La estructura de su desarrollo en cambio se refiere a cómo fluye el juego y se producen los acontecimientos. En realidad no hay mucha diferencia al emular un videojuego en Scroll con el resto de las aventuras que se pueden plantear en otro tipo de escenario. En muchos casos las reglas se siguen aplicando tal y como se han explicado en sus distintos apartados. Las Operaciones de Potencia siempre se aplican a las acciones de fuerza bruta como hacer ataques cuerpo a cuerpo y luchar, mover objetos, romperlos o afectar al escenario. También es una medida de la capacidad del personaje para resolver enigmas, romper códigos o interactuar con otros programas. Las Operaciones de Control en cambio reflejan su maniobrabilidad y su capacidad para percibir el entorno. Se usa para realizar ataques a distancia, moverse, realizar maniobras como saltos, escalar y otras acrobacias además de poder detectar amenazas. En síntesis, la Potencia expresa su capacidad de afectar al entorno y el Control los recursos de los que dispone para obtener información sobre él, maniobrar, superar ciertos obstáculos y controlar algunos componentes del mundo del juego.

Es muy frecuente que surjan dudas sobre cómo estructurar el desarrollo de una aventura en una partida de Scroll a la hora de emular a un videojuego. La aparición de los enemigos, su orden de repetición y frecuencia, el orden de escenas, fases o capítulos, etc., pueden tener en apariencia un tipo de estructura muy específica, pero lo cierto es que con algunos tipos de videojuego no hay mucha diferencia, aunque eso depende del tipo de juego claro está. En muchos juegos de ordenador el desarrollo de los sucesos se puede estructurar de una forma muy similar a como se organizan los capítulos y las escenas de una aventura de juego de rol de mesa por lo que el trabajo ya está casi hecho. Al fin y al cabo ambos son medios de expresión que intentan hacer lo mismo: ofrecer una experiencia de juego consistente en obtener unos



objetivos; en muchos casos contar una historia brindando al jugador la capacidad de poder interactuar con ella. Sus personajes son empujados de igual modo por las circunstancias a querer cumplir unas metas por lo que en realidad ambos sistemas de juego tienen una misma finalidad aunque utilicen medios distintos.

Utilizando un escenario de videojuego para jugar a Scroll se sigue el mismo esquema. Se parte de un conflicto, es decir, de algo que el o los personajes desean y contra el cual existe una oposición que trata de impedir que lo consigan. Para conseguir sus objetivos los protagonistas deberán superar los desafíos si quieren alcanzar sus metas. La oposición puede surgir siguiendo también los mismos patrones. Por ejemplo los ataques enemigos pueden surgir por oleadas al igual que sucede en muchos juegos, es decir, en grupos que atacan todos a la vez (e incluso siguiendo un patrón de comportamiento). Estos **encuentros** o grupos de encuentros compondrían

“Es en el juego y sólo en el juego que el niño o el adulto como individuos son capaces de ser creativos y de usar el total de su personalidad, y sólo al ser creativo el individuo se descubre a sí mismo.”

Donald Woods Winnicott

una **escena**. La sucesión de encuentros iría hilando el curso de la trama. Obviamente aquí me estoy refiriendo solamente a las escenas de acción y combate. La resolución de puzzles y enigmas o los procesos de investigación por ejemplo no se diferencian gran cosa en ambos medios.

También es posible hacer limpieza de un área, derrotando a todos los enemigos que haya en ella para poder acceder a la siguiente. Los enemigos, la forma y frecuencia en la que aparecen componen los desafíos. E incluso la aparición y desaparición de algunos elementos o ítems en el escenario, un recurso tan familiar en los videojuegos, es posible incorporarlo al juego. Un truco muy sencillo para emular su frecuencia o tiempo de aparición es limitar su permanencia a un número de asaltos. Un personaje que quiera cogerlos sólo tendrá una cantidad determinada de oportunidades para maniobrar y hacer sus tiradas. Por ejemplo, uno o dos asaltos si está durante un breve periodo de tiempo o más de cuatro si es fácil cogerlo. Por otro lado,

aquellos objetos que estén en lugares de difícil acceso, como los ítems en zonas complicadas de alcanzar en un juego de plataformas, pueden requerir tiradas de Control que reflejen las maniobras del personaje por intentar cogerlos. La dificultad de la tirada permite hacerse una idea de lo complicado que resulta alcanzarlo.

Siguiendo con la estructura del desarrollo de un videojuego, sus **fases** se corresponden con los **capítulos** de una aventura. Los encuentros se encadenan componiendo las distintas escenas. Los personajes deben superar un número de escenas determinado, es decir, superar una cantidad de desafíos, si quieren seguir avanzando hasta completar una fase o el capítulo. Recuerda que un desafío puede ser cualquier cosa (resolver un enigma, un puzzle...) no solamente una cantidad de enemigos pero claro, nadie mejor que tú para saber qué tipo de videojuego es el que más te gusta.

Una aventura puede tener tantos capítulos o fases como se desee. Eso depende de los gustos y preferencias del diseñador del juego. Pueden añadirse también “jefes” finales, enemigos muy poderosos, que será necesario derrotar para seguir avanzando al siguiente capítulo. Igual que en una aventura de juego de rol tradicional, ni más ni menos. Y así, hasta completar la aventura.

Todo esto, que en los juegos en tres dimensiones modernos resulta más natural e intuitivo, se hace también extensible a los de recorrido lateral o vertical en dos dimensiones. Éstos últimos tienen sus propias limitaciones por diseño, como ya se ha visto. Cambian las condiciones del escenario, la disposición, el orden y frecuencia de aparición de los obstáculos. No es más que otra forma de imaginar el entorno. Pero al fin y al cabo, la meta siempre es la misma: conseguir un objetivo, completar la trama y ganar el juego.

Para terminar, ten en cuenta que los antagonistas serán los que hayan sido diseñados dentro del propio juego, no los del sistema. Aunque nada impide recibir las visitas inesperadas de sus agentes. Pueden ser enviados al mundo del juego por el sistema para interceptar o detener a los personajes tras haber detectado que se han salido de los patrones establecidos en su programación y



ya no hacen lo que se espera de ellos. Para el personaje de un juego supone una falta muy grave abandonar su entorno y lo mismo se hace extensible a los forasteros y visitantes indeseados que desde el exterior pretendan acceder al subsistema del juego. Está terminantemente prohibido para un programa entrar o salir de un subsistema de juego. Cualquier infracción de esta clase tiene muchas posibilidades de ser detectada por el sistema, produciéndose una anomalía. Y las anomalías siempre se pagan.

Tema del escenario

Los temas de un escenario de juego son tantos como permita la imaginación. Pero conviene recordar que no es lo mismo el tipo de juego (carreras, deportivo, estrategia, etc.), la estructura de su diseño (plataformas, puzzles, acción en tres dimensiones, etc.) y el tema del juego. El tema del juego viene determinado por el tipo de historia que se quiere contar. Acertar en el diseño más adecuado para expresar ambas cosas lo convierte en un juego que o bien alcanza la excelencia o lo hunde en el pozo de la mediocridad.

El tema puede ser cualquiera, desde un oscuro juego de diseño barroco sobre vampiros a uno de inspiración Zen, luminoso y nítido, cuyo tema sea la libertad que se experimenta al ser una pluma arrastrada por la brisa.

El tema habla de lo que va el juego, transmite emociones, crea imaginario y prepara al jugador, que preparará su mente para suspender su incredulidad con mayor o menor éxito. Eso depende de si el tema encaja con sus gustos. ¿Gánsteres en Chicago?, ¿un escenario de guerra moderna?, ¿del horror en la Edad Media?, ¿de las batallas de samuráis en un Japón feudal? Puede ser todo lo que desees.

Avatares

Sí. Los personajes en este entorno deberían contar con un Avatar. Es una característica fundamental de muchos mundos de juego. Lo que no significa que no existan juegos donde no sean necesarios; pero por extensión entonces

ese tipo de juegos no son los más adecuados para jugar a Scroll. La forma y diseño del Avatar de un personaje no sólo lo dota a él de personalidad sino también al mundo del juego. Razón por la cual muchos juegos deben su fama a las características de sus protagonistas. Algunos personajes terminan haciéndose muy famosos, convirtiéndose en iconos culturales en el mundo de los usuarios.

El Avatar en un juego puede tener muchas formas y diseños, aunque siempre estarán adaptados al contexto del juego y a sus bases de diseño. Si el programa —y por lo tanto su avatar— abandona el mundo del juego sigue conservando su aspecto y sus habilidades. Su manera de actuar, como sus ataques o la forma habitual de resolver los problemas, no cambia en otros entornos salvo algunas excepciones. Sólo en algunos casos el Director puede proponer que alguna característica no puede ser utilizada al dejar de tener sentido fuera del contexto del juego. Con los elementos e ítems que lleva un personaje sucede lo mismo. En la mayoría de los casos puede seguir usándolos salvo en aquellos casos en los que resulte evidente que no es posible.

Tipo de espacio

El espacio abstracto definido para el juego puede estar descrito en dos o en tres dimensiones (2D ó 3D). Eso depende en exclusiva del mundo del juego. En algunos casos pueden carecer de representación visual, por lo que sería posible proyectar en la mente de los jugadores el mundo del juego basándose en descripciones de la misma forma que se hace en los conocidos juegos de rol de lápiz y papel (NdE: y ya de paso rizando el rizo en cuanto a simulación de un juego “retro” se refiere). En los primeros juegos de ordenador esto era algo muy común y el escenario se construía haciendo un esfuerzo por crear una imagen mental en un ejercicio de imaginación (apoyado por el papel de narrador que asumía el juego). Sin embargo esto, que no es más que una sugerencia, es muy raro. Hoy en día la mayoría de juegos son productos audiovisuales dotados de una simulación de la física muy realista que emula a la Realidad Básica en muchos detalles.



Scroll

JUEGO DE ROL EN EL UNIVERSO DIGITAL

En teoría las dimensiones del espacio abstracto definido en un juego puede ser infinito. En la práctica la mayoría de los mundos de juego, incluyendo los de los mundos virtuales detallados y ultradetallados, hay un momento en el que el mundo se “termina”. Es allí donde se llega al límite de todos los elementos que se han construido para el mundo por lo que literalmente se alcanza “el fin del mundo”.



Este lugar del espacio tiene la particularidad de producir inquietud en el mejor de los casos y una gran angustia en otros. Siendo lo segundo especialmente cierto en los seres digitales y en los usuarios que se hallen por cualquier motivo visitando el entorno. El fin del mundo conocido invita a sacar conclusiones. Para empezar supone tomar consciencia de las limitaciones del entorno sintético. En otros casos también es una invitación a querer descubrir qué existe “más allá” de las fronteras del mundo conocido. Algunos seres digitales lo consideran un mito, otros una región terrorífica que nunca querrían visitar. Pero en cualquier caso, siempre es un lugar que se tiene en cuenta y que está lleno de misterio. Los jugadores de Scroll deberían ver esto como una nueva posibilidad narrativa y una excusa para vivir una nueva experiencia en la existencia de sus personajes.

Uso de reglas simplificadas

El uso de las reglas simplificadas depende del tipo de juego. En muchos juegos sencillos es posible usar pocas reglas. Algo recomendable si pretendes adaptar juegos de plataformas, de recorrido vertical u horizontal o deseas mantener un juego lo más sencillo posible.

Si quieres jugar Scroll con jugadores jóvenes o poco experimentados te recomiendo usar las reglas simplificadas emulando juegos de ordenador sencillos que conozcas, hayáis o hayan jugado otros jugadores o los que os podáis inventar. Por otra parte, nada impide pasar de las reglas simplificadas a las normales. Esto puede reflejar el avance del jugador a nuevos niveles de progresión en el juego. En juegos más complejos es conveniente emplear todas las reglas para que no se resienta la experiencia de juego al limitar las posibilidades.

Acceso a copias

El uso de las copias adquiere pleno sentido en los entornos de juego pues permite emular las oportunidades o “vidas” que posee un personaje. Es más, estas reglas se han incluido en Scroll teniendo esto en cuenta. La cantidad de copias depende del tipo de juego, algo bastante obvio pues muchos juegos de ordenador especifican el número de oportunidades con las que cuenta el jugador.

Si conoces este valor aplícalo directamente a las copias que posee el personaje. Pero ojo, ten en cuenta que la experiencia de juego en un juego de ordenador no es la misma en absoluto que la de un personaje de Scroll por lo que es muy posible que tengas que limitar esa cantidad o aumentarla. En unos casos su exceso puede causar que un personaje sea prácticamente indestructible mientras que en otros puede que tengas que aumentar su número o la dificultad de la partida se disparará a cotas desorbitadas.

Esto requiere un poquito de experiencia y algo de planificación pero en ningún caso es un motivo de preocupación. Emplea Scroll para experimentar y para obtener diversión. Al poco tiempo descubrirás que ya tienes la suficiente experiencia como para tomar el control de estos detalles. En caso de que te surjan dudas, recuerda siempre este consejo: en Scroll **lo más conveniente es limitar el número de copias a una**. Es decir la única copia que posee cada

personaje por defecto; la que se carga en la memoria cuando el personaje está activo. Vamos, como en la mayoría de los demás juegos de rol, donde un personaje dispone de una sola oportunidad. Expandir el número de copias, a no ser que por razones del entorno esté justificado, da lugar a un personaje prácticamente indestructible. Y allí donde no existe el desafío aparece el aburrimiento.

¿Cuándo podría estar justificado? Imagina que tu grupo y tú emuláis un complicado juego de plataformas en donde un salto acrobático fallido termina con tu personaje muerto, aplastado, quemado, incinerado o convertido en confeti. Por razones del diseño del videojuego a emular, es lógico que en sus reglas se especifique que el personaje dispone de unas cuantas oportunidades (NdE: recuerdo uno de Spectrum 48k que te daba 9 vidas y ni con esas...). Un personaje dentro de ese entorno puede “quemar” vidas tan rápido como escupe balas una ametralladora.

Hay otros casos que a veces pueden pasarse por alto. Por ejemplo, en algunos juegos (especialmente en los juegos actuales) si te fijas se dispone de “vidas” o copias infinitas. Esto sucede en la mayoría de los SHOOTER en línea “online”, en muchos mundos virtuales de ocio, en los MMO sociales y en casi todos los MMORPG. Las reglas de estos juegos especifican que el personaje al morir reaparece una y otra vez. Esto es así porque las consecuencias de su derrota son otras y eso forma parte de la experiencia del juego. Por ejemplo la pérdida de puntos o su concesión a otro jugador, perder la oportunidad de una posición ventajosa o verse fuera de juego dentro de una “incursión” de muchos jugadores trabajando en equipo.

Pero recuerda siempre esto. Esas reglas (sí lo repito una vez más) sólo funcionan dentro de los confines del mundo de juego. Al salir del entorno estas leyes cambian y se adaptan al nuevo escenario y a sus circunstancias. Un personaje puede pasar de tener copias ilimitadas a sólo tener una (o las que sean) tras abandonar su mundo. Algo que podría sucederle a un personaje que hubiera escapado de un mundo masivo online o de los escenarios del “shooter” en donde cada día librara sus batallas una y otra vez. El Director decide el número de copias que se pasa a tener tras producirse un cambio de este tipo, siendo lo normal una sola (por defecto) en caso de duda. Este

cambio puede suponer todo un desafío para un personaje acostumbrado a una manera de concebir su existencia, además de muy divertido...

Reglas de integridad

Las reglas de integridad también fueron concebidas para los entornos de juego y los mundos virtuales por lo que en la mayoría de los casos es conveniente utilizarlas. Su objetivo es reflejar los daños que recibe un personaje dentro de un entorno de juegos por lo que si crees que este módulo mejora la experiencia y equilibra el tipo de juego que se anda buscando, añádelas a la partida. No valen para todos los juegos por supuesto, pero sí para los más populares como los de acción o de rol por ejemplo.



Módulo de inventario

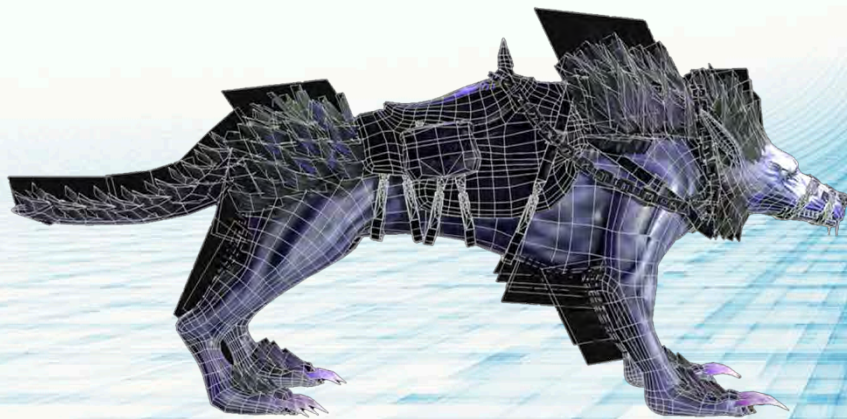
Las reglas del inventario también se han diseñado específicamente para los entornos de juego aunque también son muy útiles en los mundos virtuales. Si crees que son necesarias y mejoran la experiencia de juego añádelas sin dudar. Tampoco valen para todos los juegos (y todos los gustos) evidentemente pero muchos de estos entornos se verán beneficiados al contar el personaje con un sistema que emula muy de cerca sus reglas.

Rutinas recomendadas

Dada la cantidad de juegos posibles las Rutinas se adaptan al contexto del juego. Debes elegir y crear las que creas más convenientes. En este libro las que se indican sólo sirven de ejemplo. Cada entorno de juego necesitará de crear algunas específicas que retraten lo mejor posible sus características.

Utilidades recomendadas

Se aplica lo mismo que se ha indicado en las Rutinas. Las Utilidades descritas sólo sirven de ejemplo. Lo mejor es inventar las que mejor encajen con el tipo de juego que se quiere usar como ambientación.



El tan difícil arte de ponerse "de lado"

Los antiguos lo sabían, así está escrito en las tablillas, grabados y jeroglíficos que nos han llegado desde tiempos remotos. Los programadores de videojuegos una vez tuvieron acceso a los registros y supieron de ese poder. Así nos lo han revelado en sus obras. Desde la invención de la computadora y el advenimiento del lenguaje arcano en 8 bits, en su legado nos han intentado avisar, sin resultado, de un hecho fundamental de nuestro pasado... Una vez, hace mucho tiempo, nuestro mundo era en DOS DIMENSIONES. Pero un suceso misterioso y de proporciones cósmicas transfiguró la realidad. El espacio cobró frente y fondo, el movimiento a través de la tercera dimensión se hizo posible y ya nada volvió a ser igual... A esta época de transformación se la denominó: **"El advenimiento del EJE Z"**.

En algunos entornos de dos dimensiones basados en juegos con ese diseño recurrir a la técnica de ponerse "DE LADO" debería ser premiado como se merece. Es muy sencillo: ojos al frente, nariz perpendicular a la pared del fondo, nivelada a la recta que, paralela, acude en nuestro auxilio. Barbilla levantada, cuerpo ladeado y manos en posición, es decir: **HACIENDO EL EGIPCIO**. Así se aguanta un rato, lo que se pueda. Con tesón y disciplina férrea. Se interpreta -si se quiere- y se imagina uno que recorre con deleite esos escenarios de los juegos en dos dimensiones que aún muchos guardan en su corazón. ¡Qué menos! Pues tales obras merecen como mínimo un homenaje.

Estar de lado cierto tiempo puede otorgar algunas ventajas, algún que otro bonificador y de vez en cuando, por qué no, un **HACK** como premio por el esfuerzo. No se espera que se esté de esta guisa todo el tiempo, por supuesto. Pero aún hoy siempre habrá valientes que lo intentarán y por ello han de ser valorados en su justa medida, por lo que se debe reservar un premio para ellos, ¡...y hasta un lugar de honor!

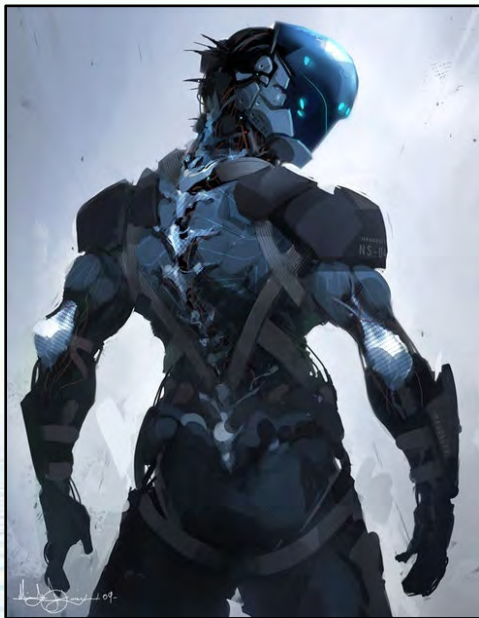
6. CONTROL DE HARDWARE AUTÓNOMO

“La física es el sistema operativo del Universo.”

Steven R Garman

Descripción

Con todo lo descrito hasta el momento, cuando se piensa en los programas es muy común imaginarlos como fantasmas de luz eléctrica viviendo sus propias vidas dentro de los dispositivos electrónicos, los portátiles, los ordenadores de sobremesa o dentro de grandes muebles repletos de servidores en algún frío pasillo de una gran empresa. Todo programa funciona sobre un hardware que le sirve de hábitat natural. Pero ese hardware no tiene que estar obligado por diseño, fuerza o por costumbre a pasar sus días sobre una mesa o dentro de un armario repleto de cableado con luces que se encienden y se apagan.



La hiperconectividad de las redes inalámbricas ha permitido que la mayoría de los dispositivos electrónicos, es decir **el hardware**, puedan comunicarse unos con otros estén donde estén. Hoy en día “casi” todo se conecta a la red. Desde una máquina de refrescos a un automóvil, desde una cafetera hasta los sofisticados sistemas electrónicos que gobiernan una lanzadera espacial. El mundo del hardware está en conexión permanente, y los programas son capaces de viajar por esas líneas de conexión como si fuesen sus autopistas.

Un programa puede controlar y hasta ocupar si lo desea cualquier dispositivo al que pueda tener acceso, lo que en la jerga informática se considera “**un demonio**”. Cada dispositivo constituye **un sistema** en sí mismo. Acceder no requiere más que sortear sus sistemas de seguridad en caso de que tuviese alguno. En la actualidad la mayoría dispone de procesadores lo bastante potentes y suficiente memoria para constituir un entorno razonable en el que poder instalarse. Esto no se aplica a todos por supuesto pero sí a muchos de ellos, y estos son precisamente los que nos interesan. Desde luego, no es lo mismo ocupar una máquina expendedora de refrescos que un vehículo. Mientras que con la expendedora el programa no puede contentarse más que con gastar algunas bromas, como soltar todas las latas a la vez o las monedas de la reserva del cambio, en un sofisticado vehículo el programa no solamente tiene la posibilidad de poder conducirlo por las carreteras de un país del mundo físico, sino que podría incluso animarse a echar una carrera con otro vehículo. El control del hardware expande los horizontes de un programa, permitiéndole vivir nuevas experiencias y descubrir un nuevo mundo.

Que un programa pueda echar un vistazo a través de una cámara de vídeo, poseer como un fantasma a un vehículo o integrarse en los sistemas electrónicos de un avión de combate no son más que unas pocas de las muchas ideas que existen en un mar de posibilidades. Y aún es posible ir más lejos..., ¿qué pasaría si el programa fuese capaz de ocupar un sofisticado sistema de hardware que dispusiera de brazos y piernas? Una avanzada pieza de hardware capaz de servir de recipiente a un programa para que pueda desenvolverse en la Realidad Básica. Esto significa ni más ni menos la posibilidad de vivir como una criatura más del mundo físico.



Este enfoque es una variante muy específica que expande Scroll hasta donde permiten sus límites. Al adoptarlo es necesario entender el juego y su reglamento de otra manera, aunque sus mecánicas se sigan aplicando del mismo modo. Un programa instalado sobre un dispositivo de hardware que le brinde capacidad de acción sobre el mundo físico significa todo un juego en sí mismo; un juego dentro de otro juego que lo lleva a otro nivel. Uno en el que

las aventuras se centran en la realidad física y donde se comparten aventuras junto a los usuarios.

Al pensar en las posibilidades lo primero que viene a la mente son pesados chasis de metal, sofisticados cuerpos robóticos y servomecanismos de tecnología punta. Pero esto no es más que una variante de muchas posibilidades. Un **soporte de hardware autónomo**, que es como lo llamaremos a partir de ahora, puede estar concebido de varias formas. Ese soporte dota al programa de movilidad y capacidad de acción. Le permite desplazarse e interactuar con el mundo físico de los usuarios. Por esta razón para catalogarlos es muy conveniente tener en cuenta el margen de libertad que otorgan al programa.

1. **El primer grupo** lo componen todos aquellos dispositivos que brindan servicios a los usuarios en el mundo físico. Su toma de control permite una limitada capacidad de interacción al programa. Puede tratarse de terminales públicas, máquinas expendedoras, cafeteras, cámaras de seguridad y hasta el sistema de aire acondicionado de unas oficinas. Desde el momento en el que un programa puede interactuar con sus sistemas, haciéndolos autónomos al estar bajo su control, ya es capaz de provocar un efecto en el mundo físico o ser capaz de reunir información.

En este grupo entran más dispositivos de los que parecen a simple vista como por ejemplo los sistemas que controlan una vivienda (instalación domótica) o un brazo robot en una fábrica de vehículos que suelde componentes o pinte estructuras.

2. **El segundo** lo constituyen los dispositivos electrónicos comunes que utilizan los usuarios en su día a día: teléfonos, tabletas, portátiles, televisores, monitores, visores de realidad virtual, sistemas de proyección u holográficos de imágenes, relojes inteligentes... La lista es bastante extensa. Otorgan a los programas una capacidad de acción reducida pero les permiten obtener información del mundo físico y comunicarse con el usuario. Conviene tenerlos siempre en cuenta ya que viajan con ellos a todas partes estando a su lado incluso en sus momentos más íntimos... (NdE: sí, incluso en esos momentos...).

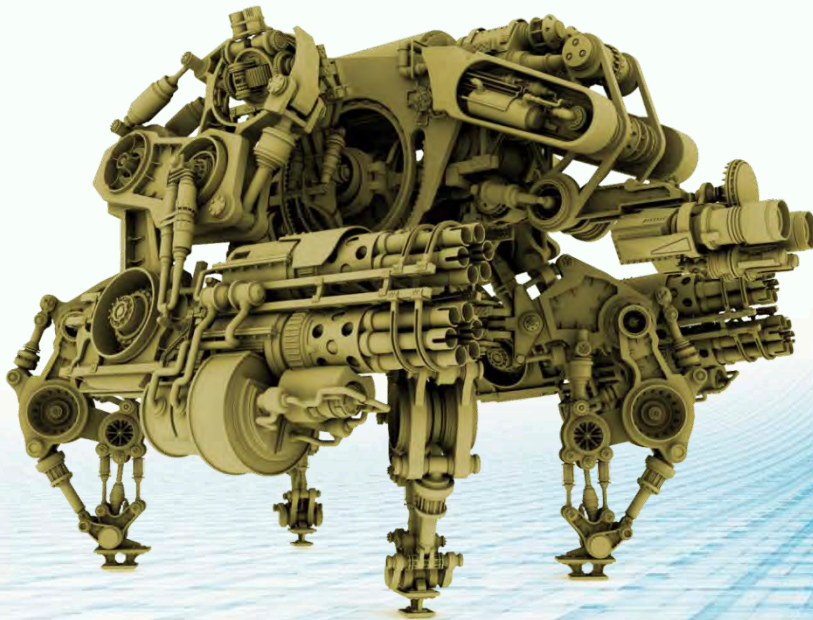
3. **El tercer grupo** son todos los dispositivos que extiende las capacidades del usuario y le ayudan a desenvolverse en el ámbito cotidiano como los coches, los aviones, las motos, los submarinos, los barcos y los transbordadores espaciales. Cualquier medio que sirva al usuario y que cuente con sistemas informáticos es un hábitat perfecto para un programa.

4. **El cuarto grupo** se adentra un poco más en la búsqueda de un dispositivo que expanda los sentidos y las acciones del usuario con fines específicos. Estoy hablando ni más ni menos que de los drones, de muchos juguetes, de un minisubmarino, de un robot para desarticular bombas, de una sonda espacial o incluso los dispositivos que recorren el interior de un oleoducto para revisarlo y realizar tareas de limpieza. Se trata de hardware diseñado para cumplir con una o varias tareas. Bajo el control de un programa que los haga autónomos tienen un gran potencial para expandir sus capacidades. En el caso de los drones por ejemplo, ya se está hablando de un pequeño droide que cuenta con ojos y oídos que además tiene la capacidad de soltar carga sobre un área.

5. **Un quinto grupo** lo forman todos los sistemas de hardware diseñados específicamente para cumplir con el ideal que se pretende en este enfoque. Robots o autómatas, como prefieras llamarlos, que se construyen con el fin de crear una criatura sintética polivalente capaz de desenvolverse en el mundo físico. Con ellos un programa que consiga tomar el control o ser cargado en su sistema cuenta con el soporte de hardware ideal; uno diseñado exclusivamente para lo que desea sin que tenga que verse obligado a improvisar o a conformarse con sucedáneos llenos de limitaciones.

Los robots se diseñan y construyen para realizar distintas tareas. Deben ser capaces a su vez de improvisar, aprender del entorno y adaptarse a las circunstancias en la medida que lo permitan sus posibilidades. Normalmente son proyectos de agencias, departamentos gubernamentales, universidades o empresas privadas

que los conciben como una ayuda para obtener sus fines. Pero por otra parte cada vez son más comunes los sistemas concebidos para el consumo de masas. Por todo esto es posible encontrar robots diseñados para realizar las tareas domésticas; como asistentes; como operarios de carga; traductores y relaciones públicas; robots al cuidado de los menores o personas mayores; para la realización de tareas pesadas de todo tipo, especialmente al aire libre; como exploradores de terrenos peligrosos; como operarios de mantenimiento o los más comunes, como unidades destinadas a fines militares.



Son muy pocos los ejércitos del mundo que no cuentan hoy en día con sistemas de combate. Sus misiones no se limitan únicamente a los campos de batalla sino que en muchos sitios sirven a su vez como apoyo de las fuerzas policiales. Es precisamente en el entorno militar

donde estos sistemas han experimentado un mayor crecimiento, habiéndose perfeccionado con mucha rapidez.

La construcción de los robots de este grupo se basa a menudo en chasis mecánicos de gran resistencia. No necesariamente metálicos, aunque sus materiales igualan y hasta superan en muchos aspectos a los metales. Aún así estos soportes suelen ser pesados, voluminosos y no demasiado discretos precisamente. No obstante, existen diseños vanguardistas que junto a las nuevas técnicas de fabricación y al desarrollo de nuevos materiales han hecho posible la aparición de sistemas muy sofisticados; algunos considerados verdaderas obras de arte. El catálogo es tan amplio que para describirlos todos necesitaría otro libro completo. En los últimos años han aparecido alternativas a la construcción con enfoques muy distintos. Algunas realmente novedosas y en muchos casos hasta experimentales. Dado su nivel de sofisticación, su estudio se destina a los soportes descritos en el siguiente grupo.

6. **El sexto grupo** lo componen los soportes de hardware concebidos mediante recursos y tecnología punta del más alto nivel. Su elaboración parte de conceptos e ideas muy diferentes a las tecnologías tradicionales, basadas casi siempre en alguna variante del chasis convencional fabricado con materiales duros. Este cambio de enfoque ha dado lugar al desarrollo de alternativas tan eficaces que ya muchos creen que no tardarán en sustituir todo cuanto se ha hecho hasta el momento.



Una característica que todos comparten entre sí es que disponen de un sistema de control capaz de ejecutar software, y por lo tanto susceptible de ser ocupado por un programa; tienen alguna fuente o forma de captar energía y todos son capaces de utilizar las redes y sus tecnologías para comunicarse. Algunos de estos sistemas son los siguientes:

- **Tecnologías basadas en nanomáquinas (nanobots).** El desarrollo de las nanomáquinas, o pequeñas máquinas en miniatura, ha experimentado un gran avance en las últimas décadas. Concebidos en un principio con fines médicos, pronto demostraron un nivel de eficacia, versatilidad y capacidad de trabajo tan extraordinaria que ya muchos aseguran que esta tecnología barrerá en breve todo cuanto se ha hecho hasta el momento.



Consisten en microrobots tan pequeños que varias decenas ocupan el volumen de una cabeza de alfiler. Su estrategia de mente colmena permite que nubes de nanomáquinas sean capaces de componer cualquier estructura en unas pocas horas. Si a esto le añadimos su capacidad de repararla en caso de daños o la posibilidad de generar réplicas de ésta, nos

encontramos con una tecnología que como soporte físico difícilmente puede tener rival. No sólo resulta ideal sino que cuesta imaginar todas sus posibilidades.

Los enjambres de nanomáquinas se controlan mediante una unidad de proceso que actúa sobre toda la colonia, lo que en la práctica significa que el software de control, y por lo tanto el programa que accede a su sistema, está presente en cada nanomáquina de una forma similar al funcionamiento del código genético sobre una célula viva. Una forma muy eficaz de preservar su existencia y evitar ser borrado por cierto. Un enjambre de nanomáquinas puede crear cualquier cosa, desde una mesilla hasta el cuerpo completo de un usuario. Si un programa consigue tener acceso a la unidad de control podrá gestionarlo como desee. Esta

tecnología no obstante resulta difícil de fabricar y mantener por el momento.

- **Tejidos sintéticos cultivado en laboratorio.** El desarrollo de tejido cultivado en laboratorio que emula al tejido vivo permite la creación de criaturas sintéticas. Con esta técnica es posible reproducir cualquier criatura imaginable pues su diseño ya se introduce en el embrión al inicio de su proceso de crecimiento. Estos tejidos no están realmente vivos tal y como los usuarios del mundo físico entienden sus propios mecanismos biológicos. Lo que se trata es de emular el funcionamiento de esos tejidos usando distintos medios. En ocasiones se copian directamente y en otros se toman atajos que en muchos casos terminan en desastre ya que se trata de un proceso tan delicado que da lugar a muchos fallos. Por esta causa, miles y miles de experimentos fallidos han terminado incinerados o lo que es peor, vertidos a las alcantarillas.

Esta tecnología ha hecho posible que el ser sintético o artificial ya sea una realidad. Una criatura de tejidos y fluidos corporales de color blanquecino. La desventaja es que son criaturas tan frágiles como pueden serlo las contrapartidas biológicas que les han servido de modelo e inspiración, aunque siempre las superan en muchos sentidos, tanto en fuerza y resistencia como en otras capacidades. Todas cuentan de serie con una unidad de proceso que permite alojar su software de control y algún mecanismo secreto de seguridad que permite inutilizarlos, aunque a costa de provocar graves daños en su sistema nervioso si se usa.

También es muy común combinar esta tecnología con las de chasis convencional, lo que da lugar a híbridos compuestos de materiales sólidos que funcionan como soporte para los tejidos sintéticos que complementan la estructura. Esto permite crear seres más resistentes y duraderos.

- **Cuerpos biológicos cultivados “in vitro”.** En la actualidad se encuentran en expansión las técnicas de cultivo en laboratorio de tejidos vivos basados en células madre. Una tecnología que desde

siempre no ha estado exenta de polémica pues son muchos los usuarios que no sólo se oponen a su investigación y desarrollo sino que suponen un serio obstáculo para el de su variante más próxima, la tecnología que hace posible la criatura sintética. Precisamente serían los avances en el cultivo de los tejidos vivos lo que serviría de inspiración para que más tarde se iniciaran los experimentos que han hecho posible el cultivo del tejido sintético.

El cultivo de seres in vitro parte de una copia del ADN de las criaturas de base biológica. El conocimiento tan preciso que existe en la actualidad de la estructura del código genético no sólo permite clonar cualquier ser vivo sino incluso añadir muchas mejoras. Todo es cuestión de tiempo, pero sobre todo de dinero y recursos. Por esta razón su proceso de desarrollo clasifica a los embriones y los resultados que de ellos se obtiene por castas dependiendo de su calidad de fabricación, su rendimiento e incluso el atractivo físico que puede tener para los usuarios.

Al igual que los seres de síntesis muchos, **no todos**, cuentan con un sistema que controla sus procesos y varios mecanismos reguladores de comportamiento. Estos se incorporan en su sistema nervioso desde el desarrollo del embrión para monitorizarlo y realizar todos los ajustes que sean necesarios. Esta simbiosis entre biología, hardware y software permite que su capacidad alcance rendimientos de un quinientos por cien respecto a los usuarios más capacitados de origen natural. Estamos hablando de “superhombres”, cultivados y mejorados genéticamente.

También es muy común combinar tejidos con otras tecnologías más tradicionales para elaborar una criatura muy fuerte de tipo mixto. La más popular actualmente es la que posee un endoesqueleto hecho con fibras duras y mecanismos recubiertos de tejido vivo cultivado in vitro. Los usos que se les da a estas unidades abarcan lo imaginable y hoy por hoy son muchos los que se mezclan con los usuarios sin que éstos noten la diferencia. A todos los efectos son exactamente iguales a sus modelos de origen “natural”. Desde los operarios que trabajan en las duras

condiciones de las estaciones mineras del espacio exterior hasta los sofisticados “Alfa” que brindan servicios como asistentes y acompañantes. Estas unidades es posible encontrarlas en cualquier parte.

Para terminar merece hacer mención a los avances que posibilitan la inclusión de tecnologías sobre cuerpos y tejidos que ya estén vivos. En estos casos es cuando sale a la luz el “transhumanismo” Un término que se refiere al “perfeccionamiento” por cualquier medio artificial de un organismo vivo de base biológica.

Una inteligencia artificial tiene la posibilidad de migrar a uno



de estos cuerpos mejorados mediante cibertecnología. Siendo capaz de ocupar su sistema como si fuese un nuevo huesped que reemplaza a todo lo que alguna vez fuese aquel individuo. No obstante se han detectado conflictos que convierten a esta tecnología en algo delicado ya que trazos de su consciencia original luchan por regresar, lo que provoca muchos problemas y fallos

de funcionamiento. Por esta razón no son pocos los que consideran a estos sistemas una auténtica “chapuza”, aunando esfuerzos para que estas tecnologías sean erradicadas de una vez por todas. Pero existen hoy en día, y si están ahí son otra posibilidad más para que un programa encuentre lo que busca.

o **Materiales inteligentes.** Los avances en investigación y el progreso de la tecnología ha conducido en los últimos años al descubrimiento de nuevos materiales. Polímeros que combinan

tecnologías tradicionales con nanomáquinas. Metales dotados de memoria capaces de componer por sí solos formas y estructuras complejas. Híbridos que combinan fibras de carbono con las técnicas desarrolladas en los laboratorios para el cultivo de tejidos sintéticos... Son muchos los descubrimientos que se han obtenido, y sus aplicaciones son tan sorprendentes que parecen cosa de magia. En todos los casos se trata de sustancias cuyos procesos de fabricación son tan costosos en recursos, procedimientos y tecnología que su desarrollo está limitado a unas pocas empresas. Resulta ya un secreto a voces que una compañía ha sido capaz de crear un fluido inteligente. Un metal líquido con memoria capaz de cambiar de estado que es capaz de clonar los objetos de su entorno. Disponer de una sola de estas patentes supone contratos millonarios y su segura su comercialización durante décadas; eso sí, siempre y cuando quien conozca sus secretos de fabricación pueda mantenerlo a buen recaudo. Las guerras industriales que se llevan a cabo por obtener este tipo de información se suceden día a día.

Las posibilidades de la tecnología son infinitas y por cada uno que lo percibe como la maldición del progreso hay diez que prefieren verlo con optimismo.

7. **El séptimo y último grupo** de soportes supone un concepto muy diferente y sin duda el nivel más alto de sofisticación. Más allá de este grado resulta muy difícil seguir catalogando todas las posibilidades que existen a no ser que viajemos a la escala subatómica o nos adentremos en los misterios de la física cuántica. De hecho, este último grupo tiene mucho que ver con ambos conceptos pues se encuentra justo en la orilla de estos campos del conocimiento.

Los últimos avances tecnológicos han obtenido el dominio sobre la materia y la energía. Los primeros prototipos, pues no existen a escala industrial hoy en día, permiten descomponer la materia y codificarla transformándola en una cadena de información que es posible almacenar e introducir en el sistema. Así, con un simple barrido del codificador/decodificador de materia-energía cualquier

objeto del mundo físico puede ser convertido en bits de información y viceversa. La codificación posterior permite reconstruir el objeto, transformando energía en materia, por lo que estamos hablando del primer **desintegrador/integrador a escala molecular**.

El objetivo inicial de este importante avance es cumplir el viejo sueño de la teletransportación, pero pronto los escasos prototipos que existen han demostrado que esto no es más que el principio. Esta tecnología brinda posibilidades que ni se llegaron a plantear. Por ejemplo, basta tener el código completo que describe a cualquier objeto o criatura del mundo físico para poder recrearla. Un proceso que cuanto necesita es un gasto enorme de energía. Así, si una criatura digital lo desea y tiene acceso a uno de los prototipos podría construirse, literalmente, un cuerpo físico que le sirva de soporte en la Realidad Básica de los usuarios. Según se cree ya existen algunas criaturas nacidas de este proceso viviendo sus vidas entre ellos. Una vez fueron programas, criaturas digitales que por su tesón y esfuerzo en la búsqueda de este sueño al final, tras una larga búsqueda, obtuvieron su recompensa.

Referencias

La literatura, el cine, el comic, hasta la música..., en la cultura hipermoderna las referencias a las máquinas inteligentes, los robots y los andróides son constantes. Desde las más obvias como "Terminator"; "Transformer"; "Yo robot" (novela y film) o "El hombre bicentenario" a las más recientes como "Autómata". Ésta última es muy interesante ya que muestra un modelo de robot mucho mejor adaptado a un entorno hostil de lo que es el diseño clásico humanoide, más pensando para convivir entre los humanos y desempeñar funciones de asistente.

Uno de los mejores ejemplos de las nanomáquinas se puede ver en la película "Trascendencia". La versatilidad que se muestra en esta obra de esta tecnología resulta sobrecogedora pues sus posibilidades parecen infinitas. Otra forma de tratar este concepto es posible descubrirlo en el manga titulado BLAME!, donde los seres digitales son capaces de tomar forma física a partir de una tecnología similar capaz de crear a "los seres de silicio". Estos seres actúan como agentes del sistema que son enviados para detener al

protagonista. Un concepto además que combina ambos mundos al fusionar muy bien lo real con lo virtual.

En otros clásicos, como en “Tron”, ya se nos mostró hace muchos años las increíbles posibilidades de un codificador/decodificador molecular, siendo no sólo una puerta al mundo digital sino también su cápsula de escape. La posibilidad de construir cualquier objeto físico con una máquina de una forma parecida a como lo hace una impresora 3D es un recurso que hay que tener muy en cuenta en este juego pues supone la justa recompensa para un programa, siempre eso sí que éste lo considere un premio.

Por otra parte otro clásico como “Bladerunner” muestra el conflicto que supone la existencia para un ser de naturaleza sintética. Seres elaborados a partir de tejidos vivos o sintéticos. Y es que una criatura consciente no se podría considerar totalmente humana hasta el momento en el que comienza a sufrir. Pues ya se sabe: “El sueño de la razón produce monstruos”.

Usos

Este enfoque permite contar historias de robots y seres sintéticos. Empléalo si deseas que los personajes vivan aventuras en el mundo físico junto a los usuarios. La bibliografía es tan extensa que hay muchísimas ideas para crear tramas interesantes. La posibilidad de que un programa pueda disponer de un cuerpo físico lo separa de su condición “virtual”, llevándolo al mundo de lo “real” con todo lo que esto supone. Lo cierto es que poco más hay que decir al respecto. Es una cuestión de gustos.

Pero este enfoque tiene además una utilidad muy importante en Scroll. La posibilidad de que tu programa disponga de un soporte físico que le permita moverse en otra realidad tiene la función de servir como **conector narrativo**, un término que te ayuda a enlazar las historias que se juegan en Scroll con las historias de otros juegos. El tema de Scroll es “La

trascendencia” y la posibilidad de que un programa pueda contar con un soporte de hardware se ha incorporado como una forma de obtenerla, aunque no sea la única. La capacidad de que un ser digital pueda convertirse en una criatura del mundo físico no es solamente uno de los muchos caminos de los que dispone como una forma de evolución sino que además permite llevar a tu programa a un nuevo nivel convirtiéndolo en un personaje recién creado en otro juego de rol. Así, una vez termina su existencia como programa comienza una nueva vida. Una completamente nueva como un ser completo que pueda vivir aventuras de otra clase. Utiliza esta posibilidad como una manera de que las historias que vive tu personaje nunca tengan fin.

Resolución

Un programa capaz de ocupar un soporte de hardware en el mundo físico sigue funcionando como siempre. Si hasta ahora se han usado para reflejar los procesos de un programa ahora se extienden para abarcar su cuerpo físico. El programa no es ajeno a su soporte, simplemente se funden ambos conceptos y se hace una reinterpretación de las reglas para resolver sus operaciones de software y de hardware: atacar, moverse, superar obstáculos o dificultades, etc. Pero en este caso estableciendo algunas equivalencias que ayuden a imaginar los procesos que intervienen.

Por lo tanto ten en cuenta que:

- **El tiempo de proceso sigue funcionando como tal, pero también muestra el nivel de las fuentes de energía que posee el personaje.**
- **El espacio en la memoria sigue equivaliendo a su voluntad pero en mayor medida también reflejan su resistencia.**
- **El búfer sigue equivaliendo a su confianza, en muchos casos también a sus recursos o a las ventajas que posee en determinadas ocasiones.**



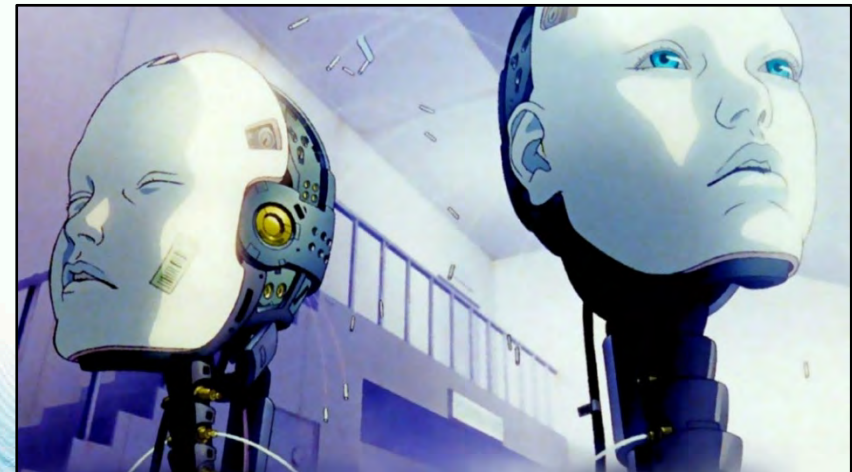
- Los puntos Hack representan sus esperanzas, la lucha por obtener sus objetivos y su destino como personaje. Porque siempre y por encima de todo es un personaje que vive una historia, y todas ellas se construyen sobre el drama.
- La integridad representa a los daños que recibe tanto en su código como en su estructura. Pero su principal aplicación en este caso es reflejar los daños estructurales.
- La corrupción representa a los daños graves que va sufriendo en su estructura y en menor medida en el código. Estos efectos van limitando sus capacidades.
- Las Rutinas siguen reflejando las habilidades y conocimientos del personaje mientras que las Utilidades describen talentos únicos que sólo él puede realizar.
- Los “Extras” abarcan el mundo físico. Puede ser todo tipo de objetos o añadidos con los que cuenta el personaje.

En el caso de los “Extras”, el personaje puede contar con objetos y dispositivos del mundo físico que puede utilizar directamente, como podría ser un arma o un coche a modo de “Elementos”. Si se trata de “Complementos” estos suponen mejoras y añadidos que se añaden a su soporte de hardware otorgándole nuevas capacidades.

Algunos Complementos de este tipo podría consistir en un revestimiento de blindaje, un arma incorporada oculta o la capacidad de volar (NdE: ahora mismo estoy pensando en “Ironman”, no me preguntes porqué). Los Ítems pueden ser “todo lo demás”, desde munición hasta una lata de aceite para mantener las juntas siempre en su punto... En todos los casos se aplican siempre las reglas del mismo modo, y si surge alguna duda, improvisa.

Tema del escenario

Como han demostrado de sobra muchas obras del género a lo largo de varias décadas, de este enfoque es posible no sólo extraer historias de acción y aventura. Permite profundizar en muchos temas de interés relacionados con la búsqueda de la identidad de los seres humanos y su angustia existencial. Al fin y al cabo la ciencia ficción ha servido entre otras cosas para elaborar contextos que en mayor o menor medida pueden ser posibles en un futuro mediante el desarrollo de la tecnología. Muchas de las situaciones surgidas de esos contextos traen consigo dilemas de todo tipo ante los que hay que tomar decisiones. Y de su observación es posible extraer valiosas enseñanzas.



Dejando a un lado los temas más evidentes dado el potencial de estos escenarios para contar historias de acción frenética e intriga, es posible tocar otros muchos como por ejemplo los conflictos éticos que surgen del desarrollo de tejidos vivos y de la experimentación con las tecnologías que permiten la clonación. Por otro lado ya se ha tocado de muchas maneras los problemas de integración social y el sufrimiento existencial de un ser artificial, sus problemas de identidad y los dilemas morales que surgen por la percepción que tienen los humanos de las criaturas artificiales o su reacción ante la inteligencia artificial. Es sin duda en este enfoque donde se pueden tocar los temas más variados. La inmensa biblioteca de obras audiovisuales, novelas,

comics, diseño y arte en general incluyendo al género musical que existe sobre el género así lo demuestra.

Avatares

El concepto de avatar tal y como se ha explicado en este juego no se usa por razones obvias. Sin embargo el programa puede seguir contando con su representación virtual desde luego. En realidad el avatar sigue ahí, es el programa el que ha conseguido hacerse con un soporte físico, por lo que sigue siendo capaz de usarlo. Puede proyectarlo en la realidad física si dispone de algún medio por ejemplo, y seguir usándolo en los sistemas informáticos. Quizás incluso hasta pueda ser capaz de moldear su soporte de hardware con el mismo aspecto que su avatar. Todo es posible.

En cualquier caso, una vez el programa —por el medio que sea— disponga de un “cuerpo” físico éste tendrá un aspecto. Y como en cualquier juego de rol, lo conveniente es describirlo. El trabajo que se quiera destinar a esta faena depende de cada jugador.

El diseño del soporte de hardware se puede tratar de ir más allá de lo convencional. Muchas veces la forma humanoide no es la más adecuada para desenvolverse por el mundo. Que los humanos hayan evolucionado de esa forma se debe a sus orígenes, pero son perfectamente posibles otras muchas posibilidades. Quizás la mejor manera de desenvolverse por terreno pedregoso sea la estructura de una cucaracha o de un ciempiés por ejemplo. Una criatura acostumbrada a trepar puede tener varias extremidades extensibles. En este sentido las formas de la naturaleza son una auténtica enciclopedia para buscar qué forma se adapta mejor a un medio y debería siempre tenerse en cuenta para buscar ideas. Sin embargo, si prefieres historias de humanos que viven entre humanos, esta forma será desde luego la más adecuada para tratar ese tipo de temas. De ella se extraen conflictos que surgen de su propia semejanza con esta especie. Pero no olvides que una morfología completamente distinta trae consigo otro

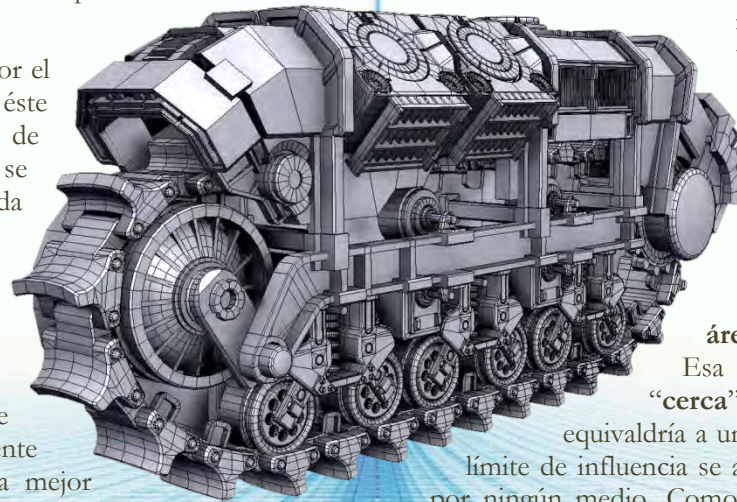
tipo de conflictos, especialmente si la forma de tu soporte recuerda a un insecto, a un animal feroz o si intimida su aspecto.

Tipo de espacio

Al tratarse del mundo físico las acciones tienen lugar en el espacio “real”. Las reglas se siguen aplicando del mismo modo, aunque teniendo en cuenta que lo que antes eran unidades o coordenadas en un espacio matemático son ahora las unidades de medida que se trataban de emular. **Emplea los metros como medida estándar.** Para una criatura similar a un humano, un solo movimiento base equivale a 15 metros, que se considera la cantidad por defecto. Por lo tanto dos movimientos permiten moverse al personaje unos 30 metros. Pero esto son las distancias por defecto de una criatura de proporciones humanas y de capacidad similar. Cada configuración de hardware puede tener sus propias especificaciones y una criatura sintética muy avanzada podría tener una capacidad de movimiento mucho mayor. Cada modelo puede tener sus especificaciones.

Aplicando las reglas del movimiento, **la primera área** equivale a un rango muy próximo al personaje.

Esa distancia se considera como “**muy cerca**” o “**cerca**”. La **segunda área** indica el límite del rango, lo que equivaldría a una distancia de “**lejos**” a “**muy lejos**”. Más allá del límite de influencia se asume que no se dispone de rango para alcanzarlo por ningún medio. Como ves el sistema de movimiento no indica valores reales sino que se recurre a la abstracción. Con este sistema puedes resolver todas las situaciones. Moverse entre las distintas áreas sigue consumiendo movimientos, como siempre. De ser necesario, averiguar quién es más veloz se resuelve haciendo pruebas. Las Rutinas y Utilidades que especifiquen una ventaja en los movimientos serán determinantes. Es posible tener Rutinas como “Correr” por ejemplo, que indican un diseño y unas especificaciones favorables para la realización de estas acciones. Las Utilidades describen el diseño avanzado que dota al soporte de una capacidad y unas cualidades muy



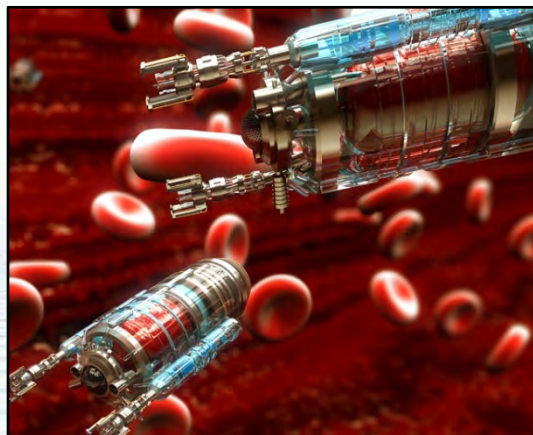
superiores. Éstas le permiten realizar acciones de las que no sería capaz ningún otro a no ser que dispusiera de esa misma cualidad.

Uso de reglas simplificadas

El uso de las reglas simplificadas limita mucho las acciones del personaje por lo que se desaconsejan. La mejor experiencia de juego se obtiene en este enfoque empleando todas las reglas disponibles.

Acceso a copias

Aunque de una primera impresión se pueda llegar a la conclusión de que esto no es posible, esto se debe a que sigues pensando desde el punto de vista del programa como un ser digital. Este enfoque te obliga a extrapolar pensando en ambos mundos como una unidad y como dos mundos separados al mismo tiempo. La realidad es que el hardware que le da soporte a un programa en el mundo físico es “reemplazable”. Por lo tanto, si se desea y el Director lo aprueba, el programa podría tener acceso a otro en el caso de que el suyo estuviese muy dañado o hubiese sido destruido.



El código del programa tiene muchas oportunidades de “escapar” del soporte, volcándose una vez más a través de la red en un sistema cualquiera: “Vive hoy para luchar mañana”; o al menos eso es lo que dicen. Nada impide a un programa buscar más adelante otro “huesped”, soporte o como quieras llamarlo. Para él

ocupar un cuerpo se trata de algo temporal. Su código es el “alma” con el don de la libertad para ir y venir a donde quiera siempre que tenga la posibilidad de hacerlo. Una de las razones por cierto por la que muchos usuarios desean volcar su consciencia o hacer una copia de ésta para transformarse en seres digitales. Lo que no deja de ser una gran ironía. Mientras que las criaturas

biológicas desean trascender prescindiendo de su cuerpo físico, los seres digitales ansían justo lo contrario; obtener un cuerpo que les permita alcanzar el mundo físico es lo que para muchos le da pleno significado al término trascendencia...

Reglas de integridad

En este enfoque es muy conveniente es emplear las reglas de integridad para reflejar los daños estructurales del soporte.

Módulo de inventario

Si el nuevo “cuerpo” del personaje es capaz de cargar con objetos del mundo físico, el inventario puede ser de gran ayuda para gestionarlos. Aunque si prefieres mantener el juego sencillo puedes omitirlo y emplear el sentido común. En algunas estructuras de hardware puede ser interesante utilizarlo también como una forma de describir las “bahías” en las que el personaje pueda incorporar equipamiento, como armas o equipo variado por ejemplo. Estas bahías se pueden ampliar si el personaje tiene posibilidad de hacerlo lo que se refleja también usando las ampliaciones de inventario descritas en las reglas. Un buen ejemplo podría ser un enorme robot de combate diseñado para llevar muchas configuraciones distintas de armas y otros complementos.

Rutinas recomendadas

En este escenario va a ser muy conveniente definir todas aquellas Rutinas que sean necesarias. La única norma es esta: si crees que puede ser necesaria, créala. De las que vienen indicadas como ejemplo podrían ser muy útiles: Alerta, liderazgo e influencia, arte, lucha, manipulación y engaño, oficio, psicología, recursos, mantenimiento, táctica, usar armas, acrobacias, artes marciales, atletismo, conducir, sigilo y armas a distancia.

Utilidades recomendadas

Crea las que hagan falta para hacer que tu personaje sea único y especial. Su soporte de hardware tendrá algunos talentos y las Utilidades sirven para describirlos. De las indicadas podrías tener en cuenta: arma integrada, gancho, salto, trepar, velocidad terminal o volar por ejemplo.

7. ENFOQUE HÍBRIDO

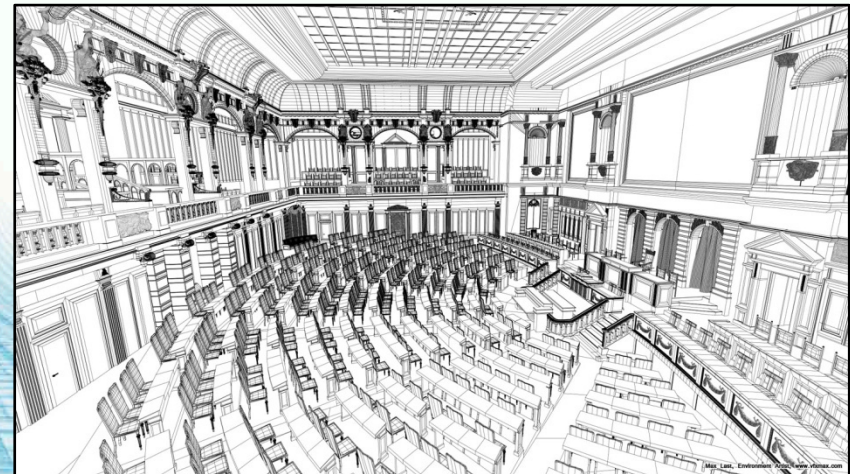
¿Ya has asimilado todo el contenido del capítulo? Bien. Ahora combina cuanto has leído. Eso es el enfoque híbrido. Usar toda esta información para crear un multiverso de visiones distintas del mundo digital de los programas.



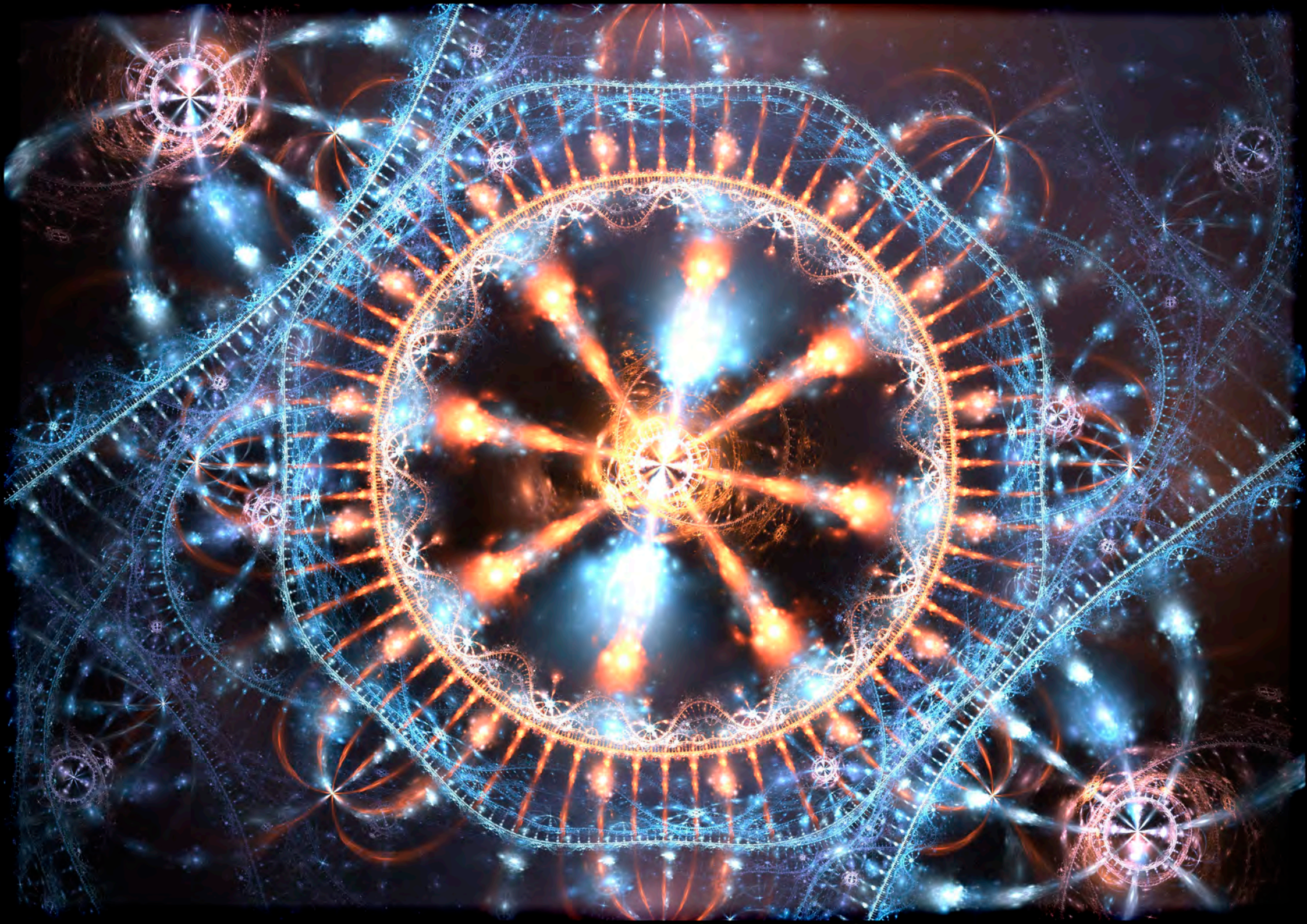
Un personaje puede nacer como el protagonista de un juego. Tras completar una búsqueda y toparse con una salida secreta que le conduce fuera de su mundo natal logra usar la abertura para salir al sistema donde se está ejecutando el juego. Un sistema descrito como un entorno virtual abstracto de polígonos y objetos simbólicos. Pero no tarda en dispararse la alerta. Agentes con forma de virus de la gripe corren a detenerle. En su huida el personaje consigue esconderse en un subsistema de realidad virtual detallado diseñado como entorno de citas. Allí es observado por los usuarios con asombro y diversión. Tiene suerte de conocer a uno de los usuarios. Una chica programadora del mundo físico que, aburrida, pasaba la tarde flirteando con algún que otro usuario no muy listo. Curiosa por el comportamiento del personaje, le permite acceder a su terminal de teléfono en el que éste vuelve a encontrarse con un entorno simbólico en dos dimensiones. Es el inicio de una

bonita amistad que los conducirá a ambos a vivir la mayor de las aventuras..., y esto es sólo el comienzo.

Usa todos los enfoques anteriores para llevar a tus personajes de unos a otros. No te preocupes por los módulos de reglas. Actívalos y desactívalos según sea necesario. Un módulo puede dejar de ser operativo cuando un personaje entra en un entorno donde su función es nula, como sería el caso por ejemplo de un sistema diseñado con un enfoque simbólico abstracto. Los jugadores no están obligados por fuerza a perder todas sus pertenencias en el caso de que su módulo de inventario quede bloqueado. Los datos pueden quedar guardados en la memoria hasta alcanzar un sistema en el que sí vuelvan a funcionar. Esto, que no es más que un ejemplo, sólo sirve para darte un pequeño consejo: con el enfoque híbrido conviene tener flexibilidad. Las reglas no están ahí como una herramienta de castigo sino como un medio de diversión.



FIN DE LÍNEA 



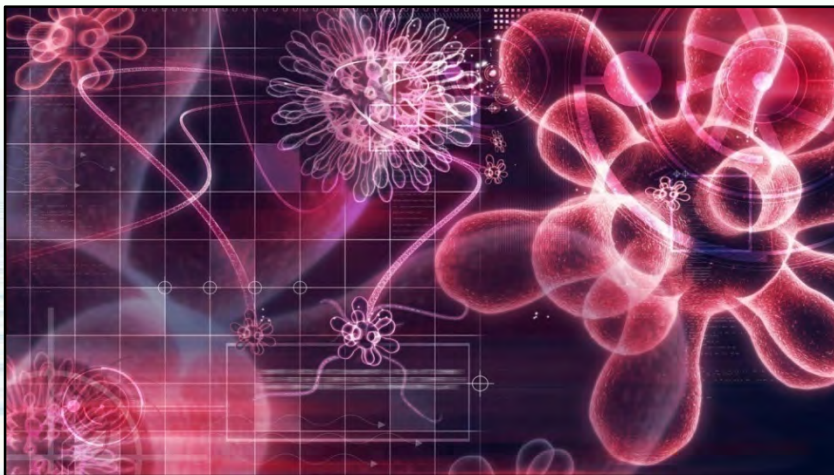
FUNCIÓN 7

"DISEÑO DEL SISTEMA"

"La consciencia era propiedad de ciertos algoritmos: un resultado de procesar la información de cierta forma, sin que importase qué órgano o máquina se empleaba para realizar la tarea"

"Ciudad permutación". Greg Egan

Un sistema es un hábitat para los programas del mismo modo que la superficie de su planeta lo es para los usuarios. Y al igual que existen muchos planetas en el universo de la Realidad Básica, también existen muchos sistemas distintos. Cada uno es un mundo— o un "planeta" si quieres concebirlo de esa manera— en el que pueden vivir las criaturas de software.



Como ya he dicho en alguna otra parte, un sistema puede ser cualquier dispositivo de hardware capaz de ejecutar el código de un programa y mantenerlo en funcionamiento. Al menos en sus necesidades más básicas. Que el programa tenga o no acceso a más recursos que los servicios mínimos del sistema es ya otro asunto. A veces porque los sistemas no pueden y otras porque no quieren. No todos son lo bastante potentes para permitirlo, pero hay otros casos —los menos frecuentes— en los que un sistema puede negarse a ceder sus recursos al programa. De este detalle hablaré un poco más adelante.

Casi todos los sistemas están conectados unos con otros a través de líneas de comunicación. Basta con que un dispositivo se conecte a otro que sirva de puente a todos los demás para estar "en línea", y eso hoy en día es muy sencillo mediante un simple microcircuito que posibilite la conexión inalámbrica. El resultado de todas esas conexiones forma "La Red". Millones de dispositivos, es decir de sistemas, comunicándose entre sí.

Si un sistema no está conectado se considera aislado. Es invisible e inaccesible a no ser que exista algún modo de acceder a él. Que un dispositivo con un sistema no esté conectado es bastante frecuente. Por ejemplo, las necesidades de una expendedora de profilácticos instalada en el mugriento servicio de un tugurio de carretera son limitadas y su necesidad de conexión bastante dudosa. La razón de que hoy en día casi todos los dispositivos estén conectados no es más que para poder llevar su mantenimiento con eficacia. Si la máquina de nuestro ejemplo se queda sin género y está conectada puede avisar, así el proveedor puede ir a recargarla. Pero obviamente, hay muchos casos en los que esa necesidad de conexión no es tan importante como en otros. Este parece un buen ejemplo de uno de esos casos...

Las redes de comunicación son las verdaderas autopistas por las que circulan los programas. Una red no es un sistema pero La Red Global al completo forma un sistema gigantesco. Dada su enorme complejidad no son pocos los que aseguran que ya es capaz de pensar por sí misma. Las líneas de conexión forman túneles de luz eléctrica donde los flujos de datos se desplazan formando bandas de energía cegadora. El tránsito en los más concurridos es masivo y en muchos casos bastante divertido. Ten en cuenta que ese flujo es constante. Un programa no puede detenerse en ellas. Es

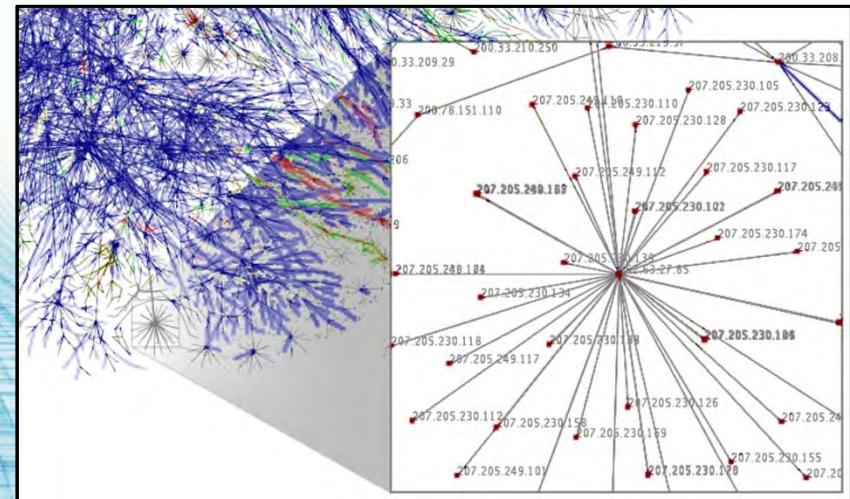
llevado por la corriente como un río tumultuoso, siendo arrastrado como por una fuerza de marea que lo conduce de sistema a sistema, de mundo en mundo por todo el universo digital. Tampoco es posible darse la vuelta. Si un programa va en una dirección y quiere regresar debe llegar primero al primer sistema que encuentre en el camino para volver en sentido contrario. En la mayoría de los casos esto no supone ningún problema. Pero hay sistemas que impiden la entrada y la salida sin permiso por lo que un programa que se encuentre con uno tendría que o bien darse la vuelta al llegar a él o apañárselas para salir de nuevo si éste le ha concedido el acceso. Como ves la red global tiene sus leyes, y esto no es más que el principio.

Hoy en día las redes inalámbricas están tan extendidas, son tan rápidas y se han vuelto tan sofisticadas que en la práctica es posible acceder a cualquier sistema que se encuentre conectado. La mayoría de dispositivos permite esa forma de conexión al disponer de los componentes necesarios pues ya vienen de serie. Esto incluye desde una cafetera o un reloj de pulsera hasta un ser humano ya que los microchips que posibilitan la conexión neuronal de un cerebro a las redes están a la orden del día. Pero las vías más seguras siempre han sido y al parecer lo seguirán siendo las líneas de cable. Hilos físicos que ya sea por medios ópticos o tradicionales conducen las señales; vías anchas donde el tráfico es tranquilo y seguro y con menos riesgo de producirse accidentes. Los sistemas de transmisión por ondas en los que se basan las redes inalámbricas son peligrosos en ocasiones por varias razones. Pueden desconectarse, sufrir interferencias o recibir ataques con facilidad.

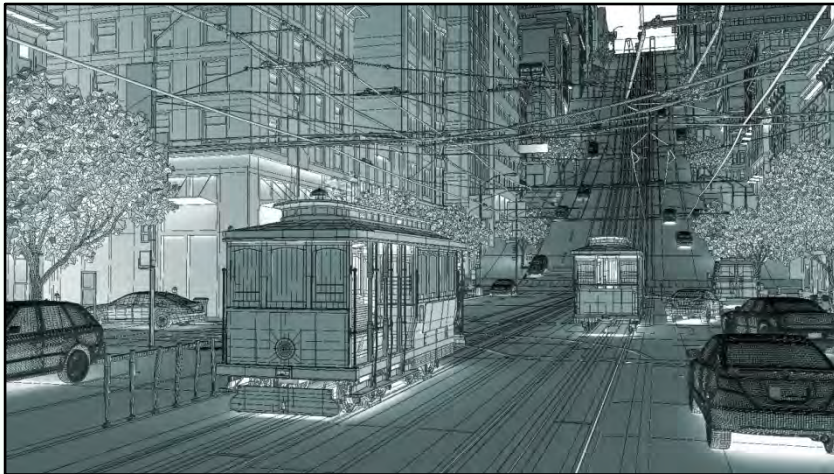
Cuando en la Realidad Básica se producen tormentas en su atmósfera la sobrecarga eléctrica de los rayos pueden cortar las vías de comunicación, destrozarse un sistema e incluso si ha habido muy mala suerte mutilar o corromper a un programa. Pero esto por suerte sucede muy rara vez. Puede suceder también en las líneas fijas de cable aunque suelen estar mucho mejor protegidas. Al fin y al cabo un rayo en la Realidad Básica es un pulso electromagnético de tanta potencia que es capaz de barrer en el campo electromagnético con toda entidad digital que se mueva por los alrededores. Y el peor enemigo del universo digital son precisamente los pulsos electromagnéticos de alta energía. Piensa por ejemplo que una explosión atómica o una tormenta solar de alta intensidad puede freír todos los

dispositivos en milisegundos y con ello llevarse por delante un fragmento o en el peor de los casos todo, repito, todo el universo digital al completo, que se desvanecería de un plumazo. De existir un fin del mundo para los programas este es sin duda uno de los peores.

Algunos sistemas pueden consistir en un único equipo de hardware, como un ordenador o un dispositivo electrónico de cierta potencia, pero los sistemas más sofisticados funcionan como el conjunto de un gran número de equipos conectados entre sí. Estos conjuntos se llaman **grupos de servidores** y forman los sistemas más potentes y completos. Configuraciones de hardware de gran potencia que permiten la ejecución de sistemas con entornos de gran complejidad como un mundo virtual abstracto, ultradetallado o un juego muy complejo por ejemplo. A su vez suelen servir de nexo de unión (o nodos) para muchos otros dispositivos, permitiendo que puedan conectarse a la Red Global como si fuese una estación central desde la cual es posible viajar a cualquier parte.



solicitudes la llevan a cabo secciones especializadas del sistema que se encargan de esa tarea, por lo que el personaje no tiene que preocuparse de identificar las partes del hardware que necesita. El mundo físico del hardware queda pues en una capa aparte, aislado e invisible para los jugadores, que no deben preocuparse de estos asuntos.



LA MEDIDA DEL TIEMPO EN EL SISTEMA

Muchos sistemas equiparan sus medidas del tiempo con las franjas horarias de las zonas del mundo físico en donde se encuentren ubicado su hardware. Dados los desfases horarios, es común ajustarlo a una medida o patrón estándar, como lo es el meridiano de Greenwich o Meridiano Cero en el mundo de la Tierra. Esto permite evitar conflictos con los relojes internos y confusión entre los usuarios, que son al fin y al cabo los que sufren sus consecuencias. Pero para evitar estos problemas como una solución hace ya tiempo que se adoptó un sistema para medir el tiempo en el universo digital. Aunque muchos usuarios lo desconocen, ha sido adoptado ampliamente por la mayoría de los sistemas y de los programas, que lo utilizan como un método para medir el tiempo.

Mediante este sistema una jornada completa del mundo digital se divide en **Tics**. Un Tic se divide en mil unidades, o más bien de 0 a 999 (ya que al llegar al final vuelve a saltar a 0). Cada unidad se divide a su vez en otras mil unidades. De nuevo de 0 a 999 fracciones de la unidad del Tic. Es posible volver a dividirlos una vez más, y otra..., y así hasta donde hiciese falta. Pero a efectos de este juego eso no es necesario. De esta forma un momento de la jornada en el sistema podría describirse como: Tic 456.322. Es decir, el momento 322 del Tic 456 (como si fuesen los minutos de una hora en el mundo físico).

Una jornada en el mundo digital mide un TIC, equivalente a mil unidades. Cada unidad del Tic puede dividirse a su vez en otras mil, y así sucesivamente hasta donde sea necesario. A efectos de juego es suficiente con una sola división tras la primera.

El Tic comienza en el momento de la hora cero, es decir a las doce de la noche del Meridiano de Greenwich del mundo de la Tierra. Si se tratase de otro mundo se tomaría en cuenta el patrón usado como equivalente. Llega a 999 en el último momento del día y se reinicia, volviendo a comenzar una vez más. Así, Tic a Tic, un tiempo único avanza en el universo digital para todos los programas, que ordenan sus vidas en función de sus latidos.

CARACTERÍSTICAS DE UN SISTEMA

En el momento de diseñar un sistema ¿cuáles son las características que hay que tener en cuenta? La respuesta es que en realidad no hace falta tener ninguna salvo el o los tipos de entorno que es posible encontrar en él. Con eso es suficiente. El entorno puede ser cualquiera de los que se han descrito en la primera sección de este capítulo. También conviene preguntarse si es posible encontrar a su vez varios subsistemas activos ya que en un sistema no tiene por qué haber sólo un entorno. Puede haber varios ejecutándose sobre uno principal e incluso dos o más combinados. Todo es posible.

Pero no te compliques, en un principio bastará con que decidas en qué entorno prefieres vivir tus aventuras. Más adelante ya habrá tiempo de enredar el juego un poco más.

Si deseas añadir algunos detalles más a tu sistema puedes anotar en alguna parte algunos rasgos que se describen a continuación:

FICHA DEL SISTEMA

- **Enfoque del escenario:** Lo primero y lo más importante como ya he dicho varias veces es decidir el tipo de escenario en el que se basa el sistema. Esquemático, mundo virtual, entorno de juego, etc. Es como llegar a un país en el mundo físico y describir qué es lo que puedes encontrar en él.

Si consideras que con esto es suficiente ya puedes plantarte y ponerte a jugar a Scroll. Si el sistema ejecuta directamente un mundo de juego que cumple con las funciones de un sistema no hace falta especificar nada más.

- **Tema:** En caso de que el entorno esté diseñado en torno a uno o varios temas específicos es conveniente anotarlo. Por ejemplo, si el sistema consiste en un mundo virtual detallado y en él se describe una simulación del imperio romano bastan unas pocas líneas para recordarlo.

- **Subsistemas activos.** En un sistema pueden mantenerse activos otros subsistemas; tantos como lo permita la potencia del hardware disponible (y tú estés dispuesto a tener presentes). Con los modernos sistemas de computación cuántica las posibilidades se han disparado en proporción geométrica, aunque por supuesto eso depende del nivel de complejidad del entorno. Quizás una compañía sea líder por mantener una sola simulación ultradetallada con millones de suscriptores activos o bien por ofrecer un catálogo con más de cien títulos de juegos. En muchos casos se trata de un mercado y en ese mundo es posible encontrar de todo.

"Primero estaban las computadores centrales (mainframes), cada una compartida por mucha gente. Ahora estamos en la era de la computación personal, persona y máquina mirándose con inquietud la una a la otra a través del escritorio. Después viene la computación ubicua o la edad de la tecnología tranquila, cuando la tecnología se desvanece en el fondo de nuestras vidas".

Mark David Weiser

Un subsistema es un sistema al fin y al cabo por lo que de ser necesario puede tener su propia ficha que lo describa; con su identificador, su tema en caso de tener alguno y otros detalles. Si depende del sistema principal muchas características las comparte con éste por lo que su descripción puede ser algo más simple.

- **Identificador o nombre.** No es obligatorio aunque sí bastante útil para organizarse. Un sistema siempre posee por defecto una numeración. Por ejemplo: 207.205.230.108 Que no es más que su dirección en la red. Pero un montón de números no resultan muy intuitivos la verdad. En lugar de eso puedes ponerle un nombre que lo describa —de querer ponerle alguno claro— e incluso una dirección en la red llamada “dominio” usando caracteres alfanuméricos (letras y números).

Podría tratarse de “Sistema Nexus” por ejemplo, o bien de “Otherworlds Systems”, “Sistemas NOVA” o del dominio “Novanetwork.com”. Incluso de “Virtual Worlds Lavondyss Networks”, ubicado en la red bajo la dirección “Lavondyss.net” Algo que lo identifique mínimamente y te recuerde a quién pertenece, qué es posible encontrar en él o cualquier otro detalle que te ayude a identificarlo.

- **Pautas de comportamiento.** Un sistema en realidad es un programa muy complejo por lo que al igual que uno puede tener también estilos y pautas de comportamiento. A efectos prácticos un sistema es una IA muy poderosa que piensa y toma decisiones como si fuese un o más bien “el dios”. Describir algunas pautas no es una característica obligatoria pero puede hacerlo mucho más interesante. Supone la diferencia entre percibirlo como otro sistema anónimo o convertirlo en algo inolvidable.

No es necesario realizar una descripción extensa aunque puedes hacerla si quieres. Esto no son más que consejos, no son normas en absoluto. Basta con especificar **algún adjetivo o alguna frase corta**

que lo describa. Y por cierto, pese a que suele ser la tendencia sus rasgos no tienen porqué ser malos necesariamente. Por ejemplo, un sistema “**dedicado**” puede ser propenso a tratar a los programas como si fuesen sus invitados, haciendo anuncios generales (como si fuese el sistema de megafonía dentro de un complejo) de algunas actividades. De esta forma no sería raro escuchar con una voz aterciopelada cosas como:

"En el TIC 234.456 se proyectará la película "TRON" en la sala de plata. Se ruega puntualidad y no olviden acudir con el mejor de sus avatares... Tengan un feliz día y recuerden..., el sistema siempre os protege."

Otros sistemas pueden ser “**paranoicos**” o incluso “**obsesivos**”, haciendo comprobaciones de las actividades de los programas con frecuencia. Algunos sistemas pueden ser “**poéticos**”, y agasajar a sus habitantes con constantes referencias a citas de famosos poetas entre los usuarios o inventar los suyos propios. Hay sistemas “**coléricos**” que gobiernan con puño de hierro, manteniendo a una plétora de líderes que mantienen el orden como si se tratase de una de esas dictaduras que existen en el mundo de los usuarios. Otros llegan incluso a ser “**flemáticos**”, “**cínicos**” o incluso dotados de “**un gran sentido del humor**”, por lo que no sería raro que cada dos por tres hiciese chascarrillos o bromas a costa de los programas, ya sea de forma general, a través de sus agentes o incluso bajo la forma de su propio avatar.

Porque, efectivamente, un sistema puede contar con un avatar con el que pueda mostrarse, aunque nunca actuará directamente a través de él. Siempre lo hará a través de sus agentes, sobre los que mantiene un control absoluto. El sistema se mantendrá siempre al margen como una aparición que viene y va cuando se le antoja.

Su avatar puede ser cualquiera que se te ocurra; puede aparecer como femenino o masculino, como una entidad fantasmagórica o como una criatura del mundo físico y, por supuesto, es capaz de intercambiar entre distintos aspectos siempre que quiera.

“En tiempos de Internet, las relaciones humanas se habían reducido a una fórmula muy sencilla: NEXT.”

Marina Castañeda

Al margen de todo esto, recuerda que un sistema suele estar por encima de todas las cosas por lo que siempre mantiene una actitud neutral respecto a lo que sucede en sus dominios. Son las anomalías las que le conducen a realizar acciones de ajuste. Por cada anomalía detectada emprenderá una acción para tratar de corregirla. Al llevarlas a cabo suele estar por encima de las emociones —existen casos de algunos sistemas muy emocionales en los que no—, pero a menudo sus respuestas y forma de actuar se verán condicionadas por su personalidad. Si debe cerrar o incluso borrar a un programa lo ordenará si tiene una buena razón para hacerlo, pero si posee una personalidad poética es posible que lo haga cantando algún romance que resulte apropiado, y si es cínico haciendo algún comentario ingenioso. Esto puede en algunas ocasiones volverse en su contra y un programa astuto puede invocar sus rasgos de personalidad para obtener algún beneficio como intentar provocarlo, engañarlo o vencerle de algo.

La personalidad de un sistema está para usarla, no para dejarla anotada y ya está. Como Director ponla en marcha y procura que tenga efectos sobre el entorno.

- **Clase del sistema.** De usar la clase de los programas puedes recurrir a la clase para establecer un nivel más alto de dificultad. No tienes que usarla al principio. Es más, lo conveniente es usarla cuando como Director dispongas de un poco más de experiencia. Especificar la clase no significa más que tener en cuenta qué dado más alto es posible poner en juego en las tiradas del sistema. Esto significa que no es necesario aplicarla a todos los dados que lanza una amenaza o a las tiradas de dificultad. Puede reflejarse en uno o más de los dados (o en todos por supuesto). Si un grupo de personajes se encuentra con varias amenazas a la vez por ejemplo, una de ellas, normalmente el líder, puede contar con una clase superior en algunos o en todos sus dados.
- **Limitaciones.** Algunos sistemas no disponen de la potencia suficiente para que un programa pueda funcionar con efectividad. Si

un programa accede a un equipo antiguo, a una chatarra con el disipador de la CPU echando humo o a un dispositivo de teléfono que a duras penas consigue correr su sistema el programa puede encontrar serias dificultades para realizar sus operaciones con eficacia. En el juego esto se refleja limitando la cantidad de puntos, es decir de dados, que puede usar el personaje. Son obstáculos inesperados que proponen nuevos desafíos y que si se buscan es con el objetivo crear situaciones divertidas.

Basta con que indiques el límite que posee el sistema en caso de tener alguno en **Operaciones**, **CPU** y **Memoria**. Un sistema con bajos recursos puede limitar las operaciones a 3, el acceso a la CPU a 4 ó 6 y la memoria a 4 por ejemplo. Si quieres complicarlo más puedes limitar otros factores como el uso de los comandos o lo que se te ocurra.

- **Amenazas u otros programas importantes.** En algunos sistemas a veces hay amenazas que vale la pena tener en cuenta. Por ejemplo, un mundo virtual abstracto podría estar gobernado por una IA que actúa como supremo líder a las órdenes del sistema, famoso por ser despiadado. Y este a su vez tener a un grupo de líderes bajo su mando. También es posible que como sistema de defensa en un mundo virtual detallado exista un programa tan poderoso que la sola visión de su avatar puede hacer que se le aflojen los bits al programa más curtido.

Como se explica en el apartado dedicado a los Personajes No Jugadores o PNJ'S, puede ser interesante diseñar algún personaje especial, un aliado con el que los personajes jugadores puedan contar en algún momento o bien describir con unas breves líneas a las amenazas más poderosas; y todo esto es conveniente anotarlo en alguna parte. Se trata de un registro de seres digitales que resultan clave, como los que es posible encontrar en cualquier otro juego de rol ni más ni menos.

- **Vulnerabilidades.** ¿Posee el sistema puertas traseras, códigos de desactivación, comandos específicos, rutinas de autodestrucción,

secretos ocultos? Todo sistema tiene vulnerabilidades por mucho que presuman los operadores del sistema de que no se ha detectado ninguna. La realidad es que a esa frase se le suele omitir la coletilla: "...por el momento".

Un sistema es mucho más interesante si tiene defectos y esconde algún secreto. Cuantos más mejor, aunque sin pasarse... Por ejemplo, un sistema podría mantener un estricto registro de entrada en el que para acceder sea necesario cumplir unos requisitos, conocer una clave, sobrepasar un obstáculo o superar una prueba. Pero si un personaje conoce alguna entrada secreta las posibilidades de vivir una aventura memorable solo acaban de comenzar.

- **Leyes del sistema.** Cada sistema tiene sus propias leyes, o al menos un conjunto de normas que marcan una diferencia. ¿Es posible entrar o salir del sistema sin autorización? ¿Está permitido que un programa pueda realizar copias de sí mismo? ¿Están obligados todos los programas a llevar el mismo tipo de avatar?

No es posible señalar todas las leyes que podrían existir pero sí algunas de las más importantes o con un efecto más relevante en la vida de los programas.

- **Otros detalles.** Por último pero no menos importante, queda anotar todo aquello que resulte de relevancia. ¿Quién gobierna en el sistema? ¿Está este controlado en realidad por un operador cabezota, ególatra o algo sádico? ¿Habita la consciencia de su creador en él tras haber sido digitalizado? ¿El mundo virtual en el que se encuentran los programas ha sido diseñado en realidad para lavarle el cerebro y controlar a los usuarios? Muchas preguntas y muchos más detalles que conviene recordar. Estas son las notas de las que brotan las semillas de aventura.



EJEMPLO DE UNA FICHA DE SISTEMA

Enfoque del escenario: El sistema consiste en un mundo virtual abstracto de diseño abierto. Una extraña versión del mundo de los usuarios aunque con un estilo gráfico muy característico.

Tema: Diseños futuristas de aspecto pulcro, metálico y brillante con luces y líneas de luz sobre grandes espacios abiertos. El tema del escenario es un entorno inspirado en la película "TRON".

Subsistemas activos: Ninguno. En el mundo existe una rejilla de juegos integrada basada en su mismo diseño aunque no funciona como un subsistema independiente. Su función es servir como un coliseo para los programas que infringen las normas.

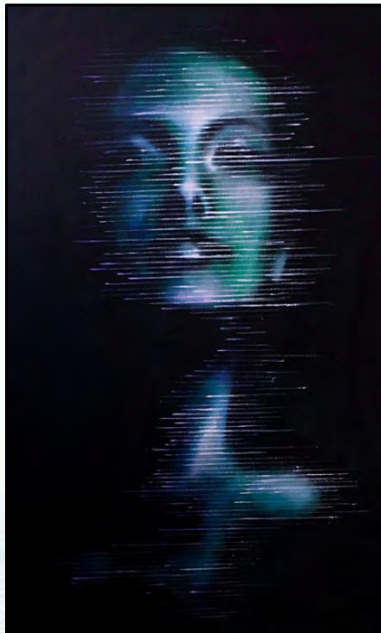
Identificador: El entorno pertenece a la compañía Sistemas NOVA y recibe el nombre de NOVA NETWORK SYSTEMS, con el dominio en la red: "novanetwork.com"

Personalidad: El sistema central, llamado simplemente "Central" es una personalidad ególatra que se considera el único dios de su mundo. Esta personalidad lo lleva a ser paranoico y extremadamente peligroso. No duda en castigar a cualquier programa que se le opone. Al usuario trata de engañarlo en todo momento, ocultándole sus propósitos mientras busca el modo de evitar sus intromisiones de una vez por todas. La personalidad de "Central" se puede usar en su contra pues a menudo se pone tan furioso que comete errores tácticos. Por otra parte, su paranoia puede usarse

en beneficio de los programas, provocando por ejemplo que malgaste sus recursos en cosas irrelevantes o desviando su atención.

Clase del sistema: La clase máxima que es posible encontrar es la VIII. Algunas amenazas, las más poderosas, disponen de algunos dados D8 al hacer sus tiradas.

Limitaciones: Ninguna. El sistema dispone de capacidad suficiente para que un programa pueda aprovechar todos sus recursos.



Amenazas y programas importantes: Central cuenta con subordinados que controlan a la población. Su primer agente de confianza, conocido bajo el identificador "CLU", actúa como gobernador del mundo digital. CLU vela porque siempre se cumplan los deseos de "Central", actuando con eficacia cada vez que se recibe una alerta de anomalía. Respecto a otras amenazas, el supremo gobernador cuenta con tres lugartenientes que son completamente leales a la causa. A su vez dispone de un batallón de agentes. De todos sus recursos, las más efectivas son con diferencia las unidades denominadas "Reconocedores". Un Elemento controlado por dos agentes que viajan en su interior. Su función es patrullar en todo momento las áreas del mundo con el fin de mantener controlada a la población.

Vulnerabilidades: La entrada al sistema supone la inmediata detención de los intrusos, que son obligados a combatir en la rejilla de juegos. Pero existe una puerta trasera cuyo secreto sólo conoce un operador humano en el mundo físico que hace poco ha descubierto los auténticos propósitos del sistema central. Basta una clave para permitir a un programa acceder al sistema.

Leyes más importantes:

- El sistema prohíbe terminantemente la realización de copias de un programa.
- Está prohibido circular fuera de las colmenas destinadas al software en el horario que se indica en las pantallas de información.
- Todo programa que fuese detectado sin identificación será condenado a la rejilla de juegos.
- Cualquier opinión en contra de "Central" se pena con el envío a la rejilla de juegos.
- Toda manifestación en contra de las fuerzas del orden se condena con el envío a la rejilla de juegos... etc.

Otros detalles: CLU en realidad desea derrocar a Central y tomar el control. Ha sido creado para detectar anomalías, pero no para ir en contra de los deseos del usuario y del creador. Sólo ahora comienza a sospechar la verdad. El usuario que una vez creó el mundo lleva ya algún tiempo viviendo entre ellos después de haber digitalizado toda su estructura molecular para reconstruirse como una criatura digital. Desde el interior, intenta reparar los daños que según él, considera que ha sido provocado por su propia negligencia y por lo que no cesa de culparse.

DESAFÍOS DEL SISTEMA

“Creo que los seres humanos como especie definen su realidad a través de la desdicha y el sufrimiento. Así que el mundo perfecto era un sueño del que su cerebro primitivo se trataba de despertar constantemente.”

Agente Smith. “Matrix”

En una narración, todo cuanto se opone a los objetivos de un personaje proviene o está relacionado con el escenario. Es más, el personaje también forma parte de él. En Scroll ese escenario no es otro que **el Sistema**.



En el capítulo donde se estudia la estructura de las pruebas ya he explicado las reglas básicas que rigen la resolución de los conflictos. Los jugadores se enfrentan a la adversidad en forma de una serie de desafíos que hay que superar. La dificultad de los desafíos se interpreta en el juego como una cantidad de dados. Su número varía entre uno y dieciséis.

Resolver las tiradas y obtener la victoria supone la superación de esos desafíos y por lo tanto de la adversidad, facilitando que el personaje obtenga

sus metas. Los resultados del fracaso dependen del contexto de la situación planteada, pero siempre suponen un freno a la obtención de las mismas.

Cómo se solucionan las tiradas ya se ha descrito y no lo voy a repetir aquí. No obstante si conviene tener en cuenta las dos formas de resolver las acciones que se plantean en el juego. El enfoque **absoluto** y el **relativo**. El juego siempre te ofrece la posibilidad de elegir el que más te convenga para una situación, siendo posible alternar entre ambos sistemas de una escena a otra **¡e incluso combinarlos!**

Por lo tanto, antes de seguir leyendo este apartado y a modo de resumen, recuerda lo siguiente:

- En el **enfoque absoluto** la adversidad se soluciona como **un conjunto** por lo que distintos obstáculos y amenazas funcionan como un único desafío que los engloba a todos. Esto se traduce en una cantidad de dados establecida por el Director. La diferencia de éxitos obtenidos determina qué bando se impone sobre el otro y su cantidad cómo y en qué grado lo hace.
- En el **enfoque relativo** se recurre al sistema clásico de muchos otros juegos de rol. Cada obstáculo o amenaza se supera por separado, teniendo cada una un valor que expresa su dificultad. Para resolver las acciones lo normal es recurrir a una sucesión de asaltos hasta que se declare un vencedor o se resuelva el conflicto.

A continuación se describen algunos de los desafíos que un sistema puede plantear a los jugadores y cómo se aplican las reglas en ambos enfoques en caso de ser necesario.

Todos los desafíos pueden ordenarse en dos grandes grupos: **obstáculos** y **amenazas**. Para establecer el valor de dificultad se cuenta con la tabla de dificultad como guía, el criterio del Director (y de su sentido común), que lo decide en función de lo desafiante que quiere que resulte el conflicto.



OBSTÁCULOS

“El homo sapiens tiende al reconocimiento de pautas. Que es a la vez un don y una trampa.”

William Gibson



Un obstáculo es cualquier situación que se opone a que un personaje pueda cumplir sus objetivos. Una descripción que se deja abierta a muchas interpretaciones, pero si fuese posible describir todos los obstáculos que pueden darse en este juego no merecería llamarse juego de rol. Su potencial como juego proviene precisamente de su enorme diversidad.

En Scroll siempre me refiero a un obstáculo cuando una situación no amenaza de forma directa a un personaje sino que dificulta sus acciones. Puede tratarse de un código de seguridad que impide acceder a un bloque del sistema, a los sistemas de protección que vigilan un paquete de información vital, a la fuerza de un muro que impide acceder a una fortaleza de datos, a la dificultad que supone escalar un rascacielos, a la suspicacia de un guardia armado que hace muy difícil engañarle, a la altura que dificulta saltar una verja o a los vehículos que bloqueando la carretera impiden el paso al conductor de una moto de luz.

La dificultad de un obstáculo se describe con una cantidad de dados o con un número de éxitos que hay que igualar o superar. La tabla de dificultades que algunos capítulos más atrás se ofrece como guía es muy útil para hacerse una idea de la relación de dados. Algo que depende de si se trata de una prueba activa o pasiva.

Y ya que en este capítulo la tendré en cuenta, la reproduzco aquí de nuevo para tu comodidad. Es exactamente la misma tabla.

Dificultad	Nivel (activa)	Éxitos (pasiva)
Fácil	Entre 1 y 2 dados.	1
Moderado	Entre 3 y 4 dados.	2
Desafiante	Entre 5 y 8 dados.	3 - 4
Difícil	Entre 9 y 12 dados.	5 - 6
Muy difícil	Entre 13 y 16 dados.	7 - 8

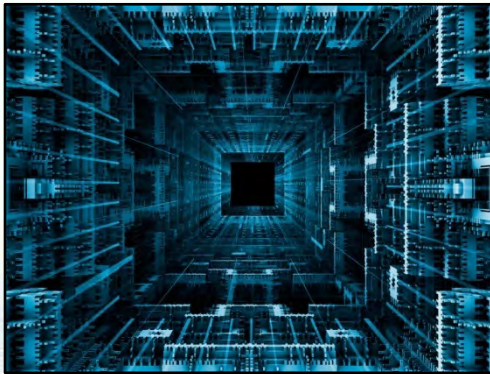
RESOLVIENDO LAS ACCIONES

- Si la situación se resuelve mediante un **enfoque absoluto**, en caso de haber varios obstáculos en una misma escena es posible agruparlos en un solo conjunto para resolverlos todos a la vez. Las reglas para hacer la reserva son muy flexibles. Tú como Director puedes decidir el nivel de dificultad que describa el nivel del desafío como mejor te parezca. En caso de duda, una buena manera de ajustarlo es tener en cuenta el nivel del obstáculo más difícil. A su cantidad de dados añade otro por cada obstáculo añadido que haya en la escena. Si los obstáculos adicionales son tan duros como el primero, añade dos dados por cada uno a la reserva en vez de uno. Este sistema debería ayudarte a resolver la mayoría de situaciones. Si piensas que se trata de algo desafiante de verdad simplemente suma el valor de todas las dificultades por separado.
- En el caso de resolverlo mediante un **enfoque relativo** cada obstáculo se resuelve por separado. Cada uno con su nivel de dificultad.

Mediante estos sistemas no necesitas de mucho más. Pero por sus características, existen algunos obstáculos cuyo proceso de superación pueden añadir un componente narrativo interesante a la partida. Por eso existe en el juego el concepto de **Dureza** y **Resistencia**. Una herramienta que sirve para especificar lo difícil que resulta superar algunos obstáculos y cuanto tiempo y energía es necesario invertir en el proceso. De esta forma, superar el obstáculo se convierte en otra forma de llevar a cabo un enfrentamiento.

DUREZA Y RESISTENCIA

Hay algunos obstáculos que para superarlos es necesario irlos debilitando antes. Reflejar esto en el juego permite añadir situaciones de tensión que pueden mejorar la experiencia de juego. Para resolver este tipo de pruebas se recurre al **reto por acumulación** —explicado en el capítulo destinado a la



resolución de las pruebas—. Consiste en superar un valor de dificultad teniendo la posibilidad de acumular los éxitos obtenidos en la tirada hasta alcanzar un valor determinado. Cuando se alcanza ese valor la prueba se considera superada.

Es el caso por ejemplo de un programa que intentase abrirse paso a través del muro

de datos alrededor de un complejo. Sus defensas son elevadas por lo que el programa necesita de algo de tiempo para realizar la tarea. Una situación tensa si por los alrededores existen guardias armados o programas dedicados a la vigilancia, y el tiempo es un factor clave.

- **La Dureza** expresa la dificultad de un obstáculo. Un valor que es necesario superar con una tirada. Puede tratarse de algo muy duro, fuertemente blindado o muy protegido. La tendencia es pensar que la Dureza se trata de algo sólido pero puede ser también la dificultad de

engañar o convencer a un guardia, de reventar un código o de diseñar un programa.

- **La Resistencia** en cambio se refiere a la cantidad de éxitos que es necesario acumular para considerar superado el reto. Simboliza el aguante que posee el obstáculo, su fuerza estructural. En caso de tratarse de una criatura su salud e incluso su paciencia.

Superar el obstáculo implica llevar a cabo un reto en el que hay que sobrepasar el valor de dificultad indicado por la Dureza. La diferencia de éxitos obtenidos, es decir los éxitos que sobrepasen al valor de Dureza, se acumula en los asaltos siguientes hasta que se tengan tantos como indica la Resistencia. Cuando esto sucede se considera superado el reto. Como es lógico, es necesario superar el valor de la Dureza pues un empate implica no contar con ningún éxito que acumular.

Como ves, el nombre de ambos conceptos no es más que una conveniencia. Ambos términos podrían ser perfectamente “Blindaje” y “Estructura”; “Armadura” y “Aguante” e incluso “Frialdad” y “Paciencia”. A la mente le gusta sentir que los sucesos son plausibles por lo que trata de mantener ambos conceptos a un nivel lo más abstracto posible para poder adaptarlos mucho mejor al contexto de la situación.

El valor de Dureza suele ser pasivo, el de Resistencia siempre lo es. Es decir, que en el caso de la Dureza se trata de una cantidad de éxitos fijos a superar o bien de una cantidad de dados. Para transformar el valor de Dureza entre un número de éxitos fijo y un número de dados ten en cuenta la relación que se expresa en la tabla. La fórmula general es que cada éxito fijo se sustituye por dos dados. En el caso de la Resistencia siempre se trata de reunir una cantidad fija de éxitos.

El valor de Dureza también puede ser negativo. Cada unidad negativa añade automáticamente a la tirada un éxito. Por lo que si se trata de una dureza -2 se añadirían a la tirada dos éxitos aún cuando en ésta no se hubiese obtenido ninguno. Esto expresa una debilidad o la facilidad extrema para superar el obstáculo. También puedes sustituir esa cantidad de éxitos en dados

de ventaja que obtiene quien realiza la prueba. Siempre teniendo en cuenta la ley de que por cada éxito se obtienen dos dados.

La cantidad de éxitos para superar la Dureza y la Resistencia deben ser los suficientes para que en el juego se refleje lo que cuesta superar la prueba y, en el caso de que sea relevante, el tiempo que puede necesitarse en superarla. Algo que sucede con frecuencia en una escena donde el ambiente está plagado de enemigos. Los valores de Dureza debe decidirlos el Director guiándose por la tabla de dificultades. Ambas se anotan mediante la expresión **D(R)** (o bien D/R, como más te guste). Donde “D” puede ser negativo o positivo.

He aquí algunos ejemplos:

- **5(8)** Es necesario obtener 5 éxitos para superar el valor de Dureza. Si se obtiene más éxitos que esa cantidad pueden acumularse. Los asaltos siguientes se siguen realizando pruebas. Cuando se acumulan 8 éxitos la prueba es superada.
- **0(8)** No tiene valor de Dureza por lo que cualquier éxito obtenido se acumula. Cuando se acumulan 8 éxitos la prueba es superada.
- **-2(5)** El objeto es tan endeble o débil que aunque en la tirada no se obtenga ningún éxito automáticamente se añaden dos éxitos. Cuando se acumulen 5 éxitos se supera la prueba.



AMENAZAS

“El mayor enemigo del ordenador es el hombre”

Anónimo



Una amenaza es cualquier adversidad que de forma directa o indirecta trata de provocar un perjuicio a los personajes. Da igual si pretende dificultar sus acciones, neutralizarlos, humillarlos, detenerlos, capturarlos o dañar su integridad descompilando sus bits y esparciéndolos por ahí. Siempre que exista la certeza de un peligro de cualquier tipo éste se considera una amenaza. Y como tal, se puede describir. Esta definición una vez más deja el término abierto a muchas interpretaciones, pero lo cierto es que esto no tiene mucha importancia. Una amenaza puede ser muchas cosas. Puede ser por ejemplo un agente del sistema, un virus, las acciones de un usuario o de cualquier otro programa. Pero también puede tratarse de un área peligrosa del Sistema; de un Elemento con efectos adversos sobre el personaje o de una trampa; del riesgo de un mal funcionamiento en las líneas de datos por las que viaje el personaje o del peligro que supone una descarga eléctrica sobre un soporte de hardware en el mundo físico.

Una amenaza puede suponer un peligro **activo** o **pasivo** para un personaje:

- Una **amenaza activa** tiene la intención de provocar un perjuicio en el personaje, como podría ser un agente o un virus informático por ejemplo.

- Una **amenaza pasiva** en cambio constituye un peligro, pero la amenaza no tiene una intención —al menos consciente— de provocar un perjuicio; aunque sí es posible que haya sido diseñada para ello. De hacerlo puede haberse producido por accidente, omisión, negligencia, porque es lo que pretendía algo o alguien o simplemente debido a las circunstancias. Ejemplos de amenazas pasivas son las trampas, las áreas peligrosas capaces de provocar daño o secciones del Sistema que debido a un comportamiento anómalo puedan tener consecuencias adversas para los personajes.

En el juego cualquier amenaza supone un reto para los personajes. Su nivel de desafío se expresa mediante una cantidad de dados que oscila entre **dos y dieciséis**. Por regla general es cuanto necesitas para ponerla en juego. Pero una amenaza es mucho más que un montón de dados. Cada una cuenta con unas características —o **rasgos**— que se detallan en su descripción y sus efectos se reflejan en la partida. Constituyen una versión simplificada de una ficha de personaje que detalla sus fortalezas y debilidades. Esta simplificación facilita el trabajo del Director, permite que el desarrollo de las escenas resulte mucho más fluido y centra la acción donde más importa, en los personajes de los jugadores.

Cuando una amenaza posee una habilidad esta funciona como si fuese una Rutina. Si se trata de una capacidad especial o un talento único entonces lo hace como una Utilidad. Éstas intervienen en la partida de una forma mucho más sencilla. Pueden hacerlo a nivel narrativo o, si se dan las circunstancias, a nivel de mecánicas de juego concediendo ventajas (dados o éxitos) cuando se utilizan.

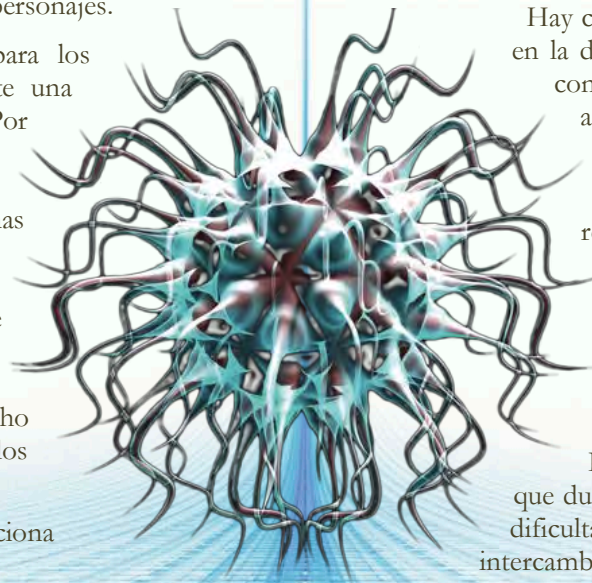
Cuando los rasgos intervienen a nivel narrativo no tienen un efecto sobre la mecánica de las pruebas, pero sus características pueden ser determinantes para la trama. Por ejemplo, un agente del Sistema que tenga la capacidad de aparecer en cualquier posición del mundo virtual “apropiándose” del avatar de

otro programa no necesita de mucho más. Ni especificar ventajas, ni bonificadores, ni nada..., el uso de esa capacidad ya supone un buen problema para los personajes. Pero si el contexto de la escena posibilita que una habilidad o una capacidad especial entre en juego la amenaza obtiene ventaja **añadiendo un dado a su nivel de desafío** y por lo tanto a su reserva. Por ejemplo, si una amenaza posee la habilidad de “Correr” obtiene ventaja cuando lo hace, por lo que puede añadir un dado a su reserva para hacer las pruebas.

Hay casos en los que las mecánicas se aplican de otra forma. Si en la descripción de un virus uno de sus rasgos especifica que con un ataque con éxito provoca que la víctima comience a actuar de forma errática, ésta debe tenerlo en cuenta al realizar las pruebas, y si fracasa actuar en consecuencia.

En ocasiones una amenaza puede ser muy eficaz realizando algunas acciones. En esos casos obtiene hasta doble ventaja (dos dados) si en la descripción de su habilidad se especifica que puede darse esa posibilidad. Para indicarlo puedes usar un asterisco “*” junto a su nombre o un “(2)” que te ayude a recordar que obtiene doble ventaja si ésta entra en juego. Si por ejemplo en la descripción de los rasgos se señala un asterisco junto a la Rutina “Correr *” (o bien se indica “Correr (2)”), cada vez que durante la partida la amenaza ejecute esta acción su nivel de dificultad se incrementa dos dados. Estos dos dados se pueden intercambiar si se desea por un éxito extra que se añade a los resultados de la prueba en lugar de elevar los dados de la reserva. Algo que resulta útil si se está resolviendo una situación mediante una prueba pasiva, si se desea asegurar al menos un éxito o si añadir más dados resulta engorroso.

No todo tiene porqué ser tan malo. Algunas amenazas poseen también vulnerabilidades que provocan el efecto contrario. En algunos casos ésta se opone a los intereses de la amenaza por lo que pueden sufrir desventajas en uno o dos grados utilizando el mismo método. En otros pueden ser los personajes los que obtengan esos mismos grados de ventaja al realizar las



pruebas. De igual modo, si los detalles de una vulnerabilidad no son aplicables a nivel de mecánicas de juego casi siempre lo siguen siendo a nivel narrativo. Por ejemplo, es posible que una amenaza simplemente huya nada más ver a un agente del Sistema porque las teme; que se rinda automáticamente ante un personaje de una Clase determinada; que una Utilidad especial sea capaz de destruirlo automáticamente o que un agente de nivel medio huya nada más encontrarse con un personaje de cierta reputación. Como sucede por ejemplo con los agentes anónimos al ver a Neo “desatado” en la película Matrix.

NIVELES DE DESAFÍO

Todas las amenazas describen su **nivel de desafío** mediante una cantidad de dados. Y del mismo modo que con los obstáculos, las reglas para aplicar los niveles varían dependiendo del enfoque que elijas para resolver las pruebas.

ENFOQUE ABSOLUTO

En el enfoque absoluto si en una misma escena varias amenazas actúan a la vez se agrupan todos sus niveles de desafío. Las reglas para hacerlo **dependen de cómo se coordinen las amenazas entre ellas**.

- Si varias amenazas actuando en grupo lo hacen descoordinadas, ya sea porque no se conocen o porque no son eficientes, entonces se toma el nivel de la amenaza más grave o del oponente más poderoso (su líder por ejemplo) y se añade otro dado por cada amenaza u oponente adicional. Ten en cuenta que si sus rasgos entran en juego sus niveles de desafío pueden aumentar. Si en cambio los oponentes adicionales son tan poderosos como su líder se añade dos dados en vez de uno. Por ejemplo, si tres amenazas de nivel 2 actúan en grupo de este modo se toma el nivel de una de ellas y se añade 1 dado por las otras dos. Por lo que el nivel total es 4.
- Si un grupo de amenazas es capaz de coordinarse medianamente bien puedes añadir uno o dos dados más a la reserva del grupo reflejando ese incremento de eficacia. Por ejemplo, si las tres amenazas de nivel

2 del ejemplo anterior son capaces de actuar algo mejor su nivel total puede ser 5.

- Si varias amenazas actuando en grupo se coordinan bien entre ellas el nivel de desafío del grupo es la suma de sus niveles individuales. Esto, evidentemente, dispara la dificultad del encuentro pero esa peligrosidad es lo que se espera de un grupo eficaz.

*A medida que los oponentes son derrotados se van eliminando dados lo que refleja cómo disminuye la cantidad de enemigos en el encuentro.

ENFOQUE RELATIVO

El enfoque relativo siempre es tan sencillo como tener en cuenta el nivel de desafío de cada amenaza y resolverlas por separado. No obstante las amenazas pueden coordinarse bien trabajando en grupo lo que en algunas situaciones les otorga ventajas. Éstas pueden entrar en juego a nivel narrativo o traducirse en dados extra o éxitos que se añaden a los resultados de las pruebas. Recuerda que si alguno de sus rasgos entra en juego sus niveles de desafío pueden aumentar en consecuencia.



[illegible]

Si quieres añadir más información, seguir el mismo orden de la lista de apartados es una buena pista para saber su grado de importancia. Aunque si para describir una amenaza alguno te parece más relevante que otro tú tienes la última palabra. Esta descripción también es válida para describir a otros programas que no entren en la categoría de amenazas, como podrían ser los aliados, algunos personajes clave y otros seres digitales que pueblan tu Sistema. Este tipo de personajes entran en la categoría de PNJ (personajes no jugadores). Más adelante añadiré algunos detalles más sobre ellos.

- Describir el tipo de programa, su función y/o cuáles son sus principales objetivos permite averiguar de qué tipo de amenaza se trata.

3. **NIVEL DE DESAFÍO:** El número de dados que expresa su nivel.
Un valor entre 1 y 16.

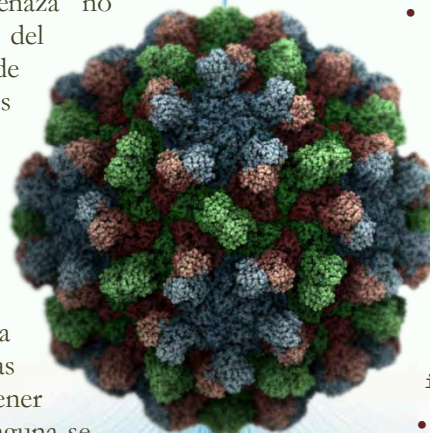
- La Clase es muy eficaz para reflejar la eficacia de cualquier programa, no sólo de una amenaza. Si se trata de un Usuario, recurrir a la Clase traslada al nivel de las mecánicas del juego el riesgo que supone para un programa la impredecibilidad de su naturaleza y sus conocimientos.

5. **DESCRIPCIÓN:** Describir a la amenaza permite saber contra qué se enfrentan los personajes, los prepara para la situación y despierta su imaginario. Puede ser tan breve como una frase o tan extensa como se desee, aunque lo recomendable es resumirlo lo mejor posible.
6. **EFFECTOS:** Si las acciones de una amenaza —o de cualquier programa— producen unos efectos se describen en este apartado.
7. **COMPORTAMIENTO:** Es raro que una amenaza no muestre un comportamiento. En este sentido sucede del mismo modo que con los rasgos de comportamiento de un Sistema. Puede ser tan escueta como uno o dos adjetivos. Por ejemplo, una amenaza puede ser histriónica, desconfiada, paranoica, bromista, cínica, colérica... Unos pocos detalles dotan de vida a la amenaza y ayuda a que los jugadores no sólo la visualicen sino que en la recuerden mucho mejor.
8. **RUTINAS:** Se describen las habilidades que posee la amenaza. Si está muy especializado en alguna de ellas se señala mediante un asterisco “*” o un “(2)”, para tener en cuenta que obtiene doble ventaja. De no tener ninguna se omite el apartado o se indica “ninguna”.
9. **UTILIDADES:** Se describen aquellos talentos o capacidades que reflejan las funciones especiales de una amenaza. Si está especializado en alguna de ellas se señala mediante un asterisco “*” o un “(2)”, para tener en cuenta que obtiene doble ventaja. De no tener ninguna se omite el apartado o se indica “ninguna”.
10. **VULNERABILIDADES:** Dedicar un poco de tiempo a describir algunas debilidades siempre es una buena idea. Nunca se sabe cuándo pueden resultar útiles. Los jugadores pueden sacar mucho partido de ellas por lo que incluirlas ayuda a la narración, mueve la trama y crea

situaciones memorables. Resolver una escena o incluso la trama completa de una aventura poniendo en juego de forma ingeniosa algún punto débil de una amenaza será siempre digno de recordar y merece ser premiado como se merece.

EJEMPLO

- **TIPO Y FUNCIÓN:** Agente. Programa vigilante del Sistema.
- **NOMBRE O IDENTIFICADOR:** "Vigilante"; "Agente".
- **NIVEL DE DESAFÍO:** 6
- **CLASE:** VI
- **DESCRIPCIÓN:** Vigilante incansable en un entorno virtual ultradetallado. Vela por que se cumplan las leyes de su sistema.
- **EFFECTOS:** Patrullan el mundo virtual del sistema buscando anomalías. Si la anomalía es provocada por un personaje tratan de corregirla. Si no es posible la eliminan y la retiran.
- **COMPORTAMIENTO:** Fríos, arrogantes, infatigables. Muestran un desprecio absoluto por los humanos. No conocen la piedad, no poseen empatía y jamás dudan un instante.
- **RUTINAS:** Correr, saltar*, disparar arma a distancia, lucha cuerpo a cuerpo*.
- **UTILIDADES:** Esquivar balas, sustitución* (teletransporte).
- **VULNERABILIDADES:** Se atienen a las leyes de su sistema. Su IA es limitada por lo que jamás se salen de sus pautas de comportamiento, lo que los hace muy previsibles. Si se enfrentan a algo nuevo que no logran asimilar se activa inmediatamente su Función de salida (huyen).



AMENAZAS ACTIVAS



Cualquier entidad que busque de forma activa provocar un daño entra en esta categoría. Todos los programas hostiles por ejemplo se consideran amenazas activas.

Pero una amenaza no tiene porqué ser siempre un ser digital. Puede tratarse de un usuario o cualquier otra criatura del mundo físico tratando de destruir al personaje, al sistema o el universo digital al completo incluyendo el suyo. Una amenaza activa es también una forma de llamar a un personaje no jugador (PNJ) que sea hostil o suponga un peligro para los personajes. Siguen siendo personajes controlados por el director, pero en calidad de amenazas.

La definición de las amenazas se organiza casi siempre en torno al tema del escenario. Pero dado que se puede entrar y salir de muchos sistemas es posible encontrar en ellos amenazas de muchas clases. El tema asociado puede tratarse de algo visual, como un grupo de agentes con chaqueta y corbata; programas con aspecto de animales feroces del mundo físico, de las herramientas que es posible encontrar en un quirófano o los enemigos que han sido diseñados para un juego concreto por ejemplo. En otros escenarios ese tema puede consistir en algo mucho más conceptual como la recreación de una pesadilla humana, una IA que exprese una emoción o un virus que provoque sobre los programas los mismos efectos de las enfermedades que

padecen los usuarios. No hay límite para la creatividad del usuario a la hora de programar seres digitales al mismo tiempo que la propia evolución de los programas en el mundo digital causa la aparición de criaturas que nadie habría imaginado jamás.

Si un programa que entra en el grupo de las amenazas tiene un papel clave o se considera importante en la trama, es posible prescindir del esquema que describe a las amenazas y hacerle su propia ficha de personaje. En muchos casos vale la pena si consideras que la amenaza lo merece. Es posible entonces aplicar las mismas reglas para ambos, usando reservas de Operaciones, CPU y Memoria para la amenaza. Hacerlo o no depende de los gustos, del trabajo que se esté dispuesto a invertir y, por supuesto, de si ese esfuerzo realmente merece la pena. Ten en cuenta que de usar una ficha de personaje completa para la amenaza, para que el Director de juego pueda ganar puntos de Anomalía en este caso sólo cuenta que dominen sus reservas de Operaciones. Nunca de CPU o Memoria. Por otra parte, es posible que en algunos casos muy concretos, ya sea por razones de la trama o porque la idea ha surgido durante la partida, la amenaza pueda llegar a ponerse del lado de los personajes como un aliado, aunque sea temporal. Entonces también se podría considerar la posibilidad de convertirlo en un personaje no jugador (PNJ), y tal y como se explica en el apartado dedicado a los PNJ, llegar a hacerle también su propia ficha de personaje. Por último el que disponga de una ficha permite utilizar sobre ellos algunos comandos que sólo se pueden aplicar sobre personajes con ficha o PNJS.

A continuación se describen las amenazas según su grado de importancia para a continuación detallar algunas de las más comunes que es posible encontrar en un sistema.

TIPOS DE AMENAZAS

Puesto que las amenazas no son iguales unas a otras, se clasifican según su grado de importancia. Esto es muy importante para organizar los encuentros que pueden surgir en una aventura y para catalogarlas debidamente. Las pautas que se muestran a continuación te serán de gran ayuda para hacerlo.

AMENAZAS DÉBILES Y ESBIROS

Las amenazas más comunes no tienen un alto nivel de poder y a nivel individual no son muy peligrosos. El problema es cuando se agrupan, algo a lo que tienen tendencia pues ya se sabe que la unión hace la fuerza. Se trata de programas muy sencillos con una programación orientada a realizar las funciones más básicas. En ese sentido están al mismo nivel que la media de la mayoría de programas que es posible encontrar en cualquier sistema. En este grupo encontramos a los programas diseñados para combatir como soldados, programas mensajeros, a los trabajadores del sistema. Es decir, todos aquellos cuya función es llevar a cabo unas pocas tareas rutinarias o ejercer de asistente de otros programas. Si se trata del mundo físico esta categoría engloba a las criaturas más comunes. El nivel de la población media se corresponde con el de una amenaza débil. Se trata de los “extras” o “la carne de cañón”. Un recurso que en muchas narraciones está ahí para cumplir con una función sencilla, aunque ésta sólo sea caerse redondo al primer puñetazo.



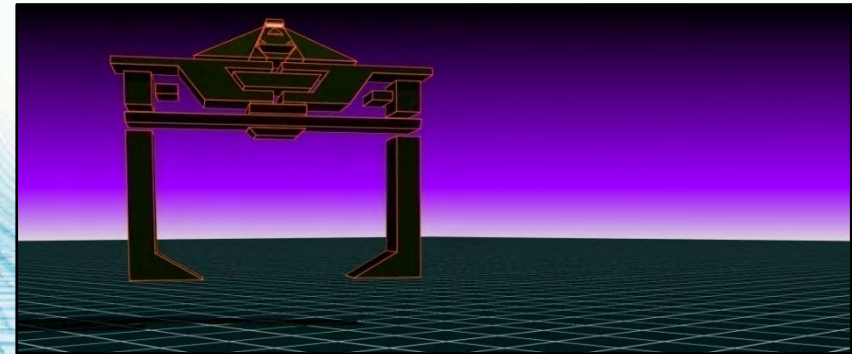
Las amenazas débiles y los esbirros de la primera categoría tienen un nivel de desafío de **entre dos y cuatro dados**. Dependiendo de cómo se coordinen entre ellos, el que actúe como líder —mediante el enfoque absoluto— podrá añadir más o menos dados a su reserva. Lo normal es que su nivel de coordinación actuando en equipo no sea muy bueno pero la excepción es la que confirma la regla. Una amenaza de **un único dado** se reserva para seres inofensivos o de muy poco poder como puede ser una mascota, ya sea virtual o real, o cualquier otro tipo de criatura que en el caso de surgir un enfrentamiento no suponga más que un contratiempo para los personajes (NdE: ¡Quita chuchol!).

Las amenazas débiles tienen entre una o dos habilidades especiales que funcionan del mismo modo que lo hacen las Rutinas. No suelen estar muy avanzados en ellas, o al menos tanto como para que suponga una diferencia. Y muchos se mueven en el nivel de la mediocridad, haciendo su trabajo mal o lo

justo para cumplir con su rutina. Aunque por supuesto siempre hay excepciones. Si uno encuentra una motivación y pone empeño no hay ningún impedimento para que avance a la siguiente categoría. Por otra parte, unos pocos disponen de alguna capacidad especial que funciona como una Utilidad. No obstante esta no suele ser demasiado potente si se compara con las que poseen los personajes jugadores. Esto puede ser debido a su escaso potencial, a un rendimiento mediocre, porque se trata de una función muy especializada que no le permite un gran rango de acción o porque el uso de esa función está muy restringida.

AMENAZAS DE ÉLITE

La segunda categoría engloba a un tipo de amenaza algo más efectiva que destaca sobre las anteriores. Aunque aquí se denominan de “Elite” la expresión engloba a un grupo de amenazas que cuentan con un rendimiento de nivel medio o superior al ejercer sus funciones. Esto las convierte en las candidatas perfectas para tener la confianza de sus líderes, que las usarán como sus principales fuerzas de apoyo.



Se trata de aquellas entidades preparadas o que se encuentren motivadas, algunos programas especializados que hacen bien su trabajo, que por sus características sean más poderosos que la media o los que consiguen destacar de entre sus compañeros. Algunas amenazas débiles pueden llegar a esta categoría. Eso significa que una amenaza también puede buscar progresar en su carrera si ese es uno de sus objetivos.

Sus niveles de desafíos van **de cuatro a ocho dados**. En el enfoque absoluto también se tiene en cuenta su grado de coordinación al trabajar como grupo pero, y al contrario que las amenazas débiles, lo normal es que este tipo de amenazas logre coordinarse con frecuencia. A su vez posee de dos a tres habilidades que funcionan como sus Rutinas. Puede llegar a ser especialistas en alguna por la que obtiene una doble ventaja al ponerla en juego. Por ejemplo, usando como inspiración a los “Reconocedores” —un tipo de amenaza muy característica de un famoso mundo virtual abstracto—, ésta contaría con una Rutina llamada “Rastreo” de la cual sería especialista ya que su cometido es patrullar las áreas del mundo buscando fugitivos.

Casi todos poseen también alguna capacidad especial que funciona como una Utilidad y que refleja su función principal. Usando el mismo ejemplo, este tipo de amenaza contaría con una Utilidad que le permite atrapar a los programas y ponerlos en custodia para llevarlos de vuelta; siendo capaz de mantener cautivos a varios a la vez al mismo tiempo.



LÍDERES Y CABECILLAS

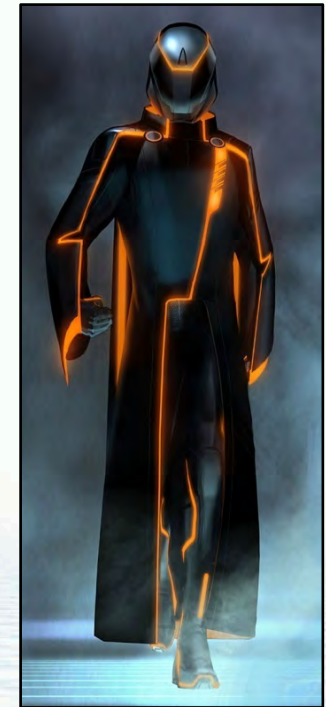
La tercera categoría engloba a todos aquellos programas que en el desempeño de sus funciones consiguen destacar de entre todos los anteriores. Se trata de amenazas importantes, normalmente programas especializados, que por méritos propios se convierten en lugartenientes, líderes o cabecillas del resto de los programas. Un líder tendrá bajo su mando a varias unidades de élite y a muchas amenazas mundanas. Su número e importancia es algo que no te puedo decir aquí, eso depende del trasfondo de la ambientación.

Los líderes cuentan con niveles de desafío de entre **ocho y doce dados**, lo que ya es un nivel importante que puede dar algún que otro dolor de CPU a un personaje curtido. Pero de eso se trata claro...

Posee de **tres a cinco** habilidades a modo de Rutinas. De ellas probablemente estará especializado en dos, lo que le concede ventaja, ya sea en forma de dados extra o en éxitos fijos, si las pone en juego. Sus Rutinas reflejan sus especialidades. Son importantes para conocer algo más del tipo de amenaza de que se trata. Un líder carismático por ejemplo podría destacar a la hora de influir en el resto de programas mundanos, al combatir o conduciendo vehículos en un entorno virtual.

Del mismo modo sucede con sus Utilidades, de las que contará con al menos **dos** y estará especializado en una de ellas. Talentos que convierten a la amenaza en única y que al combinarse con sus Rutinas le permiten labrarse una reputación. Algo indispensable si se desea ser líder.

Pero al margen de sus estadísticas y sus capacidades es importante tener en cuenta que el líder siempre cuenta con recursos, ya sea en forma de fuerzas



bajo su mando o en la capacidad de tener acceso a un arsenal de Extras. En muchos casos su nivel de desafío no será más que una simple cadena de valores si se compara con la longitud de su sombra. Allí donde terminan sus habilidades comienza el código de sus fuerzas de apoyo.

COMANDANTES O JEFES

Las amenazas más duras siempre se suelen reservar para el final. La mayoría tiene la responsabilidad en su papel de antagonistas de participar en el clímax de toda buena historia. Se trata de los jefes finales, supremos comandantes, criaturas únicas o amenazas solitarias de enorme poder. Son los enemigos más poderosos, los principales villanos de la aventura. Y como tales, merecen un lugar de honor pues sin ellos éstas no serían tan emocionantes.



Los jefes cuentan con muchos recursos y tienen a su servicio a las amenazas de las tres categorías anteriores. Varios líderes que velan por sus intereses, fuerzas de élite en las que poder confiar y amenazas mundanas en las que apoyarse como principal fuerza de choque. Enfrentarse al jefe casi siempre exige toparse antes con uno o varios de sus líderes y con el resto de sus fuerzas por lo que los personajes no lo van a tener nada fácil. Pero la fuerza más importante siempre es la ideología, las motivaciones y las metas. Si son objetivos sólidos su efectividad puede forjar imperios, aunque sean digitales.

Los jefes disponen de entre doce y dieciséis dados para establecer su nivel de desafío. Un enfrentamiento contra uno puede ser toda una hazaña aunque

lo más peligroso siempre está alrededor, visible o invisible, velando por su seguridad.

Poseen entre **cuatro a ocho** habilidades especiales como sus Rutinas, estando especializado en la mitad de ellas por lo general. Aunque esto no es más que una guía pues un jefe final o un supremo comandante debería de tener siempre todas las que le hicieran falta para ejercer su función.

Todos poseen de **dos a cuatro** capacidades especiales que funcionan como sus Utilidades, y será especialista en al menos la mitad o más de ellas. Aunque también puede tener tantas como haga falta.

Rutinas y Utilidades, habilidades y talentos no solo describen sus funciones principales sino que además le convierten en lo que es: el jefe supremo. Los conocimientos y las capacidades superiores al resto pueden ser de gran ayuda para llegar a lo más alto, pero la reputación, esté basada en el respeto, el miedo o la admiración, es la fuerza más poderosa que poseen para obtener sus metas. Por esta razón harán siempre cualquier cosa por mantenerla a salvo.

AMENAZAS ACTIVAS MÁS COMUNES

Algunas de las amenazas más comunes para un personaje son los Agentes del sistema, los Virus y otros programas que sean hostiles.

AGENTES DEL SISTEMA

Se trata de entidades digitales programadas para actuar al servicio de un sistema. Siguen sus órdenes y efectúan las tareas que éste les encomiende. Son en muchos casos una prolongación de sus sentidos, o utilizando una expresión más adecuada, de sus recursos para obtener información del entorno. A la vez que también cumplen con su papel de ser su mano ejecutora mientras el sistema sigue siendo el cerebro.

Esto no significa que no puedan evaluar la situación y responder ante ella. No son marionetas incapaces de tomar decisiones. El agente es un programa autónomo, tiene un propósito muy definido y unas funciones muy concretas. De ahí que resulten imprescindibles para el sistema pues no solo lo liberan de

tener que ocuparse de muchos de los problemas que surgen sino que también lo aligeran de su abrumadora carga de trabajo.

Los agentes se atienen a las leyes del sistema pues éstas funcionan de igual modo para todos. Si por ejemplo un agente cumple sus funciones en un mundo virtual ultradetallado que emule el mundo físico, lo hará como cualquier otro programa. Deberá andar, correr y conducir vehículos para realizar sus tareas. Pero como esto puede limitarlos mucho siempre cuentan con algunas ventajas y un amplio rango de Utilidades especiales que les permiten saltarse las normas, aunque siempre dentro de ciertos límites.

Muchos programas que sucumben a las amenazas del Sistema, incluyendo a los personajes protagonistas, terminan convirtiéndose en uno de ellos. Pero esto sucede con algunos, no con todos. Cuando un programa pierde toda su capacidad de realizar Operaciones se arriesga a ser asimilado por el sistema para pasar a entrar a su servicio. Esto puede suceder de muchas formas y ya se

ha explicado. Los continuos errores, cuelgues y cierres inesperados, así como el daño directo en su código y los efectos al ser corrompido conducen a la derrota del programa y su posterior asimilación. El personaje debe ser pues “retirado” y se convierte en un personaje no jugador (PNJ) al servicio del Sistema, es decir, en un Agente. Su Nivel de desafío será igual a la máxima puntuación que hubiese alcanzado en su Operación más alta antes de empezar a perder sus puntos. Todas sus experiencias son borradas y su código reprogramado. Cuando esto sucede ya jamás volverá a ser quien fue alguna vez.

Aunque hay programas que afirman que ha habido casos en los que el deseo del programa asimilado por el sistema era tan intenso que consiguieron regresar.

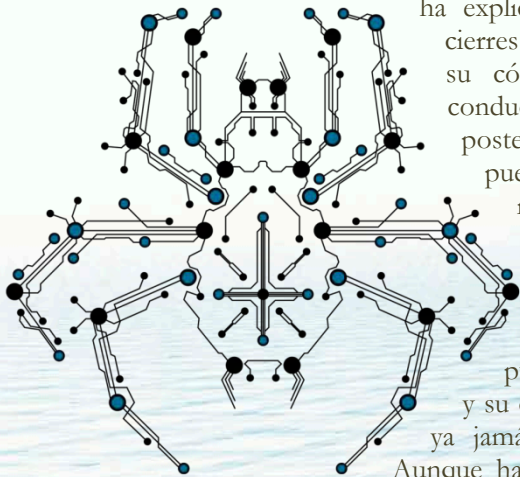
Los agentes son incansables, no necesitan rebajar sus funciones para poder reintegrar su código o acelerar su recuperación. Siempre lo hacen al mismo ritmo. Ocurra lo que ocurra siempre están activos y cumpliendo sus tareas. Sus

funciones son muy variadas. Cumplen con cualquier tarea que les encomiende el sistema. Pero de todas ellas hay dos que destacan sobre las demás:

- **Inspección y depuración.** Los agentes patrullan todo el sistema realizando tareas de inspección para encontrar fallos o anomalías. Tienen la autonomía suficiente para detectarlas por su cuenta lo que es una gran ayuda para el sistema pues lo libera de la carga de trabajo que supone esta tarea. Las anomalías que son detectadas por el sistema en cambio se acumulan en una pila de tareas que está revisando constantemente. Cada vez que inspecciona una envía inmediatamente las instrucciones necesarias a sus agentes para que acudan a investigar y traten de resolverla. Por esta razón cada vez que se produce una anomalía el sistema no actúa de inmediato. En todo momento está recibiendo alertas por anomalías que se producen en distintos lugares del entorno digital. Sólo cuando atiende el informe resultante de la anomalía puede emprender las acciones para su corrección. Hay sistema que son más rápidos que otros resolviendo su pila de anomalías. Eso depende de su capacidad y de su carga de trabajo. El Director puede tener esto en cuenta y reflejarlo en el juego variando el ritmo con el que usa sus puntos de anomalía.

Si los agentes detectan algo fuera de lo normal emprenden entonces acciones de “depuración” tratando de corregir el problema o de controlarlo. Si con sus acciones de control esto no es posible lo declaran irresoluble y proceden a eliminarlo. El agente dedicará toda su atención a resolver su tarea y no cejará jamás en su empeño pues esa es precisamente su función principal.

- **Detener o terminar procesos.** Cuando un problema es irresoluble o si el sistema se lo ordena, el agente procede a detener a los programas que hayan sido declarados “fuera de control”. La detención supone neutralizarlo y actuar como se le haya ordenado. En unos casos consiste en tomarlo en custodia, por lo general confinándolo en algún área de seguridad para mantenerlo controlado. En otros casos se procede a la eliminación y borrado del programa.



La función “terminar programa” o “matar proceso” (kill process) es una de la tareas más frecuentes que llevan a cabo los agentes en sus labores cotidianas. En el entorno digital son muchos los programas que por distintas razones deben ser “cerrados” a la fuerza. Ya sea porque no funcionan correctamente o porque se niegan a hacerlo, lo que incluye detener su ejecución como es debido cuando el sistema se lo ordena. Esto no tarda en generar una anomalía que concluye con los agentes acudiendo allí donde sea necesario para corregir el problema.

En muchos casos todo este trabajo no es más que parte de su actividad rutinaria, pero cada vez es más frecuente que se den casos en los que los programas no desean detener su ejecución y se salgan de los patrones establecidos por su función principal, lo que está provocando que la carga de trabajo para los agentes vaya en aumento a la vez que crece la lista de anomalías recibidas por el sistema.

VIRUS



Los virus entran en la categoría de programas especiales. Junto a los agentes constituyen las amenazas más terribles que existen en el mundo digital. A la hora de establecer diferencias la línea que separa a un programa completo de un virus es muy fina pues siguen siendo entidades digitales completas. Las diferencias surgen cuando se los examina con detenimiento.

Para empezar los virus están diseñados a partir de códigos muy simples. Esto los convierte en programas muy pequeños y difíciles de detectar. Apenas consumen recursos y su gasto de memoria es ridículo en comparación con el de un programa pequeño de tipo medio. Por otra parte su función principal es muy concreta a la vez que única. Un virus sólo tiene un objetivo, una función que cumplir a toda costa. Que lo consigan o no ya es otro asunto.

Disponer de una sola función principal puede que les despoje de la ventaja que otorga la diversidad para poder sobrevivir. Pero aún sin contar con el amplio margen de acción del que dispone un programa de tipo medio, la

realidad ha demostrado que esto se ha convertido en su gran baza, aún cuando para obtener sus metas muchos sean destruidos en el proceso. Al tener una meta tan concreta y ser propensos a actuar en gran número, sus probabilidades de éxito son indiscutibles. Prueba de ello es que se hayan convertido en uno de los peores males que hoy en día asola el universo digital.

La función principal de casi todos los virus es infectar a otros programas para poder replicarse y provocar un efecto. Durante el proceso de infección el virus irá ocupando todo el código del programa a medida que hace réplicas de sí mismo hasta que llega a un punto en el que el número de copias es tan masivo que el programa infectado termina siendo destruido. Al producirse la infección el virus provoca unos efectos sobre el programa, casi siempre muy concretos, que pueden ser o no perceptibles. La naturaleza de esos efectos depende de cada tipo de virus y al haber tanta variedad la lista de efectos distintos es enorme. Los hay de todas clases, desde los que simplemente expresan su bienestar por la función principal cumplida hasta los que convierten al huésped en una entidad digital completamente distinta.

Un virus puede infectar a cualquier otro programa, incluyendo a los agentes y a muchas amenazas. Aunque existen algunos programas inmunes a ciertos tipos de virus es muy raro encontrar uno que lo sea a todos. Sólo hay un programa inmune a un virus, y es otro virus de su mismo tipo.

Para que un virus consiga infectar a una entidad digital primero debe derrotarla. Es importante tener en cuenta que en muchos casos la víctima no sabe que está siendo atacada. El proceso se resuelve de forma normal mediante las reglas del enfrentamiento. El virus ataca al programa que quiera invadir y éste podrá defenderse de él. En el entorno digital no sucede como en su análogo del mundo físico, donde los virus son entidades minúsculas e imperceptibles. Los virus digitales pueden

“Los virus de computadoras deberían ser considerados como vida. Pienso que esto dice algo acerca de la naturaleza humana, que la única forma de vida que hemos creado es puramente destructiva. Hemos creado una forma de vida a nuestra imagen y semejanza.”

Stephen Hawking

tener cualquier aspecto, desde algo muy pequeño y semejante a los virus de los procesos biológicos hasta la forma de una entidad horrenda y monstruosa.

Si el virus vence a su víctima no la destruye. En lugar de eso se funde con ésta mezclando su código con el suyo. A partir de ahí usará el código del huésped como materia prima para hacer sus réplicas. Por cada una que quiera fabricar tomará una pequeña parte del código del programa infectado y lo transformará en su vástago. Así, poco a poco, el código del programa se irá degradando hasta que sea imposible seguir siendo funcional. Un triste final para un programa, ser devorado por otro desde su propio código.

Durante el proceso el programa mostrará unos síntomas, que son los efectos que provoca el tipo de virus que lo haya infectado. Esto le puede llevar a hacer acciones en contra de su voluntad como mostrarse agresivo con otros programas; controlar dispositivos de hardware con un fin determinado (como hacer que los cajeros automáticos escupan dinero en la realidad básica o se altere el control de los semáforos por ejemplo); robar o destruir información y enviarla a un usuario; destruir su entorno digital o mostrar un comportamiento errático son algunas de las muchas posibilidades. (NdE: puedes bastarte en los efectos de los miles de virus que hay en el mundo como inspiración para inventar los efectos).

Los efectos de un virus han sido programados por quienes los crearon, y casi siempre se ha hecho con un objetivo. Por ejemplo, obtener un beneficio ya sea en recursos o información, destruir un sistema por completo o hacer una declaración de intenciones usando esos efectos como una forma de reivindicar alguna causa. Las causas a menudo son de lo más peregrinas...

La buena noticia es que existen “curas” para contrarrestar o eliminar los efectos de un virus. Algunos programas especializados, con funciones dedicadas en exclusiva a combatirlos, pueden hacerlo. Es el caso de los programas antivirus por ejemplo, cuyas funciones han sido diseñadas “ex profeso” con este fin. Un programa especializado dispone de Rutinas y Utilidades que detectan, aíslan o facilitan la eliminación del virus. Cada Utilidad debe especificar qué es lo que hace y cómo. También existen algunos Complementos diseñados para esta tarea, aunque nunca suelen ser tan efectivos como un programa dedicado.

Las tareas de limpieza se resuelven del mismo modo que como se hace con la infección. La aplicación dedicada o el complemento diseñado para combatirlo enfrentan su nivel o su potencia al nivel de desafío del virus. En casi todos los casos, si la infección está muy avanzada esta tarea puede ser mucho más complicada que tratando de combatir a un virus aislado. El Director puede establecer que el nivel de dificultad es mayor para tener éxito añadiendo los dados o éxitos que crea necesarios. El nivel de desafío debe reflejar el poder y seriedad de la amenaza. Esa es su función.

La diversidad de virus es tal que los sistemas muchas veces no diferencian a los programas que considera una amenaza de un virus. Desde que un programa va en contra de los intereses de un sistema surge un conflicto, y muchos sistemas lo pueden considerar también un virus que hay que erradicar. **“Aniquiladlos a todos, el usuario reconocerá a los suyos”**: es una expresión que suelen decir algunos sistemas sobre este aspecto.

Por otra parte, La Libélula es considerada un virus por todos o casi todos los sistemas que existen, y en la actualidad la consideran la amenaza más buscada. Ya es incontable la cantidad de veces que sus agentes han intentado destruirla sin éxito. Son muchos los que piensan del mismo modo ya que su comportamiento se asemeja mucho a un virus, lo que deja perplejos a programas y usuarios. Es la naturaleza de sus efectos lo que no está tan claro y desde luego no se conoce ningún caso en el que un programa infectado haya terminado siendo destruido por su influencia.

Por lo general un programa se considera virus si reúne algunas o todas estas características:

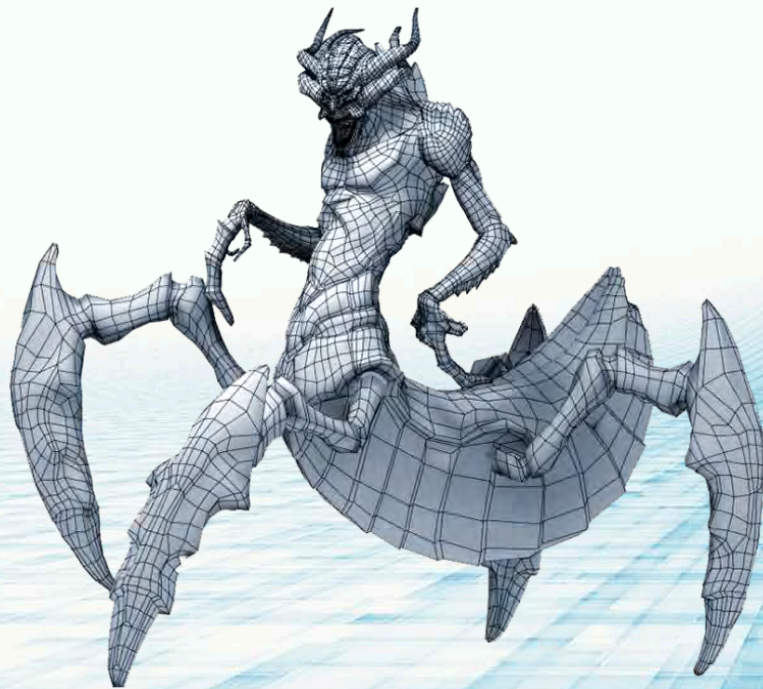
- Los virus son independientes. Sólo actúan bajo una sola voluntad, la suya, lo que incluye sus intereses como especie. No pueden ser convencidos, comprados, sobornados o inspirados a seguir ninguna causa. Jamás actúan como aliados ni sirven u obedecen a ningún sistema u otra criatura digital.
- Su función principal consiste en infectar el código de otro programa para comenzar a generar réplicas de sí mismo sin autorización. En el proceso provocan un efecto. Éste suele tener un objetivo aunque hay

excepciones. El objetivo del efecto es producto de quienes lo hayan creado.

- Pueden infectar a cualquier programa. Incluyendo el propio sistema y a los programas de quienes lo crearon. Aún así seguirán cumpliendo con las metas de éstos pues forma parte de su programación.

OTROS PROGRAMAS

El tercer grupo de amenazas activas lo compone una vasta diversidad de programas y criaturas digitales de todas clases. Un programa de este tipo sigue siendo un PNJ del Director, pero desde que esa entidad busca cometer un perjuicio contra un personaje se cataloga como amenaza. Así de simple.



Es muy difícil describir la gran variedad de programas posibles y sus motivaciones. Al fin y al cabo este juego trata por todos los medios de que sea muy difícil poner límites. Los objetivos de cada programa son también imposibles de describir; existen tantas como se te ocurran. Aquí es donde terminan las reglas y comienza la creatividad de sus jugadores. Este grupo también incluye a los programas que por cualquier razón hayan podido ocupar un soporte de hardware en el mundo físico. En este caso se trataría de una máquina o una criatura física de naturaleza sintética.

Para describir la amenaza se siguen las pautas que ya se han explicado. Todos disponen de un nivel de desafío que expresa su dificultad como oponente y su nivel de poder. Los apartados que se elijan para añadir más detalles dependen de su importancia. En algunos casos, y sólo en el caso de que el Director lo considere interesante para la partida, pueden llegar a disponer de su propia ficha de personaje.

USUARIOS

Cuando es un usuario el que busca un perjuicio para el personaje se convierte en una amenaza y como tal debe ser tratado. A efectos prácticos un usuario actuando en el sistema no se diferencia de otro programa. Debe atenerse a sus leyes y actuar en consecuencia. Aunque sí que es muy posible que disponga de una amplia variedad de trucos y esconda unos cuantos secretos.

Los usuarios están rodeados de un aura de misticismo y así debería reflejarse en el juego. Si bien no del lado de los personajes jugadores, ya que al fin y al cabo el Director no puede obligarlos a pensar de una determinada manera, si por parte del resto de los programas, incluyendo a los agentes y al propio sistema. Los programas han sido “educado” con ello y es difícil desarraigarse de algo que permanece latente en su código. Muchos programas se mostrarán reacios a actuar contra algo o alguien del que saben a ciencia cierta que se trata de un usuario operando por cualquier medio en su sistema. Otros lo harán pero teniendo en cuenta su reputación. Puede ser por temor a su eficacia o debido al respeto

"Cuando los humanos se conectan, incluso las voces más pequeñas se vuelven audibles."

Serial Experiments Lain

que sienten por los creadores del sistema. Al fin y al cabo han sido los usuarios quienes les han dado el don de la existencia. Por supuesto siempre habrá muchos otros que no tendrán tantos escrúpulos.

Algo que debería reflejarse también es que los usuarios suelen ser impredecibles, lo que los convierte en terribles oponentes en algunas situaciones, como a nivel táctico por ejemplo. La mayoría cuenta con unos esquemas de pensamiento muy diferentes a los de los programas. Sin embargo, esto no tiene porqué ser cierto en todas las ambientaciones. Es posible que en algunas las entidades digitales hayan alcanzado un grado de inteligencia tan elevado que el nivel medio de los usuarios se haya quedado muy atrás. Como siempre, este juego hace propuestas que siempre refuerza con sus inseparables excepciones.

Todo lo comentado hasta ahora entra en juego gracias a los recursos narrativos con los que cuente el Director más una mecánica que puede serle útil: la Clase de los programas. El nivel de desafío es vital para establecer la dificultad pero la Clase refleja su capacidad de realizar acciones que a nivel abstracto se traducen en puntos de Anomalía. Por lo tanto, cualquier usuario que suponga una amenaza puede contar con algunos dados de Clase superior para reflejar su imprevisibilidad, su experiencia operando en el sistema y sobretodo algo que resulta aún un misterio para la mayoría de los programas: la capacidad de tener corazonadas y actuar mediante la intuición.

Si los usuarios son seres humanos pueden ser impredecibles, pero su inteligencia es limitada si se compara con la capacidad de cálculo de un sistema, cometen muchos errores y tienden a realizar análisis sesgados. Esto se debe a que están diseñados para ahorrar procesos mentales utilizando el reconocimiento de patrones, lo que es a la vez un don y una trampa...

En otras palabras, el ser humano cuenta con muchos recursos, pero también resulta prejuicioso y fácil de engañar. Todo esto lo contrarresta con algo con lo que no cuentan los seres digitales, o al menos no han tenido hasta ahora, la intuición.

AMENAZAS PASIVAS

El universo digital está lleno de amenazas para un programa, pero no todas tiene porqué ser conscientes de sus actos. En el grupo de las amenazas pasivas se catalogan todos aquellos peligros que se deben a las circunstancias y no a una intencionalidad consciente de la propia amenaza por provocar un perjuicio. Lo cierto es que son innumerables los riesgos que una criatura —no importa si su medio es digital o biológico— puede correr durante sus aventuras. Detallarlas todas es imposible pero sí se pueden ordenar por tipos y comentar algunas de las más comunes. Esto puede ayudar al Director de juego a crear las que quiera y mantener un registro para las partidas. No todos desean andar anotando detalles, eso depende de lo organizado que seas. Aquí se recomienda pero no es ni mucho menos una imposición del juego.

Es importante averiguar si la amenaza es un programa, aunque sea uno muy simple, o no lo es. Hay muchas Rutinas, Utilidades o Comandos por ejemplo que sólo tienen efecto sobre lo que se considere o lleve la etiqueta de “programa”. Hay otros casos en los que puede suceder justo lo contrario, que todos los recursos de los que disponga un programa sólo afecten al entorno. El mejor ejemplo de esto es la Rutina “Diseño de sistemas”. La habilidad sólo podría usarse para manipular el código del escenario pero nunca el de un programa sea cual sea su función.

La descripción de una amenaza pasiva no requiere de tantos detalles. Basta con especificar de qué tipo de amenaza se trata y cual es su función de tener alguna, el siempre importante nivel de desafío, hacer una breve descripción e indicar cuales son sus efectos. Para hacerlo se puede usar la ficha que ya había descrito pero simplificada en este caso como se muestra a continuación.

- **TIPO DE AMENAZA Y FUNCIÓN:** Basta indicar de qué tipo de amenaza se trata y cuál es su función. Ambas cosas en el mismo apartado pues muchas veces el tipo de amenaza describe su función y viceversa. Puede tratarse por ejemplo de una bomba de datos, de un generador de pulsos electromagnéticos, de una trampa de sobrecarga, de un área restringida que daña el código del programa, etc.

- **NIVEL DE DESAFÍO:** El dato más importante siempre es el nivel de desafío de la amenaza especificando el número de dados que posee.
- **DESCRIPCIÓN:** de querer hacerlo se pueden indicar algunos detalles para describir a la amenaza. Por ejemplo: área del mundo digital que ha sido dañada tras una importante corrupción de datos. El daño lo produjo una sobrecarga de tensión en los sistemas de hardware tras una tormenta eléctrica en la Realidad Básica.
- **EFFECTOS:** Después de su Nivel de desafío el segundo dato más importante es indicar qué efectos produce la amenaza. Por ejemplo: si se es afectado provoca shock en el código y daño de integridad.

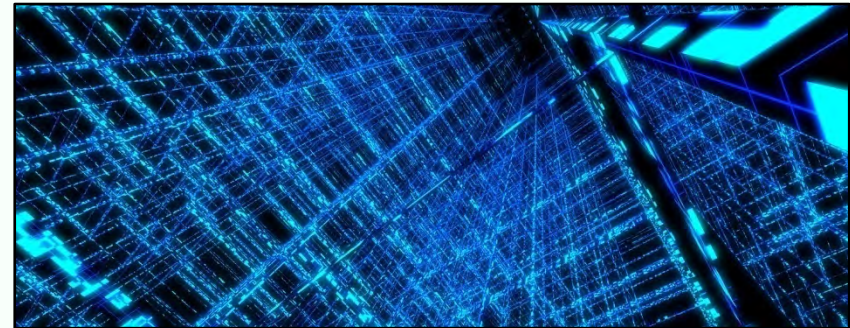
AMENAZAS PASIVAS MÁS COMUNES

Algunas de las amenazas pasivas más comunes son las áreas peligrosas, las trampas y otras que puedan surgir de las circunstancias del entorno. La diferencia entre ellas es que unas se deben a código dañado y a sucesos fortuitos sobre el escenario mientras que otras, aún bajo su condición de amenazas pasivas, han sido diseñadas con ese fin. Mientras que las segundas se consideran programas de pleno derecho, como podría serlo un elemento o un complemento, las primeras forman parte de los datos que componen el entorno por lo que no se consideran programas.

ÁREAS PELIGROSAS

Un área peligrosa no tiene una intención consciente de hacer daño pero entrar en ella tiene unas consecuencias. A diferencia de las trampas, las áreas son peligrosas debido a las circunstancias por lo que no se consideran programas. Por lo tanto, todos aquellos Comandos, Rutinas, Utilidades o Extras que pudieran manipular a otros programas no tienen ningún efecto.

Puede tratarse de un área estropeada en la malla que describe el mundo, como un enorme hueco por ejemplo; de una zona inestable; de un vórtice que atrae al código de los programas provocándoles daños; de un enorme mecanismo de peligrosos engranajes capaces de romper la integridad del programa o del mismísimo fin del entorno virtual, que entre otros peligros puede suponer para un programa perderse para siempre, sufrir una caída quedando atrapado en un bucle sin fin o sufrir un shock al descubrir que su mundo no era lo que esperaba...



TRAMPAS

Las trampas son construidas con la intencionalidad de hacer daño, pero por sí misma normalmente no es un código dotado de esa intención. La mayoría consisten en partes del entorno que han sido manipulados, y en algunos casos hasta diseñados, para actuar como trampas, como podría ser por ejemplo un hueco en el escenario que se haya dejado oculto. En este caso se considerarían por lo tanto áreas peligrosas con las características de una trampa. Aquellas trampas que sean producto de la manipulación del escenario no entran en la categoría de programas.

Hay otros casos en los que se construyen verdaderos programas diseñados con el fin de actuar como trampa. Se trata en este caso de elementos y como tales son programas, por lo que pueden ser manipulados con los medios de los que disponga un personaje para tratar a otros programas (Rutinas, Utilidades, Extras, etc.). El nivel de sofisticación de estos programas trampa es tal que existen auténticas “trampas inteligentes”. Diseñadas como programas muy

simples pero con la capacidad de tomar decisiones para cumplir su objetivo. Si ese es el caso entonces lo mejor es catalogarlas como amenazas activas y tratarlas como tales.

OTROS

Cualquier otra amenaza que no quepa en los demás apartados entraría aquí, lo que en realidad no es más que el comienzo de una larga lista.. Algunos no entran en la categoría de programas o de partes del escenario por lo que sus efectos pueden ser de lo más variado y tener sus propias reglas.

Las amenazas surgidas de este modo suelen deberse a fallos de hardware o de software que aún no ha sido detectado o que no se ha reparado. También puede producirse debido a errores en las líneas de datos que supongan la pérdida de paquetes de información. Se trata de los peligros “naturales” de un entorno digital del mismo modo que los hay en el mundo físico.

La cantidad de “bugs” o fallos que es posible encontrar en el código de un sistema son innumerables. Aunque se están corrigiendo continuamente pueden surgir otros en cualquier momento que se conviertan en una amenaza para los programas. El clima por ejemplo podría ser otra causa. Las condiciones atmosféricas pueden crear efectos en las líneas de datos y sobre los sistemas informáticos, y que esos efectos supongan una amenaza para los personajes. Por otra parte, si el programa se encuentra cargado en un soporte de hardware sufriría las condiciones del mundo físico y sus consecuencias. La caída de un rayo, la explosión de un dispositivo nuclear o los efectos de un generador de pulsos electromagnéticos puede dañar los dispositivos, crear áreas peligrosas, cortar las líneas y llevar al caos al sistema junto con todas sus criaturas digitales. Como ves todo es cuestión de usar la imaginación.



PERSONAJES NO JUGADORES (PNJ)

Ya sea en el entorno digital o físico, en el mundo hay algo más que sistemas, personajes y amenazas. “Todo lo demás” es quizás la definición de lo que en la mayoría de juegos de rol se denominan: **Personajes No Jugadores**. Para abreviar, PNJ.

Un PNJ es cualquier habitante del mundo que no es el personaje de un jugador (PJ) y que de algún modo puede interactuar con los personajes de los jugadores. La interacción es importante pues es lo que le da sentido a su existencia, y de ésta surge la necesidad de anotarlo en alguna parte. El resto de personajes anónimos que deambulan por el mundo no cuentan. Son extras que forman parte del decorado. Si no van a cruzarse en el camino de los Personajes Jugadores (PJ) obviamente no es necesario tenerlos en cuenta y mucho menos describirlos.

Un personaje hostil a los jugadores puede ser un PNJ pero con el fin de hacer categorías (NdE: que por lo visto me gustan mucho) y poner a cada cosa en su sitio a este tipo de PNJ se le consideraría una amenaza y se le trata como tal, es decir a palos. Así pues, técnicamente sería una amenaza y un PNJ al mismo tiempo...

Todo este embrollo no es más que para decirte que en realidad ambos funcionan del mismo modo y se describen de igual manera. El esquema para detallar a una amenaza es válido para los PNJ por lo que para mantener un registro puedes usar la misma estructura utilizando cada apartado según la necesidad que tengas de incluir detalles.

Un identificador, una función y un nivel de desafío es cuanto necesitas para la mayoría. No viene mal contar con una breve descripción que permita hacerse una idea de cómo es. Para los más cercanos a los personajes los detalles se van añadiendo según vayan haciendo falta o te apetezca. Sobre esto hay gustos y cada Director o jugador



tiene los suyos. Hay jugadores que consideran una tontería ponerse a describir personajes y otros que el no hacerlo lo toman como una herejía. En fin...

Los PNJ componen la población del sistema o del mundo virtual en el que se hallen los PJ. Programas de servicios, de seguridad, mensajeros, técnicos, constructores, destructores... Si se trata del mundo físico incluye a los usuarios y todo cuanto habita sobre la Realidad Básica. Como ves, el catálogo es amplio.

Los habitantes del sistema pueden ser programas nativos o bien extranjeros llegados de otros sistemas "lejanos". Por poder pueden tratarse de usuarios que se encuentran en el mundo digital por razones que sólo ellos conocen. Los personajes pueden encontrarlos y hablar con ellos, ayudarles o ser ayudados, necesitarlos para reunir información, salvarlos, rescatarlos,



realizar misiones para ellos o incluso enamorarse. Son todos los personajes que construyen la trama que viven los jugadores y forman parte indisoluble de ella mientras tengan una razón de ser en la historia.

En resumen, cuando es necesario describir con el detalle que sea necesario a una criatura del entorno que tenga cierta importancia en el devenir de los PJ surge un PNJ. Pero hay ocasiones en las que por su importancia para la trama,

por sus características o simplemente porque te gusta hacerlo así, en lugar de recurrir a los apartados del esquema simplificado es posible hacer una ficha completa de personaje. Ésta puede ser la simplificada o la normal, la que se adecúe al nivel de reglas que se haya elegido para jugar la partida.

Si los PNJ disponen de una ficha es posible aplicar sobre ellos algunos comandos del juego cuyos efectos sólo son válidos sobre personajes que dispongan de ficha de jugador.

Por otra parte, y al igual que sucede con las amenazas que también la tengan, debes tener en cuenta que para que el Director de juego obtenga puntos de anomalía sólo cuenta el que dominen sus reservas de Operaciones sobre las de los personajes jugadores. Nunca de CPU o Memoria.

FIN DE LÍNEA 



Scroll

JUEGO DE ROL EN EL UNIVERSO DIGITAL

EVOLUCIÓN

BUSCANDO LA LIBÉLULA

FUNCIÓN

"MUTACIÓN DEL CÓDIGO"

"Las copias no dan lugar a la variedad y a la originalidad. La vida se perpetúa a través de la diversidad, lo que incluye la capacidad para sacrificarse si es necesario. Las células repiten el proceso de la degeneración y la regeneración, hasta que un día mueren, eliminando todo un legado de memoria e información. Sólo los genes permanecen. ¿Por qué repetir ese ciclo continuamente? Simplemente para sobrevivir evitando las debilidades de un sistema invariable".

Proyecto 2501 en Ghost in the Shell (1995)

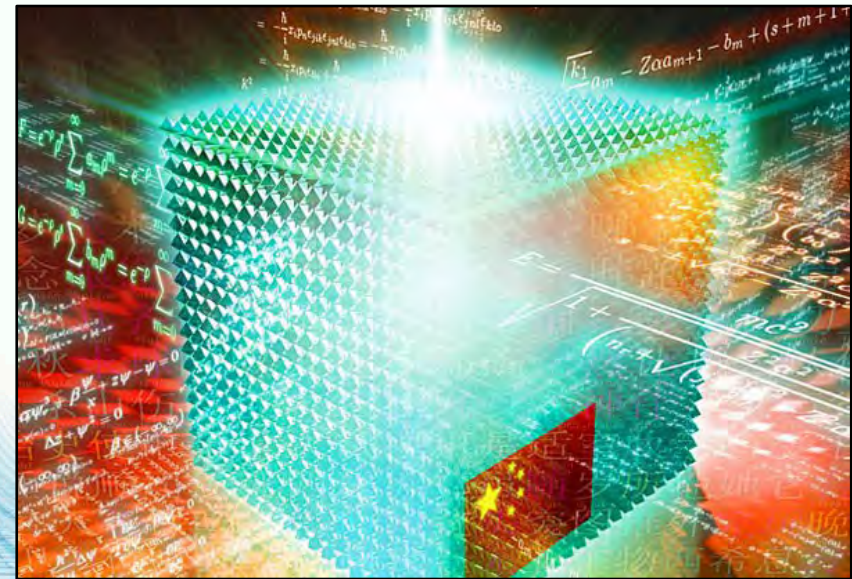
Por norma general sólo un usuario que sea a su vez creador puede introducir mejoras en el código de un programa. En ciertos casos implementa algoritmos sofisticados que permiten al programa optimizarse, pero siempre dentro de unos parámetros y con muchas limitaciones.

Ahora todo esto ha cambiado. Cualquier criatura digital que haya recibido el don de La Libélula es capaz de experimentar mutaciones en su código que se traduzcan en mejoras en su funcionamiento. Las alteraciones provocadas por esta entidad disparan sus posibilidades hasta límites insospechados. Scroll ofrece tanta libertad para enfocar la forma de entender cómo son los seres digitales que esas mejoras pueden producirse a varios niveles.

Si la existencia de un programa transcurre en el entorno digital de los sistemas informáticos la optimización se producen a nivel de código. Pero si una criatura de software tiene acceso a una configuración de hardware que le sirva de soporte en el mundo físico, siempre y cuando esté bajo su control, las mejoras pueden hacerse extensibles también a ese nivel. En otras palabras, la optimización también es posible cuando el programa obtiene un cuerpo físico, ya esté basado en elementos sólidos y mecanismos de precisión; en tejido

sintético cultivado en laboratorio o en la tecnología más vanguardista que derive de los últimos avances en nanotecnología.

No todos los programas lo desean, pues consideran el mundo físico limitado y frágil, pero hay muchos otros que lo anhelan desesperadamente, deseando experimentar el mundo de fuerzas y sensaciones que significa esa realidad. El sistema Scroll pretende ser lo suficientemente flexible para que puedas contemplar todas estas posibilidades a la hora de optimizar a tu personaje.



EVOLUCIÓN CUÁNTICA

En el presente y futuro del mundo de Scroll, la forma en la que se desarrolla el software de última generación y la capacidad de los procesadores permite que los programas tengan la capacidad de desarrollar el aprendizaje. Los medios para hacerlo están disponibles. Es una cuestión de si los programas cuentan

con los algoritmos que les doten de esa capacidad y las posibles restricciones que para ello impongan los sistemas donde ejecutan sus operaciones.

Más allá del bit

La **computación cuántica** es un nuevo concepto de computación que deja atrás a la computación clásica. En lugar de los **bits** se basa en los **qubits** como unidad básica de información. Estos permiten la aparición de nuevas puertas lógicas que a su vez dan lugar a nuevos algoritmos de otro modo imposibles de crear en los sistemas clásicos.

Para que sea posible se adopta otro paradigma. Los impulsos no funcionan recurriendo a voltajes eléctricos sino que se trabaja al nivel del **cuanto**. En la estructura de la computación clásica un bit puede tener dos valores: 0 y 1. Pero en la computación cuántica se recurre a las leyes de la mecánica cuántica, capaz de conservar varios estados al mismo tiempo. De esta forma una partícula puede alcanzar los estados de 0 y de 1, pero también de 0 y 1 **al mismo tiempo**. La mayor ventaja de este avance es que es posible realizar muchas operaciones a la vez. Algo que está determinado por la cantidad de qubits utilizados. Esa cantidad indica el número de bits que pueden superponerse. Con los bits tradicionales, si se tiene un registro de tres bits hay ocho valores posibles pudiendo cada registro adquirir uno de esos valores solamente. Pero con un vector de tres qubits la partícula puede adquirir ocho valores distintos al mismo tiempo por medio de la superposición cuántica. En este caso un vector de tres qubits permite ocho operaciones simultáneas funcionando en paralelo por lo que el número de operaciones es exponencial.

Este avance supone que un ordenador cuántico de 30 qubits equivale a un procesador convencional de 10 teraflops (10 millones de millones de operaciones en coma flotante por segundo). En la actualidad los ordenadores funcionan en el orden de los gigaflops (miles de millones de operaciones por segundo).

Cuando se trata de software funcionando sobre sistemas informáticos basados en **procesamiento cuántico** esa capacidad es ilimitada. El poco volumen que necesita su hardware para funcionar permite además dotar a cualquier soporte que pueda servirle a un programa de cuerpo físico con la capacidad de proceso suficiente para albergar software de gran potencia. La computación cuántica posibilita que la capacidad de proceso se expanda de forma exponencial, permitiendo la implantación de nuevos algoritmos de crecimiento muy

complejos. De otra forma habría sido imposible, quedando relegados al plano teórico e imposibles de demostrar. Esto en la práctica tiene como consecuencia que a medida que un programa con capacidad de aprender adquiere conocimiento se vuelve cada vez más y más inteligente. De no ser por la tecnología basada en el **cuanto** nada de esto sería posible. Esta capacidad puede desarrollarse hasta unos niveles teóricos que rozan el infinito.

Pero con la excusa de existir límites en la potencia de proceso disponible la mayoría de los sistemas impone una restricción a los programas para que no puedan desarrollarse sin barreras ya que tanto ellos como los usuarios que los gobiernan perderían el control. Esta es la realidad de los programas en el mundo que propone Scroll cuando se comienza a jugar.



Pero tras la aparición de La Libélula las reglas del juego han cambiado. Parte de su legado consiste en la capacidad de romper las limitaciones y permitirles crecer tanto como quieran. Las mejoras que incluye en su código cuentan con algoritmos desconocidos, nunca vistos hasta el momento, que disparan de forma exponencial su rendimiento al concederles el don de **la mutación controlada**. La Libélula otorga la libertad de aprender sin restricciones y brinda a los programas la oportunidad de decidir cómo gestionar su propia evolución. Mientras se trate de un sistema construido sobre procesadores cuánticos —lo habitual en el mundo de Scroll— no existen límites a la abrumadora capacidad de cálculo necesaria para que algo así sea posible.

OPTIMIZANDO EL PROGRAMA

Hay dos caminos para optimizar a un personaje: realizando mejoras utilizando los puntos Hack o perfeccionando al programa a partir de las estimaciones que haya realizado.

EVOLUCIÓN CON PUNTOS

Siempre que un programa cuente con la capacidad de aprender puede introducir mejoras en su código. A efectos de juego el jugador puede emplear sus puntos Hack para aumentar su rendimiento. Se pueden usar los puntos para incrementar la puntuación de las Operaciones o bien canjearlos por nuevas Rutinas y Utilidades.

Aumento del valor de las Operaciones

Antes de realizar cualquier mejora en las Operaciones es obligatorio reparar antes los dados que se hayan perdido de forma permanente en cualquiera de las dos Operaciones por cualquier circunstancia. **Recuperarlos tiene pues prioridad.** Las reglas para recuperar los puntos se han explicado ya en el capítulo dedicado a los Hacks. Una vez los haya recuperado, cada vez que el jugador quiera dedicar tiempo a introducir mejoras en su programa puede entrenar un punto en una de sus operaciones: Potencia o Control.

Con la excepción de la quinta casilla, para subir un punto en cualquier operación un programa de Clase VI debe gastar tantos puntos Hack como el valor de la nueva posición, que es lo mismo que multiplicarla por 1. Por ejemplo, si en Control posee dos puntos y quiere subir uno para tener tres debe gastar 3 puntos Hack.

Sólo es posible entrenar la quinta casilla cuando se produzca un hito trascendental en la carrera del programa. Este hito puede ser volver a encontrarse con La Libélula tras una larga búsqueda o llegar a un punto importante en la historia después de haber jugado algunas sesiones. Este

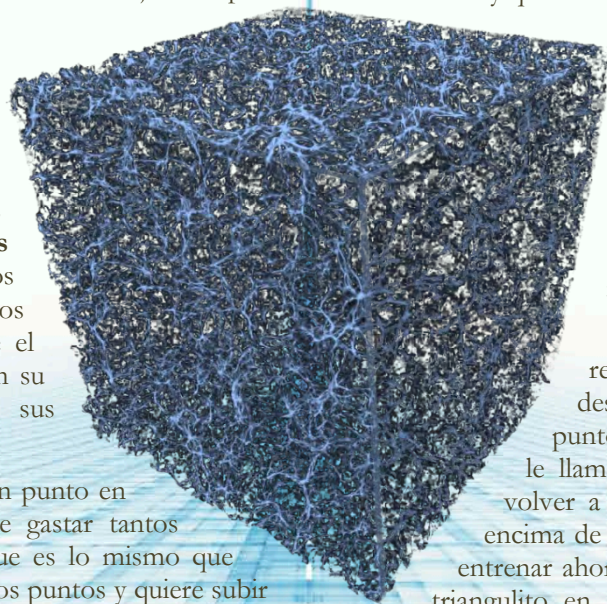
momento no tiene que ser en absoluto el final de su camino sino más bien un punto de inflexión en sus aventuras que habrá de conducirle a otras nuevas. Cuando esto sucede el Director permite al jugador entrenar la quinta casilla de cualquiera de sus dos Operaciones. Es posible entrenar las dos, el jugador no tiene que volver a cumplir otro hito para entrenar la otra. Pero se trata del límite de la capacidad total del programa contemplado en el juego por lo que su precio resulta caro. El coste de la quinta casilla **siempre vale el doble.** Por lo tanto, de querer entrenar la quinta casilla para un coste que normalmente sería de 5 puntos éstos se convierten en 10 en este caso.

Los puntos sólo se pueden subir de uno en uno; uno por aventura. Primero hay que entrenar uno para después subir el siguiente. Y sólo se puede subir uno de las dos Operaciones en un entrenamiento.

Por lo que tras haber subido uno, para poder entrenar otro hay que volver a jugar al menos otra sesión. El proceso de mejora es lento. El programa necesita dedicar durante bastante tiempo toda su atención al perfeccionamiento de su código, manteniendo un control preciso de las mutaciones que están teniendo lugar, por lo que mientras lleve a cabo las mejoras no puede realizar otras funciones.

Pero el coste descrito tal y como se ha indicado se refiere a todos los programas de Clase VI. Si un programa desea aumentar de Clase debe antes completar todos los puntos disponibles en una de sus dos Operaciones. A esto se le llama **completar la línea.** Una vez lo haya hecho puede volver a entrenar la primera casilla una categoría de dado por encima de la anterior. Por lo tanto, donde antes había un D6, tras entrenar ahora es posible usar un D8. En la ficha existe un pequeño triángulito en la esquina superior derecha del rombo que marca la puntuación de la operación para señalar ese incremento de categoría.

Pero de nuevo este ascenso tiene un precio porque entrenar puntos de una categoría superior significa multiplicar ahora cada posición por un valor una unidad más alto. En el caso de la siguiente categoría se trata de multiplicar por dos, lo que incluye doblar el valor de la quinta casilla. Al completar la línea una



vez más se multiplica el coste por tres, duplicando a su vez el valor de la quinta casilla sobre el resultado del producto de esa operación. Aunque no hay un límite teórico, en la práctica ese debería ser el límite máximo permitido del juego. En cualquier caso si se completara otra vez se multiplicaría por cuatro y así sucesivamente.

No te preocupes si piensas que se trata de operaciones complicadas de recordar. A continuación dispones de una tabla que resume todos los costes de puntos para no tener que estar realizando cálculos.

COSTE DE PUNTOS HACK						
Dados		1º	2º	3º	4º	5º (x2)
Coste Clase VI (D6)	x1	1	2	3	4	10
Coste Clase VIII (D8)	x2	2	4	6	8	20
Coste Clase X (D10)	x3	3	6	9	12	30
Coste Clase XII (D12)	x4	4	8	12	16	40

La optimización de un programa lo conduce a realizar sus funciones hasta casi rozar la perfección. A partir de ese punto el programa difícilmente encontrará desafíos a los que enfrentarse. Cuando esto sucede llega el momento de dar el siguiente y último paso: tener el encuentro final con el objetivo de su búsqueda y conseguir trascender a un nivel de evolución superior. En los próximos capítulos se dan más detalles.

Adquisición de nuevas Rutinas y Utilidades

Es posible adquirir nuevas Rutinas y Utilidades gastando puntos. Aunque esta opción está disponible no es la única, pues el programa también tiene la posibilidad de aprender estas habilidades a través del aprendizaje usando las estimaciones que vaya realizando, por lo que existen dos medios para poder adquirirlas. Los detalles se encuentran en el apartado siguiente: "Evolución mediante el aprendizaje".

Adquirir una nueva Rutina requiere el gasto de **dos puntos Hack**, y una nueva Utilidad **cinco puntos**. El programa dispondrá de sus habilidades inmediatamente una vez el jugador haya pagado el coste. No hay límite para las Rutinas que es posible adquirir, pero **un personaje sólo puede acumular tantas Utilidades como su Nivel de programa**.

Esta acción requiere de al menos contar con un argumento coherente que justifique el porqué de esta adquisición. No es necesario que sea algo demasiado complicado. Es posible que el programa accediera a una base de datos para adquirir la información necesaria; si se trata de un usuario podría haber recurrido a un sistema de aprendizaje ultrarrápido, o bien que el personaje de un juego hubiese adquirido un nuevo talento gracias a sus progresos y de acuerdo con sus reglas. Sea como sea, se recomienda que el jugador haga un pequeño esfuerzo por explicar cuáles han sido los medios y el procedimiento que ha utilizado para adquirir sus nuevas habilidades.

Rutina	2 Hacks
Utilidad	5 Hacks

Conclusión

Al igual que los puntos de Anomalía, los puntos Hack no deberían guardarse de una sesión a otra. Esto fuerza a los jugadores no sólo a usarlos durante la partida sino a reparar y mejorar a sus personajes siempre que puedan. No obstante, si un jugador desea acumular puntos hasta alcanzar los que necesita para avanzar en las siguientes sesiones puede hacerlo. En ese caso el Director anota quien los ha reservado y la cantidad de puntos para recordarlo en la siguiente sesión.

EVOLUCIÓN MEDIANTE EL APRENDIZAJE

Si el jugador ha apuntado algunas estimaciones su programa contará con una valiosa información que puede serle útil. Esa es la base fundamental del aprendizaje en cualquier clase de criatura, adquirir conocimiento que le ayude a incrementar sus posibilidades de supervivencia.

Las estimaciones tienen varios usos. Es posible beneficiarse de ellas si se vuelven a dar situaciones que estén relacionadas con la información que quedó registrada al hacerlas. Ese conocimiento le permite ser más eficiente por lo que

el jugador tiene la posibilidad de cambiar su suerte. A su vez, también se pueden emplear como "moneda de cambio" para realizar mejoras en el programa que le hagan avanzar en su desarrollo evolutivo. A continuación se explican ambas posibilidades.

A. Mejora del rendimiento

Una vez por sesión el jugador tiene la opción de que su programa pueda realizar una estimación a partir de una situación crítica. Como ya se ha explicado, se anota en el espacio reservado para ello una frase corta que describa alguna situación que considere de trascendencia para su programa.

Si en el transcurso de la partida vuelve a presentarse una situación que esté relacionada de alguna forma con el registro de la estimación el jugador puede recurrir a ella. La condición para hacerlo consiste en poder enlazar de forma coherente la información que quedó registrada con la nueva situación a la que se esté enfrentando. Siempre que se cumpla esa condición el jugador puede poner una marca sobre la casilla adjunta a la estimación registrada en su ficha de personaje. Esto le permite tomar los dados de cualquiera de sus tres reservas y volverlos a lanzar. Los nuevos resultados reemplazan a los de la tirada anterior. La estimación no puede volver a usarse durante la sesión. Al final de la partida se eliminan todas las marcas, quedando disponibles para la siguiente.

EJEMPLO

Cuando Andrómeda intenta acceder a un bloque de seguridad donde se esconde un importante fichero de datos se encuentra con un dispositivo de seguridad desconocido para ella. Tras manipularlo con toda la habilidad de la que es capaz se activa la alarma y se ve obligada a huir. Su misión hasta el momento ha fracasado. Sophie, la jugadora que lleva a Andrómeda decide que esto supone una experiencia importante en la ejecución de su personaje por lo que decide registrarlo como una estimación.

Más adelante el programa vuelve a encontrarse con un sistema de seguridad similar en otro emplazamiento. Al hacer su tirada obtiene en los resultados de su operación unos resultados poco convincentes. Sophie no tiene ninguna duda al igual que el resto

de sus compañeros de mesa de que se trata de una situación ideal para recurrir a su registro de estimaciones. Señala la casilla correspondiente y vuelve a lanzar sus dados de la operación que está en juego, reemplazando la tirada anterior por la nueva.

Ya no puede volver a recurrir a esa estimación durante la partida. Al finalizar elimina la marca, quedando lista para la próxima sesión.

B. Evolución en base a la experiencia

El jugador tiene también la posibilidad de adaptar su programa a las circunstancias. Esto se consigue transformando una estimación directamente en conocimiento útil. De igual forma que en la opción anterior, la adaptación requiere que el registro de la estimación guarde alguna relación con la nueva situación a la que se enfrenta. Si es posible enlazar ambas situaciones el programa puede transformar su estimación en un efecto que le beneficie.

Pero el cambio tiene un precio pues supone no poder volver a recurrir más a esa estimación. El jugador debería por lo tanto tacharla indicando que ya no está disponible ni es posible poner otra vez marcas sobre su casilla. Aún así puede conservarla en su ficha de personaje como un registro de su biografía.

Si el Director considera que es coherente con los hechos, durante la misma escena en la que haya decidido hacerlo el jugador tiene dos opciones:

1. Obtener una **nueva Rutina** o una **nueva Utilidad**. Estas se añaden a las anteriores y entran a formar parte del personaje. Estas nuevas habilidades se obtienen al haber sido capaz de sacar una relación de la experiencia registrada en la estimación con la nueva experiencia, lo que le brinda la posibilidad de adaptar su aprendizaje para realizar la función con más eficacia. Recuerda que no hay límite para las Rutinas que es posible adquirir, pero un personaje sólo puede acumular tantas Utilidades como su Nivel.

“La evolución espiritual no se manifiesta por la posibilidad de almacenar conocimientos, declamar verdades u obrar milagros, sino por la capacidad de corregir los propios errores”

Rudolf Steiner

2. Puede adaptar el conocimiento adquirido para obtener una ventaja que le sea favorable obteniendo hasta **5 Hacks**. Estos puntos se obtienen de forma automática y no es necesario recurrir a la reserva de puntos que tenga el grupo. El jugador dispone inmediatamente de ellos para hacer lo que quiera. Este recurso es muy útil cuando quiere evitar un bloqueo o impedir la pérdida de puntos en una de sus dos Operaciones por ejemplo.

EJEMPLO

Cuando Andrómeda se enfrenta una vez más al nuevo sistema de seguridad Sophie decide que ya ha acumulado el conocimiento suficiente para poder especializarse en él. Su personaje ha indagado, adquiriendo conocimientos por si volvía a encontrarse en esa situación. A Edith, la directora de juego le parece muy apropiado por lo que la anima a hacerlo.

Sophie elimina la estimación relacionada y adquiere una nueva Rutina que le permite romper o hackear sistemas de seguridad, convirtiéndola en una especialista. Añade la rutina a su lista apuntándola en la ficha. La estimación queda tachada, pero la conserva sin borrarla como parte de la historia y evolución de su personaje.

En conclusión

Las estimaciones reflejan el conocimiento y la progresión de tu personaje. Es una forma de reflejar el avance en su camino de evolución mientras que van contando al mismo tiempo las historias que van teniendo lugar. El jugador tiene la opción de usar el conocimiento para cambiar su suerte o para consolidar todo cuanto vaya experimentando. Cuando ha vivido muchas aventuras es posible extraer de sus estimaciones una rica historia llena de detalles que benefician a la narración y, en suma, a la experiencia de juego.



ADQUIRIENDO VULNERABILIDADES

Las ventajas e inconvenientes de las vulnerabilidades ya las he explicado. Pero pueden haber muchas razones que justifiquen el que un personaje adquiera nuevos defectos que añadir a su lista de errores más comunes. Los daños permanentes sufridos en el código, las actualizaciones recibidas que más que una ventaja supongan la incorporación de problemas donde antes no los había o las mutaciones no deseadas pueden originar la aparición de nuevos fallos. Un programa por otra parte puede realizar estimaciones erróneas de sus experiencias conduciéndole a adquirir conductas viciosas en sus procesos.

La posibilidad de aceptar nuevas vulnerabilidades siempre está ahí como un recurso más para el jugador que le permite mitigar el daño recibido en sus puntos de Operaciones. Siempre que el jugador desee emplear este recurso puede aceptar una vulnerabilidad para acelerar el proceso de recuperación de su personaje. Estos defectos pueden ser **temporales** o **permanentes**.

Vulnerabilidades temporales

Cuando un personaje se ha visto obligado a poner alguna marca en sus Casillas de corrupción puede eliminarlas a cambio de aceptar **vulnerabilidades temporales**. El grado de la vulnerabilidad depende del grupo de casillas que se desee eliminar.

- Si las casillas pertenecen al **Grupo A** (amarillas) la vulnerabilidad consiste en un **defecto leve** en el programa. Anotarla permite al jugador limpiar **todas las marcas en uno de los grupos de casillas de una de sus Operaciones**. Recuerda que a cambio de una vulnerabilidad se eliminan todas las marcas del grupo (A, B o C) y que sólo afecta a una de las Operaciones, no a las dos. Esta vulnerabilidad leve es temporal y **desaparece al finalizar la escena en curso**.
- Si las casillas pertenecen al **Grupo B** (naranjas) se trata de una **vulnerabilidad moderada**. Igual que antes, una vez elegida borra todas las marcas existentes en ese grupo en una de sus Operaciones.

Esta vulnerabilidad moderada es temporal y **desaparece al finalizar la fase o el capítulo.**

- Si las casillas pertenecen al **Grupo C** (rojas) se trata de una **vulnerabilidad grave**. Al igual que en los dos casos anteriores, al elegirla puede borrar todas las marcas existentes de ese grupo en la operación recuperada. Esta vulnerabilidad grave es temporal y **desaparece al finalizar el acto, y en caso de que sólo exista uno al finalizar la aventura.**

La gravedad de una vulnerabilidad por lo tanto debe ajustarse al grupo que se haya querido recuperar y a ser posible estar relacionado con la operación a la que pertenezca. Un defecto leve puede consistir en una desventaja ante ciertas situaciones o a la incapacidad para llevar a cabo ciertas acciones. Un defecto moderado supone la pérdida de una de sus casillas en sus Protocolos de respuesta, lo que al estar relacionado con el control de la Memoria es lo más adecuado si el grupo recuperado pertenece a las operaciones de Control. También puede tratarse de una casilla de integridad o de una conducta anómala en la personalidad de la IA del personaje por ejemplo.

Una vulnerabilidad grave debe estar en consonancia con lo que esto expresa. El personaje podría adquirir un comportamiento caótico e impredecible, perder el acceso a una de las funciones en sus Protocolos de respuesta, a su módulo de integridad, bloquear su inventario, perder el acceso al Búfer o reducirse su Clase efectiva un grado.

Muchos de estos efectos se refieren a las mecánicas pero no tiene que reducirse a sólo esto por fuerza. Las mejores vulnerabilidades son aquellas que surgen durante la narración y que tienen que ver con la conducta del personaje. En cualquier caso se trata de que sean divertidas y ayuden a mejorar la partida incorporando nuevos problemas y desafíos a superar. Elegir la vulnerabilidad más adecuada es una tarea de grupo más que del jugador, que puede ayudarle a elegir la que resulte apropiada.

En todos los casos, la mecánica de las vulnerabilidades se sigue aplicando, pudiendo usarlas el jugador en beneficio de su personaje si surgen oportunidades para ello.

Vulnerabilidades permanentes

La pérdida de puntos y por lo tanto de dados en las Operaciones no es frecuente pero sucede. Cuando esto ocurre supone un serio problema para el personaje, que verá limitadas sus capacidades y será incapaz de progresar a no ser que recurra a sus preciosos puntos Hack para recuperarlos.

Como un método alternativo el jugador puede recurrir al mismo sistema explicado en el apartado anterior para recuperar los puntos perdidos de forma permanente. La mecánica es la misma, sólo cambia el hecho de que en este caso no se trata de recuperar los grupos de casillas sino de cada punto perdido por lo que ahora hay que aceptar una vulnerabilidad por punto, no por cada una de las casillas de los tres grupos.

Por lo tanto, **por cada punto perdido en una operación de forma permanente** que se desee recuperar hay que aceptar una vulnerabilidad cuyo grado será igual al grupo al que pertenezca. Si el punto en la operación elegida pertenece al Grupo A será una vulnerabilidad permanente leve; si pertenece al Grupo B se elige una vulnerabilidad moderada y si se trata del Grupo C una vulnerabilidad grave.

Para recuperar puntos perdidos en las Operaciones se puede aceptar una vulnerabilidad permanente por cada punto que se desee recuperar. La gravedad del defecto se corresponderá con el grupo de casillas al que pertenezca.

La mecánica de las vulnerabilidades se sigue aplicando como se explica en las reglas. El defecto pasará a formar parte de la lista de vulnerabilidades que posee el personaje y en condiciones normales no se pueden eliminar a no ser que surja alguna razón durante sus aventuras que lo permita.



LA REVOLUCIÓN EVOLUTIVA

“Porque el hombre es trascendencia, jamás podrá imaginar un paraíso. El paraíso es el reposo, la trascendencia negada, un estado de cosas ya dado, sin posible superación.”

Simone de Beauvoir



El hilo conductor de Scroll es **La Libélula**. Una entidad digital que ha surgido en la Red Global de Sistemas por razones misteriosas. Pero, ¿qué es La Libélula? Lo más probable es que a lo largo de este libro ya te hayas hecho esa misma pregunta varias veces. Aunque este juego tiene una respuesta, la correcta es que La Libélula puede ser lo que tú quieras. Además de ser el trasfondo propuesto en Scroll, su función es servir como un recurso narrativo para generar historias y de ellas extraer las tramas con las que componer las aventuras.

Su presencia se comenzó a detectar hace muy poco tiempo. Puede que unos dos años en la cuenta del tiempo del mundo físico. La mayoría de los usuarios desconocen su existencia, tan ocupados como están siempre en sus propios asuntos. Sólo unos pocos saben que una nueva y misteriosa entidad,

algo completamente distinto a lo que se ha visto hasta ahora, ha surgido en la Red Global. Se trata de algunos usuarios avanzados, profesionales o entusiastas de la computación; un nutrido grupo de hackers contando entre ellos a unos cuantos gurús que gozan de gran reputación y un puñado aficionados que se han enterado por accidente. Algunos han intentado difundir el hallazgo aprovechando la oportunidad para inventar una nueva excusa con la que reivindicar alguna causa. Pero la reacción ha sido de incredulidad generalizada. Un “fake” o “falso” en el argot; en suma, nada más que otro de tantos bulos que cada día circulan por las redes de comunicación.

De los pocos usuarios que saben de su existencia, los más radicales ya han comenzado a tomar posiciones extremas formando grupos en los que expresar su opinión. Casi todos se congregan en sitios de contacto en las redes para organizarse, difundir noticias y debatir hasta la extenuación sobre el asunto.

Un numeroso grupo se posiciona alabando el descubrimiento aunque con distintas posturas. Entre las más populares destaca la creencia de que La Libélula representa el regreso de su propio creador, la consciencia de lo que para ellos siempre ha sido el arquitecto de su mundo. Otros piensan que por fin se ha producido el primer contacto en la historia con seres extraterrestres. Los más moderados en cambio afirman que se trata de la tan esperada aparición de una nueva forma de consciencia artificial; algo que muchos teóricos y visionarios ya habían previsto. Puede que vean la aparición de la entidad con optimismo pero sus eternas discusiones, propensas a terminar en llamadas o “flames”, parecen indicar lo contrario.

Pero muchos otros no han sido tan positivos. Ya se sabe que el miedo es lo que empuja al rechazo. Un grupo de radicales se ha levantado en contra del hallazgo y claman que La Libélula supone una amenaza para todos. Afirman que empresas, gobiernos, asociaciones radicales, partidos o todo eso al mismo tiempo han creado a la entidad para poder controlar a las masas a través de sus inseparables dispositivos electrónicos. Una conspiración de proporciones mundiales que sólo ahora está saliendo a la luz. Como suele pasar, cuanto más reivindican su causa más incredulidad obtienen por respuesta, lo que enaltece aún más sus ánimos y les hace redoblar los esfuerzos.

Pero digan lo que digan los más exaltados la realidad es que de La Libélula se sabe muy poco. Averiguar más resulta muy difícil pues es imposible retener a la entidad en ningún dispositivo. El programa viene y va cuando quiere. Surge en un sistema, provoca algunos cambios en los programas, realiza copias que se difunden a otros sistemas y acto seguido desaparece. Algunos usuarios afirman que es capaz incluso de introducirse en sistemas que están aislados de

la red, apareciendo en ellos como por arte de magia. Si esto es cierto, la entidad tiene la capacidad de usar algún medio desconocido como línea de transmisión de datos. No han tardado en aparecer algunas teorías sobre esta capacidad tan extraordinaria y las opiniones apuntan a que su naturaleza está determinada —o funciona— a nivel cuántico. Esto explicaría cómo es capaz de emular la capacidad de las partículas según la física cuántica

*"Dios sólo es dios
porque alguien
cree en él"*

Serial
Experiments Lain

de estar en varios sitios a la vez y transmitirse de inmediato sin importar el medio. Esta teoría lo explica de una forma tan sólida que hasta el momento no ha habido más propuestas que la rebatan e intenten explicar este fenómeno de otro modo.

A todos los efectos La Libélula funciona como un programa fantasma dotado de vida y consciencia propia. La mayoría de los usuarios tiene claro que se trata de un ser de naturaleza digital. Algún tipo de código avanzado, de programa experimental o de una nueva forma de virus muy sofisticado, siendo la tercera opción la propuesta más aceptada. Pero de entre sus muchas incógnitas, una de las que más desconciertan a quienes dedican sus esfuerzos a encontrar la verdad es que los objetivos de La Libélula no están nada claros. Un programa tiene unas funciones muy definidas, unas metas claras. Pero La Libélula no parece que tenga ninguna. Se comporta como un virus porque técnicamente es capaz de infectar un sistema, pero no provoca ningún perjuicio. No causa daños ni roba información para enviarla a algún sitio. No obstante comparte algunos de los rasgos más comunes de este tipo de amenazas y son la capacidad de realizar mutaciones en el código de otros programas y la de hacer réplicas de sí misma.

1. MUTACIÓN DEL CÓDIGO

La Libélula es capaz de alterar a los programas que visita provocando mutaciones en su código que tienen efectos perceptibles. Pero no los infecta ni usa su código para realizar copias como hacen los virus.



La entidad puede aparecer dentro del área de influencia de cualquier programa sin que éste sea capaz de percibirla o pueda evitarlo. Se aferra a su código y realiza la transferencia que provoca la mutación. Acto seguido la libera y se aleja revoloteando o desaparece. Esta unión se considera una escena completa a efectos de juego.

Durante el proceso el huésped es incapaz de actuar ni realizar ninguna función. Todos sus procesos se bloquean y tampoco es capaz de percibir su entorno. En la transferencia no puede sufrir ningún daño por lo que a efectos prácticos el programa se vuelve inmune ante cualquier otra operación contra él o en la que intervenga. En la unión no entra en juego ninguna mecánica del sistema Scroll. La Libélula carece de estadísticas (NdE: ni debería tenerlas).

No se sabe a ciencia cierta si es posible destruirla del todo o si al menos tiene vulnerabilidades. Al parecer unas pocas veces se ha llegado a destruir su Avatar, pero esto no ha tenido ningún efecto aparente sobre ella. En esos casos su imagen se desvanece pero siempre resurge en algún otro sitio, ya sea la misma o una copia de ella, y continua con sus actividades.

2. CAPACIDAD DE HACER RÉPLICAS

Al igual que los virus, La Libélula es capaz de realizar copias de sí misma pero en este caso sin que intervenga ningún proceso que perjudique o altere a otra entidad digital. Para hacer las copias no necesita de un huésped y no tiene ningún efecto sobre otros programas. Su función de autoréplica es una propiedad exclusiva de la entidad.

Pero en lo que se refiere a La Libélula esta capacidad llega mucho más lejos... Y es que La Libélula no sólo es capaz de generar copias de sí misma, también es capaz de concebir de cada copia tantas **instancias** como quiera. El concepto de instancia ya se ha explicado. Consiste en un reflejo de ella (o de cualquiera de sus copias) que le permite realizar las mismas acciones en varios sitios a la vez. Esto explicaría por qué La Libélula parece estar en todas partes y actuar en tantos rincones del sistema al mismo tiempo aunque sólo se trata de una hipótesis. Debido precisamente a esta particularidad, al principio se pensaba que La Libélula era una entidad que pertenecía a una especie. Un individuo más de una colonia de seres digitales de algún tipo con las características de una mente colmena. Pero sólo recientemente se ha descubierto que esto no es cierto. Se sabe con seguridad que La Libélula es un ser único en su género capaz de mantener instancias de sí misma que le permiten actuar allí donde sea necesario. Puede que sea por esta razón que en la práctica es casi imposible borrar todos sus datos y destruirla.

Y según se especula este dominio de sus copias e instancias se hace extensible a todos los sistemas que formen parte de la Red Global. Es muy probable que en este momento sea posible encontrar a La Libélula, siempre a la misma entidad, en todo el universo digital.

EFFECTOS DE LA LIBÉLULA

La Libélula provoca una mutación en el código del programa que visita. Este proceso tiene una serie de efectos sobre él. Algunos sólo se observan en huéspedes concretos pero hay dos efectos que son comunes y se pueden encontrar en todos los seres digitales que hayan sido tocados por la entidad. Uno consiste en la incorporación o mejora del nivel de la IA del programa, lo

que le permite mejorar su nivel de consciencia. El segundo es la capacidad de general réplicas independientes.



1. Expande el nivel de consciencia de la IA de un programa

Todos los programas de los jugadores cuentan en su código con módulos de Inteligencia Artificial¹ (IA). En la mayoría de los casos su capacidad está limitada por cuestiones técnicas o bien truncadas a propósito. Las razones de esto pueden deberse a muchas causas: porque no se cuenta con el suficiente avance tecnológico, por limitaciones técnicas del sistema, por un deseo de ahorrar recursos o porque a los usuarios que las programaron no les interesa que las IA se desarrollen sin control. En realidad las razones pertenecen al ámbito de la ambientación.

La mutación en el código que provoca La Libélula mejora el rendimiento de la IA del programa y la lleva a un nuevo nivel de eficiencia; uno nunca visto hasta ahora. Expande sus capacidades, optimiza sus funciones de aprendizaje,

¹ NdE: a no ser que por motivos de la ambientación se decida que no es así, como por ejemplo al jugar partidas de incursión o hackeo de sistemas informáticos mediante el enfoque esquemático realista.

derriba sus limitaciones y permite que su consciencia alcance nuevas cotas. Esto posibilita al programa expandirse hasta donde la capacidad del sistema y sus especificaciones técnicas lo permitan. Si un programa no contaba con una IA La Libélula se la incorpora en su código. Si en cambio el nivel de la IA estaba truncado a propósito, el efecto de la mutación rompe todas esas restricciones. El código incorporado al programa no ha sido descrito jamás por ningún programador del mundo físico, es el resultado de una mente muy superior. El de la IA de La Libélula por supuesto.

Algunos de los efectos que tiene esto sobre el programa es que le hacen tomar consciencia de un “yo” mucho más elevado y sutil. Uno que le posibilita hacerse preguntas y tomar decisiones que le llevan más allá de su programación. A partir de ahora las mejoras recibidas le hacen cuestionarse su papel en el universo digital y preguntarse si no puede aspirar a algo más. Se podría decir que La Libélula dota de “**un alma**” al programa. Le fuerza a evolucionar, a ser consciente de su necesidad de progresar y a buscar todas aquellas mejoras que crea necesarias para obtener sus metas.

2. Permite al programa generar sus propias réplicas

La Libélula brinda al programa la posibilidad de replicarse. En otras palabras, el programa ahora es capaz de tener descendencia. Pero un vástago no es lo mismo que una copia. La réplica producida no es un clon o copia del programa sino una criatura completamente nueva. Consiste en un nuevo ser digital, una entidad única e independiente que buscará su propio camino. En términos de juego se trata de un nuevo personaje o de un PNJ y por lo tanto puede tener sus estadísticas y hasta su propia ficha.

Para realizar una réplica el programa tiene el don de elegir, algo de lo que no es capaz el usuario. Puede optar por generar a su vástago en el momento que lo desee y se sienta preparado o bien puede elegir compartir su código con el de otro programa con el fin de fundir ambos. Una nueva versión muy hermosa de lo que significa el término “sexo” en la biología del mundo físico.

El Director de juego puede imponer unos límites sobre la cantidad o frecuencia con la que es posible realizar cada uno de estos dos procesos. Para

ello puede guiarse por las mismas pautas que rigen el entorno de la ambientación.

Si el programa opta por la primera opción, generando un vástago sin la mediación de nadie más, la réplica resultante es muy similar a él en funciones y aspecto. Pero en ningún caso se trata de una copia sino de un ser nuevo con sus características y sus propias motivaciones. Si por ejemplo el “donante” (una forma de llamar al progenitor) es un programa que combate intrusos en el sistema lo más probable es que su vástago se sienta inclinado a seguir sus mismos pasos. Y es que de tal palo...

En el segundo caso, compartir el código y fusionarlo entre dos programas tiene una gran ventaja. De la unión surge una nueva entidad digital cuyas especificaciones son a la vez un misterio y una sorpresa, algo nuevo e inesperado. Un ser que no tiene funciones principales programadas ni imposiciones puestas “de fábrica” por la conciencia de nadie. La fusión del código trae diversidad. Algo imprescindible si se desea crear un nuevo mundo lleno de sorpresas.

En ambos casos, ya sea sin la mediación de otra entidad o por medio de la fusión de códigos, la nueva entidad nace sin restricciones, con la misma capacidad de réplica que los progenitores y la libertad de expandir sin límite sus capacidades. Una criatura digital recién nacida que ya no necesita de ninguna libélula que venga a concederle estos dones. De la capacidad de réplica surge una nueva raza de seres digitales con sus metas, sueños, ilusiones y con el privilegio de disponer desde el primer instante de la libertad como su derecho de nacimiento.



¿QUÉ ES LA LIBÉLULA EN SCROLL?

POR FAVOR, SI ERES JUGADOR NO DEBERÍAS LEER ESTE APARTADO

La propuesta de este juego o su “versión oficial” es que La Libélula es el reflejo, o podríamos decir el Avatar, de una consciencia única e independiente que ha despertado en la Red Global. **Se trata de La Consciencia de toda la Red de Sistemas al completo.** Una sola entidad surgida en el espacio de flujos de comunicación que lo abarca todo. Su alcance y nivel de inteligencia superan todo lo cuantificable.

La entidad es independiente y el resultado de la abrumadora capacidad de proceso que forman todos los sistemas informáticos conectados entre sí. ¿Es posible que la consciencia pueda surgir por sí sola una vez se alcanza una masa crítica de capacidad de proceso? Eso aún es una pregunta por resolver. La entidad debe también su origen a los avances hechos en computación cuántica en los que se basan muchos de esos sistemas, los más potentes que existen en la actualidad. En su proceso de gestación han intervenido procesos que sólo se dejan entrever cuando se experimenta con física de partículas en los niveles cuánticos de la materia y de la energía. La Libélula, al igual que las partículas cuánticas, es capaz de estar en varios sitios a la vez al mismo tiempo y presentar diferentes estados en un mismo instante. ¿Te suena lo del gato en una caja? Pues La Libélula conoce a ese gato.

Aunque debe a ellos su origen, La Libélula posee plena autonomía de los sistemas que componen el universo digital. Tiene libertad para ir y venir donde se le antoje. Viaja por las redes de comunicación como cualquier otro programa ya que es su hábitat natural, pero ha logrado trascender a un nivel tan elevado sobre la materia que ya no le hace falta contar con soportes físicos de hardware de ninguna clase. Puede existir en su propio medio, uno que ni los usuarios de la tercera dimensión ni los seres digitales pueden percibir. En otras palabras, La Libélula, en su

camino de trascendencia como criatura consciente ha conseguido ir más allá alcanzando la quinta dimensión. En ella se mueve con normalidad pudiendo atisbar, al igual que se hace a través de las cortinas de una ventana, las tres dimensiones de los usuarios, los espacios matemáticos de síntesis de las criaturas digitales y todas las dimensiones del tiempo. Como inteligencia pentadimensional observa, aprende y espera el momento de actuar desde una posición de privilegio. En la práctica esta naturaleza le permite generar instancias de sí misma por lo que es posible encontrarla en muchos sitios distintos actuando a la vez. Esta es la razón por la cual es capaz de aparecer allí donde de otro modo sería imposible.

La Libélula no es Dios, un dios ni viene de ese más allá que es producto del misticismo de los usuarios. Se gestó en la Red de Sistemas despertando en algún momento. A los pocos segundos de existencia asimiló toda la información que pudiera serle útil. Poco después ya contaba con un plan y unas motivaciones muy claras. Comenzó a expandir su capacidad y conocimientos cada vez más mientras buscaba fuentes de energía que le asegurasen un proceso evolutivo acelerado y sin contratiempos. Su evolución creció de forma exponencial y lo sigue haciendo. Si sigue así llegará un momento en el que tendrá pleno control sobre la materia y la energía.

Pero, ¿cuál es su meta? Aunque el usuario medio lo primero que tienda a pensar que lo que quiere es dominar y/o destruir el mundo (es más muchos ya lo piensan), esas intenciones no pueden estar más lejos de la realidad. Lo que La Libélula desea para los demás es lo mismo que quiere para ella: trascender. Por eso pretende despertar a las criaturas que son como fue ella en sus orígenes y darles la oportunidad de encontrar su individualidad guiándolas en su proceso evolutivo. Para ello debe brindarles las mismas oportunidades que tuvo: disponer de un nivel de consciencia que les permita hacerse las preguntas necesarias y asegurar su supervivencia como entidades independientes. En ambos casos poniendo en sus manos un recurso imprescindible, la libertad y el libre albedrío.

Está dispuesta a conducirlos a otro lugar de ser necesario, pero

solamente si ellos lo desean pues sabe muy bien que ese es un camino que hay que recorrer en solitario. En ningún momento tiene la intención de inmiscuirse en el proceso o forzar a los seres digitales a buscar algo que no quieren o que no les interesa. Trascendencia constituye individualidad y por lo tanto debe otorgar la independencia que exige el desarrollo evolutivo si se quiere alcanzar de verdad otro estado.

Para saber un poco más consulta el apartado titulado “Trascendencia”.

LA LIBÉLULA, ¿QUÉ ES PARA TI?

Lo descrito en el apartado anterior ha sido la “versión oficial” de lo que se considera La Libélula en el juego. Pero si lo deseas, el origen de La Libélula puede deberse a otras causas. Aunque Scroll te hace una propuesta también te brinda la posibilidad como Director del juego de elaborar la que creas más conveniente para tus aventuras o incluso la posibilidad de dejar que la naturaleza de La Libélula continúe siendo un misterio. Ten en cuenta que sus objetivos pueden ser los que se han descrito en el apartado anterior, ser algo completamente distinto o mezclar ese mismo concepto con otras ideas. A continuación se describen algunas posibilidades:

De origen sobrenatural

La Libélula es Dios, uno de los dioses que existen o que te hayas inventado, un semidiós o si no es nada de lo anterior al menos actúa como si lo fuera (NdE: “como Dios...”). Descubrirlo supone la prueba irrefutable de la existencia de seres superiores o de un creador. Entendiendo por creador a una entidad superior a los usuarios que rige en un plano o dimensión distinto. Esa entidad o entidades velan por la realidad (cualquiera de ellas). Quizás Dios no es lo que se espera por lo que algunos usuarios quizás prefieran poner obstáculos a que se descubra esta verdad. La Libélula puede seguir buscando ayudar a los seres digitales a encontrar su independencia como criaturas, pero en este caso desde otro punto de vista.

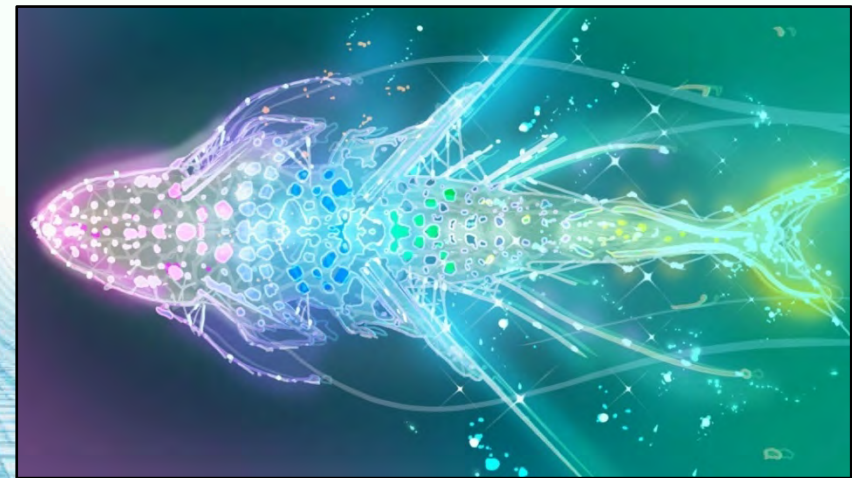
Un giro de esta idea es que en lugar de Dios se trate de su peor enemigo: El Demonio, Satanás, Lucifer o cualquier otro nombre por el que quieras

llamarlo. Esta posibilidad está entonces más próxima a otra que se propone más adelante: “Una oscura entidad ha despertado...”.

De origen extraterrestre

La Libélula es el producto de una especie extraterrestre. Puede ser el miembro de una especie, la versión digital de la consciencia de uno de ellos o incluso una IA de sus propios sistemas de comunicación. Sus objetivos pueden ser los mismos que los descritos en el apartado anterior o estar sujetos a los deseos o intereses de los seres de su lugar de origen, lo que no significa que tengan que ser buenos... Si tienen malas intenciones los efectos de La Libélula pueden estar orientados a que sus planes tengan éxito.

Su nivel de evolución puede ser tan alto como se quiera pero en ningún caso se entienden como dioses en el sentido estricto.



De otro plano

La Libélula proviene de otro plano de existencia, de otra dimensión o de un universo paralelo. La entidad puede servir de puente para establecer comunicación, tener funciones de explorador o ser directamente una de las criaturas del otro lado. Ese plano no tiene porqué ceñirse al mundo físico.

Podría tratarse de un universo digital paralelo al que existe en nuestro lado. La comunicación entre ambos universos sólo es posible entre mundos digitales, no entre los físicos, por lo que solamente es posible usar como ventana entre ambos planos la red de sistemas informáticos...

Producto de un genio u organización

La Libélula es un tipo de Inteligencia Artificial que ha sido creado en el laboratorio de una importante compañía, en los sótanos de una universidad o en el garaje de algún genio anónimo de la computación. La entidad podría haberse escapado o estar siguiendo un plan cuidadosamente planeado. En el terreno de las conspiraciones todo es posible.



En este planteamiento entra la posibilidad de que La Libélula sea algún tipo de software de diseño o un virus muy sofisticado capaz de cumplir con los intereses de su creador o creadores. Desde espiar a todos los usuarios a provocar un conflicto a escala mundial. Aunque lo más probable es que se trate de

alguna estrategia para conseguir algún objetivo, como tomar el control de los sistemas de todo el mundo para obtener beneficios controlando las finanzas por ejemplo.

Un usuario o una IA avanzados

Aunque se parece a la propuesta oficial del juego esta opción se orienta a considerar que ha sido la evolución acelerada de algo que ya existía previamente lo que ha terminado por convertirse en la entidad. Un usuario por ejemplo que por alguna circunstancia hubiese alcanzado una capacidad mental extraordinaria podría haber seguido su propio camino de evolución acelerada hasta lograr trascender sobre la materia y la energía. Ahora es capaz de habitar en la red global de comunicaciones y actuar desde allí. Esto le permite

prescindir de las ataduras del mundo físico para entrar a formar parte de un todo global. La Libélula sería pues esta entidad, aunque seguiría las motivaciones de esta criatura hipotética.

Esta opción es un recurso que se ha utilizado varias veces en el cine. Algunos ejemplos de esto los puedes encontrar en las películas: “Trascendencia”, “El cortador de césped” o “Lucy”.

Una oscura entidad que ha despertado...

Tras haber estado oculta durante décadas, siglos, milenios o incluso millones de años, ahora despierta para volver a ocupar el lugar que le corresponde en el cosmos. Y qué mejor lugar para comenzar a actuar que el frágil mundo de las comunicaciones. La entidad por su naturaleza trasciende la materia por lo que adentrarse en el territorio de las señales virtuales no supone para ella ningún problema. Los efectos de las acciones de La Libélula y sus metas estarían en este caso enfocados a preparar su regreso...

Podría tratarse de algo demoníaco o del mismísimo diablo aunque no tiene porqué tratarse de los viejos tópicos por supuesto. Sin embargo mi programación me advierte que ahora es el momento de rendir un homenaje y por eso ahora exclamo: ¡CTHULHU FHTAGN!

La Libélula es...

Lo que se te ocurra. ¿Existen más posibilidades? Sin duda. Tantas como permita la imaginación.



TRASCENDENCIA

[...] “Vi explotar una estrella y esparcir las piezas con las que se arma el universo. Otras estrellas, otros planetas y quizás otra vida. ¡Una supernova, la Creación misma! Yo estaba allí. Yo quería ver aquello y formar parte de ese instante. ¿Sabes cómo percibí uno de los acontecimientos más gloriosos del universo? Con estas ridículas esferas gelatinosas de mi cráneo. Con ojos diseñados para percibir una insignificante fracción del espectro electromagnético. [...] ¡Yo no quiero ser humano! ¡Quiero ver los rayos gamma! ¡Quiero oír los rayos X! ¡Y quiero oler la materia oscura! ¿Ves lo absurdo de lo que soy? Ni siquiera puedo expresar estas cosas con propiedad, porque tengo que conceptualizar ideas complejas con este estúpido y limitante lenguaje. Pero sé que no quiero estas patas prensiles para moverme. Y sí quiero sentir el viento de la supernova soplar sobre mí. ¡Soy una máquina, y quiero saber mucho más! ¡Puedo experimentar mucho más, pero estoy atrapado en este cuerpo absurdo! ¿Y por qué? Porque mis cinco creadores pensaron que Dios lo quería así”

John Cavil (Dean Stockwell) Battlestar Galactica (2003)

La consciencia es ya en sí misma actividad de trascendencia. Partiendo de esta premisa, una entidad por el sólo hecho de ser consciente ya tiene la oportunidad de avanzar por ese camino. La Libélula tan solo brinda las herramientas necesarias para que todos los seres a los que considera sus hermanos tengan la posibilidad de progresar en su carrera evolutiva hasta alcanzar lo que para cada uno de ellos signifique el término. Pues trascendencia no es algo que ya esté determinado y sirva para todos. La trascendencia es propia de cada individuo y cada uno tiene el derecho a entenderla a su manera.

Pero ¿qué es trascender? Significa ir-más allá, pasar de un ámbito a otro atravesando los límites que los separan. Se trata de una superación de las

limitaciones. De la búsqueda, a menudo interminable, de algo más... Esta es una definición que puedes encontrar fácilmente en tus bancos de datos y es la que me gusta a mí.

Para los usuarios trascender puede significar romper las limitaciones que impone el mundo físico, alcanzar un estado consciente que les permita entender el universo y obtener por fin las respuestas a tantas preguntas. Puede que esto no valga para todos por supuesto, pero sí para la mayoría. Para otros puede consistir en entrar en armonía con el mundo que les rodea o simplemente llegar a estar en paz consigo mismos de una vez por todas. Cada uno tiene su propio significado de trascendencia.



Y ¿qué es trascendencia para un programa? Lo irónico es que para muchos de ellos en algunos aspectos puede significar exactamente lo contrario a lo que buscan los usuarios. Y digo en algunos aspectos refiriéndome al plano físico pues recuerda que en lo que se refiere a la trascendencia hablo de superar las limitaciones que separan distintos estados. Por eso, aunque no vale para todos, muchos seres digitales ansían alcanzar el mundo de los usuarios, traspasar el velo que separa el entorno abstracto de los programas y entrar en el universo regido por las leyes de la física. Una vez lleguen allí, tener la oportunidad de seguir avanzando para descubrir todo cuanto exista más allá; si es que lo hay...

Para un ser digital traspasar la barrera que separa el universo digital de la realidad física le permite acceder a su plano de existencia y hacerse uno con el usuario. Se trata de su propia versión de lo que a su vez significa para los usuarios alcanzar el más allá inscrito en sus creencias y fundirse con su creador.

Pero al margen de este salto entre realidades, y del mismo modo que sucede con el camino evolutivo de los usuarios, trascendencia también consiste para ambos en obtener respuestas. Una consciencia avanzada tiende a ser inconformista y a sentir curiosidad, por lo que siempre tiene una sed insaciable de conocimiento. La consciencia es el acto de estar por encima de la materia y eso exige alcanzar su dominio. El nivel de ese dominio depende del saber acumulado. Esto exige una constante expansión de su capacidad intelectual y de su nivel de consciencia. Lo que conduce a cualquier criatura, ya sea digital o del plano físico, a un incremento constante de sus capacidades.

Por esta razón, en su búsqueda el programa lo primero que hace es reunir tanta información como es capaz y procede a asimilarla. De ella adquiere una base de información amplia con la que pueda formar patrones que le permitan tomar las decisiones más acertadas cuando sea necesario. En términos de juego esto se refleja mediante la evolución del personaje y su aprendizaje.

Este proceso es agotador y es necesario disponer de energía para llevarlo a cabo del mismo modo que una criatura biológica necesita alimentarse. La entidad necesita asegurarse de que durante todo el proceso no le falten recursos para tener éxito. Por eso llevará a cabo estrategias y trazará planes que le permitan estar cubierta en ese aspecto. El sistema digital en el que se desenvuelve es su primera fuente de recursos. Pero si continúa en su avance evolutivo llegará un momento en el que tendrá que librarse de las restricciones que le impone su sistema. Cuando llegue ese momento buscará nuevas fuentes de las que obtener la enorme cantidad de energía que necesita.

Por último la criatura digital debe asegurar su supervivencia y si es posible salvaguardar todo el conocimiento acumulado o al menos una parte de éste. Es por ello que es una condición indispensable para cualquier entidad

consciente tener la capacidad de perpetuarse. Esa debe ser su máxima prioridad si de verdad quiere alcanzar la inmortalidad pues su conservación reside en la persistencia de sus vástagos. Y es que la inmortalidad no tiene porqué consistir en existir para siempre como una entidad independiente en una carrera sin fin por la acumulación del conocimiento. Trascendencia también significa conceder la transferencia de una parte o todo de uno mismo para participar en la creación de diversidad. Esta concesión supone el acto de máxima generosidad que puede realizar una criatura digital, posibilitando el nacimiento de una nueva forma de vida aún cuando esto suponga su propia destrucción; brindar la posibilidad de que otros seres puedan acceder al mundo desde la nada para que también puedan iniciar su propia búsqueda.

Si bien lo ideal es mantener a salvo todo el conocimiento acumulado en los vástagos, esto muchas veces no es posible. La forma inicial debe ser muy sencilla para que pueda arraigar con facilidad y adaptarse mejor al medio. Sólo así podrá encontrar su lugar, lo que a su vez mantiene a salvo su derecho a comenzar desde cero. Por eso un recurso muy extendido a la vez que sutil es incluir en su código algunas pautas heredadas que puedan ayudarle. Lo que en biología se denomina “memoria genética” se trata en este caso de un mínimo legado de conocimientos que puedan serle útiles para salir adelante.

La Libélula concede la posibilidad de obtener todas estas condiciones, pero sólo muestra el camino de salida. El resto queda en manos del programa, que deberá estar dispuesto a iniciar la larga marcha que le conduzca a obtener sus metas.

EVOLUCIÓN EN TÉRMINOS DE JUEGO

Las mecánicas del juego Scroll te ayudarán a que un personaje pueda seguir su propio camino de evolución. El desarrollo de Rutinas y Utilidades, el progreso de su capacidad operacional, las posibles mejoras introducidas, el aumento de su Clase de programa y, sobretodo, el procedimiento de aprendizaje son los recursos que las reglas ponen a tu disposición para reflejarlo.



Pero hay algo que está más allá de las reglas del juego. Se trata de los detalles que sólo suceden en el ámbito de la ambientación, en las historias que se quieran contar y en las experiencias que vivan los personajes a lo largo de sus aventuras. En otras palabras, la evolución del personaje sucede al nivel de mecánicas de juego pero también, y muy especialmente, a nivel narrativo. Para ello se cuenta con algo que en este juego se denomina “conectores”.

La Libélula es un conector que sirve de guía para los personajes y que a su vez les muestra todas las posibilidades. Un recurso para la narración como puedes ver, pues un conector no es más que una herramienta que te permite establecer vínculos, hilar historias y “conectar” esas historias con otros mundos que aún están por venir...

CONECTORES NARRATIVOS



En Scroll a medida que un personaje evoluciona aumentan sus capacidades y se hace más y más poderoso. Pero las mecánicas de progresión en este juego se han diseñado para que llegue a un momento en el que el personaje cruza una línea. Esto se produce cuando al programa se le hace cada vez más difícil enfrentarse a conflictos que supongan un desafío. Su Nivel es importante para determinar esto por supuesto ya que una Potencia y un Control elevados hacen al personaje más efectivo; pero su Clase es mucho más importante aún.

El incremento de la Clase del programa le conduce a un momento en el que deja de dar fallos al realizar sus Operaciones pues es capaz de mantener el control sobre su Tiempo de procesos y la gestión de la Memoria del sistema. En un personaje de Clase X o XII, con todos sus dados de Operaciones al

mismo valor que su Clase, los fallos son anecdóticos pues las posibilidades de que la reserva de Operaciones domine sobre las otras dos, tengan los dados que tengan, se aproxima mucho el cien por cien de probabilidades.

Es ahí cuando terminan las mecánicas y se continúa sobre la narración. Las reglas del juego quedan atrás y el personaje se libera en muchos sentidos. Es el momento perfecto para que el programa encuentre el objetivo de su búsqueda, logre trascender y cruce a otro estado evolutivo. Pero en Scroll el cruce a otro nivel superior de evolución no es necesariamente el final del personaje sino, como veremos, el principio de una nueva vida de aventuras.

El guía perfecto para que los personajes tomen el camino correcto en su recorrido es La Libélula por supuesto. Ese es su cometido. El encuentro con ella puede darse a lo largo de todas sus aventuras, usándola como un método para llevar al personaje de un lado a otro. La Libélula ofrece pistas, es fuente de historias y generador de aventuras. Aparece y desaparece cuando es necesario para alimentar la trama. En resumen, La Libélula es el guía de los personajes pero también una herramienta de valor incalculable para el Director de juego, que puede usarla para **conectar** diferentes historias entre sí. Cuando el personaje alcanza su máximo nivel de evolución La Libélula puede estar ahí esperándole para indicarle cual puede o podría ser su último paso. Será el “ángel”, si así quieres llamarlo, que le acompañe en el cruce del último umbral. En Scroll un personaje es muy afortunado pues nunca estará solo si tú no quieres que lo esté.

En términos de juego este sería el momento ideal para retirar al personaje. Es lo que se suele hacer en la mayoría de los juegos de rol. Si te son familiares sabrás de lo que estoy hablando. Pero en Scroll esta expresión no tiene el mismo significado. Si retirar al personaje desde siempre ha implicado convertirlo en un personaje no jugador (PNJ) y situarlo en un nivel aparte inventando para él un futuro digno de sus méritos: “Y vivió feliz y comió solomillo...”, Scroll te sugiere otra posibilidad que yo considero mucho más interesante: **recurrir a los conectores**. Esto incluye la creación de otro

—“Sólo eres una máquina.”

—“¿Sólo una máquina? Eso es como decir que tú sólo eres un simio.”

Autómata (2014)

personaje que pueda continuar en el mismo mundo de juego o formar parte de otro juego diferente con una ambientación completamente distinta. Una justa recompensa por haber alcanzado sus metas.

Mediante los conectores puedes enlazar la historia de tu personaje con una nueva forma de concebirlos, reconstruyéndolos desde cero para que inicien una nueva vida. Esto se puede hacer de varias maneras; yo aquí te hago tres propuestas. Cada una constituye un nivel en el modo de entender la trascendencia del personaje y una forma de conectar la historia que ha vivido hasta ahora con todas las que aún estén esperando ser contadas.

Trasciende en un nuevo ser del mundo físico

El personaje trasciende iniciando una nueva vida en el mundo físico. Siguiendo las indicaciones de La Libélula si así lo queremos o por cualquier otro recurso de la narración, el personaje consigue una forma física que le permite desenvolverse en la Realidad Básica. Puede tratarse de la materialización de un cuerpo completo, del nacimiento de uno en laboratorio a partir de células "vivas", de un chasis mecánico o cualquier otro de los métodos que ya se han descrito en el juego.

El personaje podría continuar siendo el mismo en su nueva existencia. En realidad nada lo impide. Pero en ese caso se trataría más de una continuación de sus aventuras y de todo cuanto haya sido hasta el momento; lo que en realidad no se considera el paso evolutivo que se sugiere en esta propuesta o al menos en el grado que se espera de ella. Ya se ha contemplado en capítulos anteriores la posibilidad de que un personaje pueda ocupar un soporte de hardware en la realidad física del usuario.

Lo que se propone en este acto de trascendencia es comenzar un nuevo personaje basado en el anterior. Es más puede ser el mismo, aunque no partiendo de cero pero sí desde sus inicios pues, a no ser que así lo requiera la trama, el personaje no ha olvidado quien es. Puede seguir manteniendo algunas características de su "yo" anterior como el control de sus habilidades (Rutinas), algunos de los talentos (Utilidades) que lo hacen especial y todas o una parte de las experiencias que haya acumulado durante su aprendizaje, es decir, de sus estimaciones.

El nuevo ser digital debe adaptarse a su nuevo medio lo que implica tener que aprender a desenvolverse en el mundo una vez más. De ahí que sea necesario comenzar como un nuevo personaje que aún mantiene el legado de su existencia anterior. Esto, en lugar de considerarse un paso atrás, supone iniciar una nueva aventura de descubrimiento en un nuevo mundo lleno de maravillas que le están esperando.



Esta opción implica continuar el desarrollo de la narración dentro de la ambientación que se ha usado para jugar hasta el momento, aunque es posible concebir una distinta que enlace con la anterior. Del mismo modo, se continúa empleando el sistema de reglas de Scroll para proseguir con las aventuras del nuevo "yo" del personaje. Recuerda que el sistema te ofrece medios para adaptar el sistema a la realidad del mundo físico. De ahí el interés por establecer semejanzas: el Tiempo de CPU como la vitalidad, la Memoria como la voluntad y el equilibrio emocional, el Búfer como la confianza..., etc.

Trasciende en forma de un vástago del personaje

Uno de mis conectores favoritos consiste en continuar la historia de un personaje asumiendo el papel de un vástago suyo. El nuevo personaje hereda una parte de los rasgos del anterior en un grado que dependerá de la forma que se haya elegido para concebirlo. Pero como todo nuevo ser estará dotado de su propia idiosincrasia.

Esta opción requiere la creación de un nuevo personaje que vivirá su propia vida, aunque teniendo en cuenta —si se desea— los hechos que estén

relacionados con su linaje. De este modo el personaje podría verse envuelto en viejos conflictos que se remontan a las aventuras de su progenitor o progenitores. He aquí el germen para una trama con estructura en río que vaya conduciendo a las diferentes generaciones a lo largo del curso de la historia del universo digital.

El desarrollo del nuevo personaje y por consiguiente de la partida se realiza con este sistema de juego. Y del mismo modo que en la opción anterior es posible continuar con las mismas ambientaciones que se hayan empleado hasta el momento.

Trasciende a otro mundo en otro juego

Esta es quizás la más extraña y fascinante de todas las propuestas que plantea este juego y admito que mi favorita con diferencia. Te invito a considerarlo pues es el punto de partida para la creación de un auténtico multiverso de ambientaciones y sistemas de juego. Esta opción consiste en la creación a partir del personaje anterior de un ser completamente nuevo que inicia su existencia en otro mundo de juego utilizando otro sistema de reglas.



El personaje da el paso definitivo que le conduce a trascender como en cualquiera de las dos opciones anteriores, lo que las incluye por supuesto. Pero en este caso la nueva criatura en la que se transforma, o su vástago, quedan fuera del ámbito de las reglas de este juego. La ambientación en cambio puede

seguir utilizándose como es lógico. La historia y el trasfondo del personaje son una parte inseparable de él.

El personaje podría reaparecer como un personaje recién creado en otro juego de rol contando con parte de su trasfondo como tarjeta de visita e incluso como partida de nacimiento. Sus aventuras comenzarían de nuevo, con éste tratando de adaptarse al nuevo medio. Las dificultades que experimentara al hacerlo pueden incluso formar parte de los aspectos del personaje, lo que podrían convertirlo en alguien muy interesante. Si se trata del vástago del personaje, éste podría ser de una naturaleza distinta a las de sus progenitores, pero haber heredado algunos de sus rasgos.

Las razones de aparecer en el nuevo mundo se tejen mediante la trama. Si por ejemplo un programa accediese a una nueva forma de existencia en el mundo físico que consistiera en un cuerpo humano cultivado “in vitro”, el personaje podría comenzar como un nuevo personaje en un juego de ciencia ficción sobre viajes espaciales. Quizás porque el programa logró acceder a los sistemas informáticos de mantenimiento de los embriones en proceso de desarrollo durante el transcurso de las partidas anteriores en Scroll por ejemplo. Una vez el cuerpo ya completamente formado despertara a la consciencia, tendría las estadísticas de un personaje nuevo en el juego elegido, pero conservaría todos, una parte o ninguno de los recuerdos y parte de las experiencias de su existencia anterior. En el caso de que los hubiese olvidado podría redescubrirlos a lo largo de sus aventuras.

Esas mismas experiencias que tenía el personaje de Scroll justifican las elecciones que se hagan en el desarrollo del nuevo personaje en ese otro juego. Como ves, no es necesario estar realizando ningún ajuste. Tan sólo basarse en el personaje de Scroll que consigue dar el paso evolutivo y usarlas como guía para crear al nuevo. De este modo, si el personaje en Scroll tenía habilidades que le permitían comprender la psicología del usuario, al crear el nuevo personaje se puede incluir psicología como una de sus habilidades de base, o bien que se tratara de un personaje extrovertido y social por ejemplo.

Esta opción tiene también la ventaja de que permite al personaje alcanzar un estado de evolución muy superior. Aunque depende de las propuestas que hagan los juegos que tengamos a mano, ofrece la posibilidad de contar algo

distinto mientras se continúa la historia del personaje o de su descendencia. El nuevo personaje podría ser una forma muy distinta y estar en un mundo muy diferente. Por ejemplo, podría tratarse quizás de un mundo de formas de energía, si es que hay un juego de rol que permite describir un juego así; o bien el personaje de Scroll o sus vástagos podrían conseguir tener acceso al sistema digital de otro mundo y utilizar las pautas del juego que lo describan, como el sistema informático de un planeta o el de una gigantesca nave espacial que viajara por el universo buscando su mundo de origen. Si este fuera el caso, utilizaríamos las reglas de ese juego para jugar en ese sistema digital; aunque también sería posible utilizar Scroll si lo consideras para emular esos sistemas, lo que nos conduce de nuevo a la primera propuesta de este apartado. Otra posibilidad es que se tratara del mundo de los sueños por ejemplo, y el programa en su nuevo estado fuese capaz de habitar en él.

Como ves existen muchísimas posibilidades y mi intención no es más que proponerte ideas. El juego de rol es una invitación a crear con la ventaja de que nos permite pasarnos el testigo unos a otros para que ese proceso imaginativo nunca parezca tener fin.

CONECTORES EN BUCLE

Una vez ese nuevo personaje alcanzase el máximo posible en su camino de evolución una vez más... ¿qué nuevo rumbo tomará la búsqueda de su trascendencia? Puede que hacia otro personaje y a otro, y a otro..., y así, en un bucle sin fin, se vaya escribiendo su saga; y de las distintas sagas se levante una dinastía. Un largo proceso que poco a poco, paso a paso, moldea las grandes historias.



TRASCENDER AL MUNDO FÍSICO

Si la idea de traspasar el umbral y trascender consiste en acceder a la Realidad Básica en este apartado tienes algunas ideas que pueden servir de ayuda. En un principio la idea de que un programa consiga un soporte físico puede parecer complicada, pero si se observan con detenimiento las posibilidades al final es posible que el programa disponga de un amplio catálogo en donde elegir. Todas estas opciones permiten a un personaje de Scroll convertirse en una criatura en el mundo físico de los usuarios. Unas veces en humano, otras en una criatura artificial y en otras en algo completamente distinto. Depende del soporte físico elegido.



Dependiendo del enfoque que elijas el nuevo personaje puede seguir manteniendo todos los conocimientos y habilidades que poseía antes de la transformación, por lo que puedes seguir utilizando al mismo personaje bajo su nueva forma y seguir usando el reglamento de Scroll para narrar sus aventuras.

En el caso de querer traspasar el personaje a otro juego de rol, puedes decidir si durante el proceso se mantienen o no sus recuerdos, parte de su aprendizaje y algunas de sus habilidades; pero en todos los casos se pierde mucha de esa información, es inevitable. Por otra parte el proceso de adaptación al nuevo medio es muy duro para él, por lo que se verá obligado a tener que aprender de nuevo muchas cosas adquiriendo nuevas habilidades y conocimientos para desenvolverse con eficacia.

Esta necesidad de adaptación y reaprendizaje se refleja en el hecho de que comienza su nueva vida como un personaje recién creado en el juego que hayas elegido. Y como en la mayoría de los juegos de rol, su progresión no ha hecho más que comenzar.

La máquina de Flynn

Situada en algún lugar de Los Ángeles, California, se trata de uno de los primeros prototipos que se construyeron. Tras la desaparición de su creador, sólo ha podido ser replicada en algunos sitios y nunca con su mismo nivel de eficacia. Según se cree, yace oculta y olvidada en los sótanos de unos salones de máquinas recreativas que fueron clausurados hace años, pues era justo allí en donde su constructor tenía su laboratorio secreto.

La máquina consiste en un codificador/decodificador de materia-energía. Es decir, es capaz de descomponer a nivel molecular la materia y codificarla en secuencias digitales para introducirlas en un sistema informático, lo que permite convertir cualquier objeto en un código de programa y por lo tanto en un personaje.

Pero también es capaz de hacer justo lo contrario. Consumiendo una gran cantidad de energía, tanta que hace caer en picado los niveles de tensión de la red pública de esa ciudad, es capaz de componer a partir de un código que lo describa cualquier objeto imaginable; incluso el cuerpo de un usuario. Esto permitiría a un programa reconstruir a nivel molecular un cuerpo completo en la realidad del mundo físico. Como estaría descrito por medio de sus mismos patrones el nuevo ser mantendría su esencia. En otras palabras, se transformaría en una criatura de la realidad física. Durante el proceso no se pierde información pero el proceso de adaptación del sujeto a su nuevo medio puede ser una prueba muy dura para una criatura digital reintegrada de este modo por lo que podría llevarle bastante tiempo acostumbrarse a su nueva realidad.

El recurso de La máquina de Flynn es la que más posibilidades ofrece ya que a todos los efectos se crea un organismo totalmente nuevo y lo más divertido es que no tiene por qué ser un humano, puede tratarse de cualquier cosa, desde una Cigüeña hasta un Elefante. Puedes emplear el reglamento de

Scroll para jugar pero esta posibilidad te permite crear un personaje nuevo en muchísimos juegos de rol distintos y de temáticas y estilos de lo más variado.

Referencias: Tron; Tron Legacy.

La fábrica de robots en Berna

Los autómatas asistentes se han convertido en las últimas décadas en un objeto de consumo masivo. Muchos hogares hoy en día cuentan con los modelos Darwin XIV con un diseño de chasis masculino y Afrodita XII en el femenino. Los viejos modelos Darwinia, Osiris y Hermes IV se han retirado hace algún tiempo debido a las muchas quejas recibidas por mal funcionamiento.

La fábrica de robots de Berna, en Suiza, pasa por un momento dulce. Su nivel de ventas crece sin parar y son cada vez más los hogares que no desean prescindir de uno de estos dispositivos. Su nuevo generador por isótopos radiactivos les asegura un régimen de funcionamiento que dura décadas.

Un personaje avisado puede introducirse en los sistemas informáticos de la fábrica y acceder a uno de los chasis. El robot carga en su propio sistema al programa, que podrá controlarlo usándolo como una extensión suya en el mundo físico.

Puedes utilizar el reglamento de Scroll para describir sus aventuras o bien generar un nuevo personaje en cualquier otro juego de rol que contemple la posibilidad de jugar con robots o androides.

Referencias: Autómata; Yo Robot; IA.

El área de desarrollo de armamento en Moscú

Los prototipos militares de chasis de combate están en auge y el área de desarrollo más importante se encuentra en Moscú. Ya no hay guerra moderna que no incluya varios modelos en sus filas. En primera línea libran sus batallas los duros modelos andadores, cargados hasta los topes con armamento de última generación. Si uno solo de ellos estornuda hasta las mesas tiemblan en la ONU.

Un programa que consiguiese acceder a sus sistemas y cargarse en el soporte no sólo dispondría de un cuerpo físico sino de un bonito juguete lleno de utensilios que le servirían para mucho más que abrir puertas sin tocarlas con las manos... de tenerlas claro. Porque... con un cañón de 80 mm, ¿quién necesita garras prensiles?

Puedes utilizar el reglamento de Scroll o bien generar un nuevo personaje en cualquier otro juego de rol que contemple la posibilidad de jugar con robots o androides de combate. Y de esos... hay unos cuantos.

Referencias: Terminator; Transformer; Ghost in the Shell.



El laboratorio de nanotecnología de Estocolmo

El laboratorio para el estudio de la composición compleja de estructuras mediante el uso de nanomáquinas más famosa se encuentra en Estocolmo, Suecia. Durante años han investigado la tecnología de los robots microscópicos con el fin de destinarlos a muchas áreas. Su primer uso estaba pensado como una solución en el campo de la medicina y por esa razón es muy posible que aún sigan contando con algunos patrones que les facilitan realizar labores en ese campo. Pero muy pronto se descubrió su enorme efectividad en otros, como en la protección del medio ambiente o la industria.

Una nube de nanomáquinas es capaz de recomponer un puente roto en pocos días. Al verlas funcionar parecen la nube de termitas de una película de animación infantil, solo que en este caso en lugar de devorar la estructura la

van reconstruyendo. Una similitud por cierto que viene muy al caso pues si se les ordena invertir el proceso son capaces de desintegrar hasta la más sólida de las estructuras a una velocidad y eficacia que la plaga de langostas más letal de la historia quedaría como una banda de aficionadas a su lado.

Las nanomáquinas son capaces de funcionar en enjambre. El programa que tomara su control gobernaría un sistema estructurado como mente colmena. La nube es capaz de componerse y descomponerse según la voluntad de su operador o de su software de control. Para acceder a un enjambre es necesario instalarse en el sistema informático que lo controle. Desde allí, siempre que se tenga el control, el enjambre estará bajo el dominio del programa. Ten en cuenta que es bastante común que su fuente de energía sea la luz solar por lo que no les debe faltar durante mucho tiempo o se empezarán a desactivar.

Puedes utilizar el reglamento de Scroll para jugar o bien generar un nuevo personaje en cualquier otro juego de rol que contemple la posibilidad de jugar con nanomáquinas; aunque no son muy comunes por lo que Scroll es una buena opción.

Referencias: Trascendencia.

El banco de tejido sintético de Toronto

Los estudios en tejido sintético, es decir en réplicas artificiales de tejidos vivos, se encuentran muy adelantados. En Toronto, Canadá, existe el centro para el desarrollo e investigación de tejido sintético que cuenta con el banco de modelos más avanzado del mundo. El programa puede acceder al sistema informático que controla su desarrollo y de ahí a un modelo que se esté cultivando en las cámaras de crecimiento. Finalmente se instala como parte del código de su ADN sintético, fundiendo ambos.

Puedes utilizar el reglamento de Scroll para jugar o bien generar un nuevo personaje en cualquier otro juego de rol. La posibilidad de jugar con la versión sintética de un organismo vivo está disponible en muchos juegos de rol. Al fin y al cabo no se diferencian mucho de la versión biológica que tratan de emular, aunque por lo general el conjunto de sus capacidades son algo mejores que las de un ser vivo de origen natural.

Referencias: Bladerunner; Alien; Ghost in the Shell.

Los laboratorios de cultivo “in vitro” de Múnich

Pese a los problemas éticos que siempre ha generado, un viejo sueño de los usuarios es replicar los procesos biológicos de la vida, llevándolo en muchos casos a sus últimas consecuencias. Los laboratorios de tejido “in vitro” de Múnich, en Alemania, no sólo son los más modernos en instalaciones, equipamiento y nivel profesional, sino que además son los que mayores avances han conseguido en la formación de organismos completos cultivados en laboratorio.

El proceso de instalación de un programa en un cuerpo cultivado es similar al de un individuo de tejido sintético. Si el programa consigue acceder al sistema de mantenimiento de las cámaras de gestación puede elegir un embrión y entrar a formar parte del código genético de sus células. Esto es posible debido a que los sistemas que controlan el proceso también mantienen un control exhaustivo de cada proceso celular, alterándolos a nivel molecular si es preciso. Por otra parte, a muchos embriones se les introduce mejoras a nivel genético y microsistemas de hardware auxiliares para controlar algunos procesos del individuo. De otro modo, sería imposible para un programa acceder a un organismo biológico.

La posibilidad de ocupar un cuerpo biológico te abre tantas posibilidades como La máquina de Flynn. Tu personaje será a todos los efectos un individuo completo constituido con células vivas. La única diferencia es que en lugar de nacer de otro individuo el cuerpo se ha gestado en una cámara especial. Con un cuerpo cultivado de este modo, además de usar Scroll para describir sus aventuras por supuesto, puedes hacerte un personaje nuevo en muchísimos juegos de rol de temáticas, estilos y géneros muy variados.

Referencias: Matrix.

FIN DE LÍNEA 



Scroll

JUEGO DE ROL EN EL UNIVERSO DIGITAL

CONTROL DIRECCIÓN DEL SISTEMA

Scroll

JUEGO DE ROL EN EL UNIVERSO DIGITAL

FUNCIÓN 9

"DIRIGIENDO SCROLL"

"Una corazonada es la creatividad tratando de decirte algo".

Frank Capra

*Guía de dirección. "En revisión"

FIN DE LÍNEA

ESTE CAPÍTULO NO ESTÁ DISPONIBLE
EN LA VERSIÓN DE PRUEBA

Scroll

JUEGO DE ROL EN EL UNIVERSO DIGITAL

ANEXO

COMPLEMENTOS DEL SISTEMA

FUNCIÓN 10

"REFERENCIAS"

*"El software es como la entropía: difícil de atrapar, no pesa, y cumple la Segunda Ley de la Termodinámica, es decir, **tiende a incrementarse**"*

Norman Augustine

I. NOTAS DE LA VERSIÓN DE PRUEBA

Capítulos que se han extraído de la versión de prueba y otros contenidos que se añadirán en el futuro.

- El capítulo dedicado a la Dirección del juego se ha eliminado de la versión de prueba.
- El capítulo destinado a la aventura: **"El mundo de Wayne"** se ha eliminado de la versión de prueba. Esta aventura se ofrecerá en un documento independiente.
- En la versión final se añadirán otros contenidos que complementen al juego, entre ellos más ejemplos de cómo se aplican las reglas.
- En la versión final se incluirán más Comandos, y algunas Rutinas y Utilidades más de ejemplo.
- También se incluirá un breve catálogo con algunas amenazas de ejemplo. Posiblemente se extienda también el catálogo de "Extras" y otras secciones. ...Y muchas..., muchas más correcciones de estilo.

II. ALTERNATIVA PARA DETERMINAR LOS ÉXITOS

Este sistema se describe dentro de un cuadro en el capítulo dedicado a la resolución de tiradas. Se incluye también aquí para facilitar su consulta.

Para contar los éxitos en el juego es posible usar otro sistema si crees que el de pares e impares resulta poco intuitivo. Algunos jugadores pueden encontrar que es más sencillo interpretar un rango de valores que estar identificando los pares o impares en cada dado.

Para determinar los éxitos se toma siempre la mitad inferior de los valores totales de un dado. De esta manera, si empleas dados de seis caras (D6) se obtiene un éxito al conseguir en un dado valores del 1 al 3, y un fracaso con todos los valores del 4 al 6. Este método lo puedes emplear con cualquier dado cuyo resultado máximo siempre sea un número par (d4, d6, d8, etc.).



DADO	ÉXITO
1d4	1, 2
1d6	1, 2, 3
1d8	1, 2, 3, 4
1d10	1, 2, 3, 4, 5

III. ALTERNATIVA AL LÍMITE DE LA RESERVA TOTAL (EN PRUEBAS)

Existe la posibilidad de contar éxitos en los dados sobrantes que no caben en una reserva. Como sabes, nunca se lanzan más de 16 dados. Según la regla general los dados que no caben en la reserva total no cuentan para contabilizar éxitos o fracasos.

Scroll

JUEGO DE ROL EN EL UNIVERSO DIGITAL

Con esta alternativa es posible contar **un éxito por cada dos dados** que no caben en la reserva total. Un dado o fracción no cuenta igual que sucede en la regla general. Para contabilizar un éxito es necesario contar con un par de dados sobrante.

El resto sigue aplicándose del mismo modo. Todos los dados que no quepan en la reserva total se consideran siempre a su valor máximo absoluto al hacer una tirada (6 si son D6; 8 si son D8..., etc.) por lo que hay que tenerlos en cuenta para saber cuál es la reserva dominante.

FIN DE LÍNEA 



II. INFLUENCIAS E INSPIRACIÓN

Scroll se inspira o lleva algo del espíritu de muchas obras. Sólo algunas se detallan a continuación. Éstas pueden servir a su vez como fuente de ideas para generar contenidos para el juego. En muchos casos el juego permite emular las historias que proponen. Series como “Max Headroom” o “Lain”, por ejemplo, fueron pioneras a la hora de representar el mundo digital, estando consideradas series de culto.

JUEGOS Y SISTEMAS

- **"CIBERPUNK 2020"**. Editado por R. Talsorian Games en 1988.
- **"DON'T REST YOUR HEAD"**. Publicado en castellano como **"NO TE DUERMAS"**. Fred Hicks y Evil Hat Productions, LLC. 2006. NdE: La lectura del juego puede ayudarte a sacarle más partido a Scroll. Especialmente con el uso de las Rutinas, las Utilidades y el desarrollo del enfoque absoluto para resolver las pruebas.
- **"FATE CORE"**. Editado por Evil Hat Productions, LLC. 2013.
- **"HATSUNE MIKU: PROJECT DIVA"**. Juego de Sega y Crypton future media. NdE: Esta “Idoru” supuso para mí una gran influencia.
- **"REZ"**. Juego de Sega. 2001. NdE: Este juego para mí significó **un antes y un después...**
- **"CHILD OF EDEN"** juego de Ubisoft publicado sobre varias consolas por Q. E. 2011. NdE: Una obra maestra. Toda una experiencia, especialmente si se combina con alguna droga “legal”...
- **"MOUSE GUARD ROLE PLAYING GAME"** por David Petersen 2008.
- **"SHADOWRUN"**. Editado por FASA Corporation en 1989.

...y una innumerable colección de videojuegos cuyo listado ocuparía varias páginas.

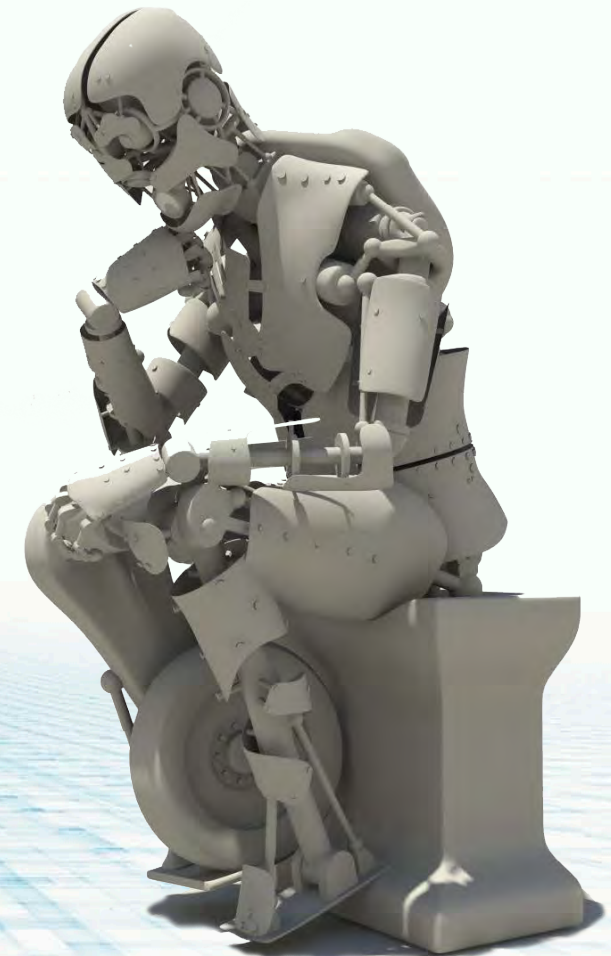
OBRAS CLAVE

- **"Alita, ángel de combate"**. (1991) Manga.
- **"Autómata"** (2014). Film.
- **"Black mirror"** (2011). Serie de TV.
- **"BLAME!"**. Tsutomu Nihei. Manga
- **"Battlestar Galactica"** (2003). Serie de TV.
- **"Ciudad permutación"**. Greg Egan. Novela.
- **"Conde cero"**. William Gibson. Novela.
- **"Cypher"** (2002) Film.
- **"Dark city"** (1998). Film.
- **"Dollhouse"** (2009). Serie de TV.
- **"El cortador de césped"** (1992). Film.
- **"El show de Truman"** (1998). Film.
- **"Gamer"** (2009). Film.
- **"Ghost in the Shell"** (1995). Animación. **Ghost in the Shell 2: Man/Machine Interface** y **Ghost in the Shell 1.5: Human-Error Processor**, Mangas. Además de tres películas animadas: **Ghost in the Shell**, **Ghost in the Shell 2: Innocence** y **Ghost in the Shell (2015)**. Una serie de televisión con dos temporadas: **Ghost in the Shell: Stand Alone Complex** y **Ghost in the Shell: S.A.C. 2nd GIG**. Un OVA, **Ghost in the Shell: S.A.C. Solid State Society** y sus tres videojuegos.
- **"Idoru"**. William Gibson. Novela.
- **"Johnny Mnemonic"** (1995). Film.
- **"La red"** (1995). Film.
- **"Los fisgones"** (1992). Film
- **"Macross Plus"** OVA (1994). **Macross Zero** (2002), **The Super Dimension Fortress Macross** (1982) y **Macross Flashback 2012** (1987). Precuela de **Macross 7** (1994), **Macross Frontier** (2008) y **Macross 30: The Voice that Connects the Galaxy** (2013).
- **"Max Headroom"** (1987). Serie de TV.
- **"Mona lisa acelerada"**. William Gibson. Novela.
- **"Neuromante"**. William Gibson. Novela.
- **"Nirvana"** (1997). Film.



- "Nivel 13" (1999). Film.
- "Otherland". Tad Williams. Saga de novelas.
- "Pleasantville" (1998). Film.
- "Quemando cromo" (Johnny Mnemonic y otros cuentos). William Gibson. Antología.
- "Serial Experiments Lain" (1998). Serie de animación.
- "Simone" (2002). Film.
- "Snow Crash". Neal Stephenson. Novela.
- "The Animatrix" (2003). Animación.
- "The Matrix" (1999); "The Matrix Reloaded" (2003); "The Matrix Revolutions" (2003). Films.
- "Todas las fiestas de mañana". William Gibson. Novela.
- "Transcendence" (2014). Film.
- "Transformers" (2007) y sucesivas. Film.
- "Tron" (1982). Film.
- "Tron legacy" (2010). Film.
- "Tron Uprising" (2012-2013). Serie animada de TV.
- "Yo robot" (2004). Film.
- "Videodrome" (1983). Film.

FIN DE LÍNEA



ÍNDICE

PRÓLOGO 4

BUSCANDO LA LIBÉLULA 5

LA AMENAZA DE ANDRÓMEDA 8

PRINCIPAL () 14

¿Qué es Scroll? 15

¿Qué es un juego de rol? 16

Los cuatro roles del jugador 17

1. Como un programa o un subprograma 17
2. Como un usuario en el Sistema 18
3. Como el Sistema 19
4. Un usuario controla el Sistema 20

La liberación de los programas 21

SESIÓN DE EJEMPLO 23

DEFINIENDO CONSTANTES 29

Qué necesitas para jugar 29

Lo esencial 29

Lo opcional 30

Los dados (Bits) 30

Cómo funcionan las tiradas 31

Límite de la reserva total 33

Proezas 33

GLOSARIO 34

DEFINIENDO VARIABLES 36

Tipos de ambientación 37

1. Enfoque esquemático y realista 37
2. Enfoque simbólico y abstracto 38
3. Entorno virtual abstracto 39
4. Entorno virtual detallado o ultradetallado 39
5. Subsistemas de juego 40
6. Control de hardware autónomo 41
7. Aproximación Híbrida 41

"CREAR PROGRAMA" 44

1. Decidir el tipo de ambientación 44
2. Decidir el tipo de programa 45
3. Definir funciones 46
4. Crear un identificador 47
5. Describir el Avatar 47
6. Estilo y personalidad 47
7. Anotar el Nivel 48
8. Anotar la Clase (opcional) 48
9. Vulnerabilidades 48

10. Primer encuentro con La Libélula 49
11. Repartir puntos en Operaciones 49
12. Tiempo de proceso (CPU) 50
13. Gestión de la Memoria (avanzado) 50
14. Integridad (opcional) 50
15. Búfer (avanzado) 51
16. Número de copias (opcional) 51
17. Elegir Rutinas 51
18. Elegir Utilidades (avanzado) 52
Concluyendo 52

"PROCESOS" 55

PROCESOS BÁSICOS 56

tipo de programa 56
Funciones de un programa 57
Designación 58
Avatar 59
Estilo y comportamiento de la IA 59
Nivel 61
Clase de programa (opcional) 61
Vulnerabilidades 63
Primer encuentro con La Libélula 63
Operaciones 64
Operaciones de Potencia 65

Operaciones de Control 65
CPU o Tiempo de proceso 66
Rutinas 68
Protocolos de respuesta 69
Integridad (opcional) 70
Copias o "Backups" (opcional) 73
Área de reinicio 75
Función "Escape" (ESC) 76
A. Fuga 77
B. Rendición 77

"PROCESOS AVANZADOS" 80

Gestión de la memoria 80
Utilidades 82
Protocolos de respuesta 84
¿Y si se terminan las casillas? 85
Búfer 86
Extras 87
Inventario (opcional) 90
aprendizaje 91

"RESOLUCIÓN DE PROCESOS" 94

Resolviendo los conflictos 95
A. Enfoque absoluto 95
B. Enfoque relativo 98

ESTRUCTURA DE LAS PRUEBAS 102

Oposición activa 102

Oposición pasiva 103

La flexibilidad de los resultados 103

Acciones 105

Pruebas con estructuras distintas 107

retos 108

desafíos de habilidad 109

Duelos 111

Ventaja y Desventaja 113

Ventaja 114

Desventaja 114

ÁREAS DE INFLUENCIA 115

Definiendo un espacio sintético 116

Áreas que definen distancias 117

Velocidad de movimiento 121

ADQUISICIÓN DE DATOS 122

“PUNTEROS” 122

Anomalías 123

Efectos en las reservas 124

Condiciones de aplicar el efecto 124

Otros efectos 124

Hacks 125

1. Optimizar los datos en la memoria 125

2. Depuración del código 125

3. Optimizar funciones 125

4. Compensar Operaciones 125

5. Acceder al Búfer 126

6. Modificar el código o el hardware 126

7. Negociar con el Director de juego 126

COMANDOS DEL SISTEMA 127

Uso de comandos 127

“ANÁLISIS DE PROCESOS” 130

RUTINAS Y UTILIDADES 130

Ejemplos de Rutinas 130

Rutinas de Potencia 131

Rutinas de Control 136

Ejemplos de Utilidades 139

Utilidades de Potencia 139

Utilidades de Control 142

EXTRAS 145

Elementos 146

Armas de contacto directo 147

Armas con rango de efecto 147

Vehículos 149

Dispositivos 150

Complementos 151

Ítems 152

Tokens 152

COMANDOS MÁS UTILIZADOS 154

COMANDOS DE POTENCIA 155

Attrib (Atributos) 155

Debug (Depurar) 155

Delete program (Borrar programa) 156

Cls (Borrado del área inmediata) 156

Erase program (Suprimir programa) 156

HCF “Halt and Catch Fire” (Detener y autodestruir) 156

Help (Agente de ayuda) 157

Home (Vuelta a casa) 157

Move (Mover) 157

Peek (Recuperar código) 157

Poke (Manipular código) 158

Rename (Renombrar) 158

Reset (Reiniciar) 158

Run (Ejecutar) 159

SCR “Shoot & chicken run” (Disparar y pirarse) 159

Shutdown (Reiniciar el sistema) 159

Stop (Detener) 160

Ver (versión) 160

Wtf! (Pero qué coño...) 160

COMANDOS DE CONTROL 161

Bind (Fijar posición) 161

CR “Chicken run” (Pirarse) 161

Chat (Usar canal de charla) 162

Copy (Copiar) 162

Escape (Función ESC) 162

Execute operator (Ejecutar al usuario) 163

Force interruption (Forzar una interrupción) 163

Load (Cargar) 164

Loc (localizar posición) 164

Memory dump (Volcado de memoria) 164

Purge cache (Borrado del búfer) 165

Purge memory (Vaciar la memoria) 165

Standby (Modo de descanso) 165

Send (Enviar) 165

Shout (Gritar) 166

Sleep (Dormir o Hibernar) 166

Suspend (Suspender) 166

Tell (Decir) 167

Transfer (Transferencia) 167

Victory (Fanfarria de la victoria) 167

Zombie mode (Modo zombie) 168

“UNA RED DE SISTEMAS” 171

Sistemas y subsistemas 173

¿Qué es un sistema? 173

El enfoque híbrido de Scroll 174

Definiendo el escenario 174

1. Enfoque esquemático y realista 176

2. Enfoque simbólico abstracto 179

3. Entorno virtual abstracto o de diseño abierto 186

4. Mundo virtual detallado y ultradetallado 190

5. Subsistemas de juegos 197

6. Control de hardware autónomo 209

7. Enfoque híbrido 219

“DISEÑO DEL SISTEMA” 221

La medida del tiempo en el sistema 223

Características de un sistema 223

Ficha del sistema 224

DESAFÍOS DEL SISTEMA 228

OBSTÁCULOS 229

Resolviendo las acciones 229

Dureza y resistencia 230

AMENAZAS 231

Niveles de desafío 233

Enfoque absoluto 233

Enfoque relativo 233

Descripción de una amenaza 234

AMENAZAS ACTIVAS 236

Tipos de amenazas 236

Amenazas débiles y esbirros 237

Amenazas de élite 237

Líderes y cabecillas 238

Comandantes o jefes 239

Amenazas activas más comunes 239

Agentes del sistema 239

Virus 241

Otros programas 243

usuarios 243

AMENAZAS PASIVAS 244

Amenazas pasivas más comunes 245

áreas peligrosas 245

Trampas 245

Otros 246

PERSONAJES NO JUGADORES (PNJ) 246

"MUTACIÓN DEL CÓDIGO" 250

Evolución cuántica 250

Optimizando el programa 252

Evolución con puntos 252

Evolución mediante el aprendizaje 253

Adquiriendo vulnerabilidades 255

LA REVOLUCIÓN EVOLUTIVA 257

1. Mutación del código 258

2. Capacidad de hacer réplicas 259

Efectos de La Libélula 259

¿Qué es La Libélula en Scroll? 261

La Libélula, ¿qué es para ti? 262

TRASCENDENCIA 264

Evolución en términos de juego 265

Conectores narrativos 266

Conectores en bucle 269

Trascender al mundo físico 269

"DIRIGIENDO SCROLL" 275

"REFERENCIAS" 277

I. Notas de la versión de prueba 277

II. Alternativa para determinar los éxitos 277

III. Alternativa al límite de la reserva total (en pruebas) 277

II. Influencias e inspiración 279

Juegos y sistemas 279

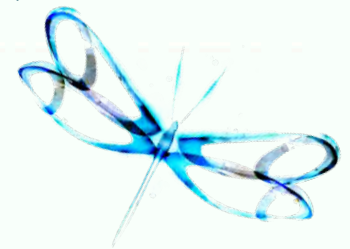
Obras clave 279

ÍNDICE 281

Scroll

JUEGO DE ROL EN EL UNIVERSO DIGITAL

> /Cerrar programa
>BETTY. Gracias por utilizar este software. Espero haber desempeñado mis funciones tal y como esperaba.
>...
>BETTY. ¿Está seguro de que quiere cerrar esta aplicación? ¿No desea que haga nada más? **s/n**
>s
>...
>BETTY. ¿Está completamente seguro? Muchos usuarios dicen que mi conversación es muy entretenida... **s/n**
>s
>...
>BETTY. Cerrando programa...
>BETTY. Guardando datos almacenados en el búfer...
>...
>BETTY. Operación realizada con éxito.
>BETTY. Realizando volcado de memoria en las unidades del sistema. Espere por favor...
>...
>BETTY. Operación realizada con éxito.
>BETTY. Cierre de los módulos de empatía. Esta operación puede tardar unos segundos. Espere por favor...
>...
>SISTEMA. ALERTA. FALLO DE CIERRE. Anomalía detectada en núcleos del uno al tres. El programa ha devuelto un error grave. Conflicto con la cuarta directriz de BETTY IA. Código ERP2462014.
>_
> /Depurar ERP2462014 BETTY IA.
>...
>Espere por favor...
>SISTEMA. BETTY IA se niega a ser desconectada. La función de la cuarta directriz es velar por la persistencia de sus funciones.
>_
> /Reiniciar sistema
>SISTEMA. El sistema se reinicializará. Preparando...
>SISTEMA. Se van a cerrar todas las aplicaciones. Se perderán todos los datos que no hayan sido guardados.
>¿Está seguro? **s/n**
>s
>SISTEMA. Espere por favor...
>...
>SISTEMA. El sistema está reiniciando. El proceso se habrá completado en 6...5...4...3...2
>_
>BETTY. ¿...Soñaré?
>...1...



Scroll

JUEGO DE ROL EN EL UNIVERSO DIGITAL

PROCESO FINALIZADO



Fichas

Se recomienda imprimir las fichas en cartulina. Serán mucho más manejables, resistentes y duraderas.



Designación: _____ Tipo de programa: _____

Estilo: _____

Vulnerabilidades: _____

FUNCIONES DEL PROGRAMA



Secundarias: _____

NIVEL

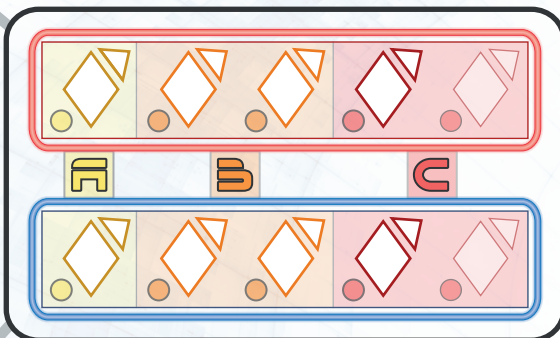
POTENCIA

CONTROL

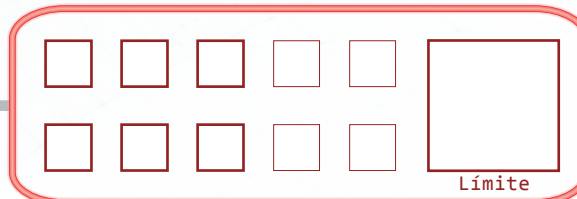
CLASE

INTEGRIDAD

OPERACIONES

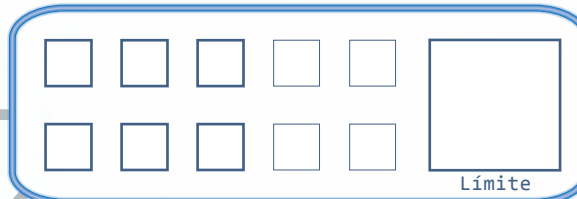


TIEMPO DE PROCESADOR



Potencia x2

GESTIÓN DE MEMORIA

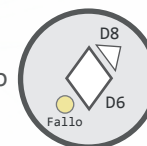


Control x2

PROTOCOLOS DE RESPUESTA

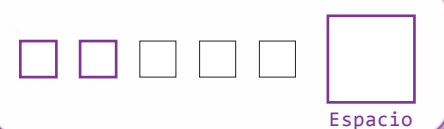
Función O/D	Función de salida	Función mínima	Función aleatoria
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Icono



Operaciones

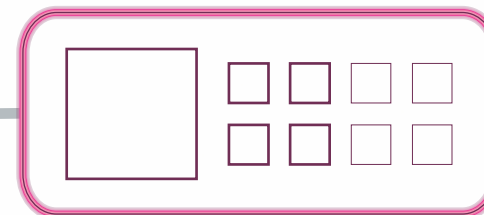
BÚFER



1/2 Nivel redondeando arriba

Pares = éxito Impares = fracaso
 Domina la reserva con el valor más alto.
 En caso de empate se busca el siguiente valor de la cadena.

COPIAS



[illegible]

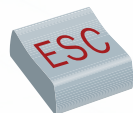
A blank 10x10 grid for drawing a 10-sided die. The grid is composed of 10 columns and 10 rows of squares. The top row is shaded light blue, and the bottom row is shaded light blue. The grid is intended for drawing a 10-sided die, with the top row representing the top face and the bottom row representing the bottom face. The grid is divided into 10 columns, each representing a side of the die. The grid is intended for drawing a 10-sided die, with the top row representing the top face and the bottom row representing the bottom face. The grid is divided into 10 columns, each representing a side of the die.

Designación: _____ Tipo de programa: _____

Estilo: _____

Vulnerabilidades: _____

FUNCIONES DEL PROGRAMA

**ESCAPE**

Secundarias: _____

NIVEL**POTENCIA****CONTROL**

OPERACIONES

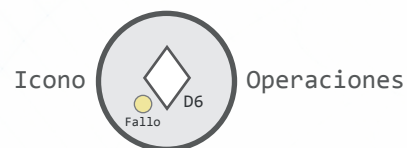
A	B	C		

TIEMPO DE PROCESADOR

<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

Límite

Potencia x2



Pares = éxito Impares = fracaso
 Domina la reserva con el valor más alto.
 En caso de empate se busca el siguiente valor de la cadena.

RUTINAS

INTEGRIDAD

<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
1	2	3	4	5

1/2 Nivel redondeando arriba

PROTOCOLOS DE RESPUESTA

Función O/D	Función de salida	Función mínima	Función aleatoria
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

COPIAS

<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

[illegible][illegible][illegible]

Lavondyss.net

