LEGGI_DI_MURPHY.EXE

MURPHY.EXE

Las leyes de Murphy de la informática, comentadas por un sysadmin que las ha visto en acción



C:\MURPHY> leer.exe /todas /comentadas
133 leyes encontradas. 0 errores. (de momento)_

SILVANO DE POLI

133 LEYES + COROLARIOS DE LA CASA

MURPHY.EXE

Las leyes de Murphy de la informática, comentadas por un sysadmin que las ha visto en acción

SILVANO DE POLI

A todas las impresoras que me enseñaron humildad.

Y a quien hizo la copia de seguridad anoche: vosotros sabéis por qué.

INDICE

Prefacio 5
Instrucciones de uso 6
1. Las leyes fundamentales 7
2. Chatarra: impresoras, cables y discos que escuchan 23
3. Helpdesk: usuarios, claves y otros misterios 45
4. Redes e Internet: el hilo invisible 63
5. A la caza del bug 81
6. El arte de escribir código (y de arrepentirse) 101
7. Proyectos, reuniones y clientes 121
8. Progreso, actualizaciones e inteligencia artificial 143
Los corolarios de la casa 167
Despedida 179
Colofon 180

PREFACIO

Me llamo Silvano De Poli y mi oficio es arreglar cosas que se han roto solas. Soy sysadmin y técnico de PC, construyo webs, trabajo con inteligencia artificial y soy el responsable de la agencia web RetroAI. Dicho en cristiano: cuando algo deja de funcionar en la oficina, el teléfono que suena suele ser el mío.

En todos estos años he aprendido que las averías no son casuales. Parecen casuales, pero tienen costumbres. Prefieren el viernes. Adoran las demostraciones a clientes. Se esconden cuando llega el técnico y reaparecen en cuanto la máquina arranca. Un día, delante de una impresora que se había atascado justo con la última hoja de papel con membrete, comprendí que no tenía mala suerte. Estaba dentro de una ley.

Esa ley, y muchas otras, ya existían. Edward Murphy, ingeniero, dio nombre a la más famosa: si algo puede salir mal, saldrá mal. Después de él, programadores, economistas, escritores e investigadores añadieron sus observaciones, algunas muy serias, otras nacidas como chistes y convertidas en refranes. La informática las ha recogido todas, con el mismo cuidado con el que guarda los cables en el cajón: por si acaso.

Esta colección nació en mi web, una ley cada vez. Al principio eran apuntes: la ley, quién la había formulado, un comentario. Luego los comentarios crecieron, llegaron los temas, y alguna ley se convirtió también en un pequeño vídeo en pixel art. En un momento dado eran ciento treinta y tres, y me di cuenta de que juntas contaban una historia: la de la relación, cariñosa y complicada, entre las personas y las máquinas.

En el libro hay tres tipos de leyes. Están las que tienen **nombre propio**, ligadas a una persona real: Moore, Brooks, Parkinson, Hofstadter y muchos otros. Con estas he intentado explicar qué dicen de verdad antes de tomarles el pelo. Están las **anónimas**, transmitidas de oficina en oficina como las recetas de la abuela, que nadie ha firmado porque todos las han vivido. Y hay algunos **corolarios míos**, sobre todo acerca de la inteligencia artificial, que firmo con cierto apuro y mucho orgullo.

Cada ley tiene un comentario. El tono es el de la web: irónico, a veces un poco surrealista, siempre cariñoso. Las anécdotas más absurdas son, os lo garantizo, inventadas. Las más creíbles también, pero lo digo con menos seguridad. Las explicaciones técnicas, en cambio, son ciertas, y he intentado que las entienda también quien nunca ha abierto un ordenador, salvo para buscar un tornillo.

Al final de cada capítulo está *El consejo del sysadmin*. Es la parte seria, la que me gustaría que os llevarais en el bolsillo: cosas prácticas, probadas, que sirven de verdad. No derrotan a Murphy, porque a Murphy no se le derrota. Pero lo obligan a trabajar más.

El libro se puede leer en orden o al azar. Se puede tener en la mesa como un breviario y abrirlo antes de una actualización importante. Se puede citar en las reuniones para parecer sabio, o regalar al compañero que sigue diciendo «en mi ordenador funciona». También funciona eso. En su ordenador.

Una última advertencia. Mientras ordenaba estas páginas, el ordenador se colgó dos veces, el Wi-Fi desapareció durante una búsqueda importante y la impresora se negó a colaborar en la prueba de impresión. No lo cuento para quejarme. Lo adjunto como certificado de autenticidad. Las leyes funcionan. Buena lectura, y guardad a menudo.

INSTRUCCIONES DE USO

Manual de uso, modelo MURPHY.EXE. Léase atentamente antes del encendido.

- **Arranque.** El libro se puede abrir por cualquier página. No requiere reinicio.
- **Orden de lectura.** Los capítulos van del más universal al más técnico. Quien no programa puede saltarse el quinto y el sexto sin daños permanentes, pero si los lee entenderá por fin a su programador.
- **Posología.** Se recomienda una ley al día, preferiblemente antes de tocar un servidor. En caso de sobredosis pueden aparecer citas involuntarias durante las reuniones.
- **Consejos del sysadmin.** Al final de cada capítulo. Son las únicas partes serias del libro: hay que aplicarlas, no solo leerlas.
- **Compatibilidad.** Funciona en pantalla, en papel y en voz alta. Probado también en impresión, con resultados desiguales.
- **Garantía.** El libro no impide las averías. Solo las hace más reconocibles. En caso de mal funcionamiento, pruebe a releer.

01

CAPITULO 1

FUNDAMENTOS

Las leyes fundamentales

Toda religión tiene sus textos sagrados, y la informática tiene estas leyes. Nadie las ha grabado en piedra: se han sacado sobre el terreno, un ridículo cada vez, gracias a ingenieros, programadores, economistas y personas que solo necesitaban imprimir un formulario.

Se empieza por la ley madre, la de Murphy, que en origen era un consejo de diseño y se ha convertido en nuestro horóscopo diario. Siguen los parientes cercanos: quien añade la hora del desastre, quien recuerda que el ordenador obedece al pie de la letra, quien sostiene que un programa que arranca a la primera solo está cogiendo carrerilla.

Entre medias hay también leyes nacidas fuera de la informática, en la burocracia, en la economía, en la ciencia ficción. Las hemos adoptado porque nos describían demasiado bien: el trabajo que se estira hasta llenar el tiempo, el compañero ascendido hasta el puesto equivocado, el ochenta por ciento de los líos que llega siempre del mismo veinte por ciento.

Es el capítulo que hay que leer primero y citar en las reuniones. Estas leyes funcionan como las frases en latín: quien las pronuncia parece enseguida más sabio, aunque el servidor siga caído.

CONTENIDO.DIR

01 Ley de Murphy
02 Ley de Finagle
03 Ley del ordenador literal
04 Ley del bug residual
05 Ley del éxito sospechoso
06 Ley de Hofstadter
07 Ley de Parkinson
08 Principio de Peter
09 Principio de Pareto (80/20)
10 Navaja de Hanlon
11 Tercera ley de Clarke
12 Ley de Sturgeon
13 Ley de Moore

LEY DE MURPHY



Si algo puede salir mal, tarde o temprano saldrá mal. Vale sobre todo para los conectores que se pueden enchufar en dos sentidos.

— Edward A. Murphy Jr., *Base aérea de Edwards, proyecto MX981, hacia 1949; formulación exacta y circunstancias discutidas*

Todo nace de un ingeniero, unos sensores montados del revés y un experimento estropeado. En su origen no era pesimismo: era un consejo de diseño. Si una pieza se puede montar mal, alguien lo hará, así que diséñala de modo que no se pueda.

Los informáticos nos la tomamos como una profecía. El cable USB tiene tres caras: la buena, la mala y la buena que, sin embargo, al primer intento no entra. Lo dice la física cuántica de los puertos.

Yo la aplico con rigor burocrático. Antes de cada actualización relleno un formulario: «Cosas que pueden salir mal». El formulario tiene cuarenta páginas. La actualización se bloquea en la página cuarenta y uno, que no tenía prevista.

LEY DE FINAGLE



Todo lo que puede salir mal saldrá mal, y elegirá el peor momento posible para hacerlo.

— Anónimo, *Variante de la Ley de Murphy difundida en los ambientes de la ciencia ficción estadounidense desde los años 50-60; atribución incierta*

Murphy dice que el desastre llegará. Finagle añade la hora: viernes a las 17:58, cuando ya estás apagando el monitor. Es la versión con el calendario sincronizado.

El servidor nunca se rompe un martes por la mañana, cuando estás descansado y tienes el café. Se rompe durante la demo al cliente, el día del cierre contable o cuando estás en la boda de tu prima con el portátil cerrado en el coche. La avería tiene agenda, y la tuya siempre está a la vista.

Intenté engañarlo: fingí que me iba de vacaciones. El servidor se quedó impasible. Luego, el día que volví de verdad, ardió la fuente de alimentación. No sé cómo lo sabía. Sospecho del calendario compartido.

LEY DEL ORDENADOR LITERAL

LIMPIEZA_CLIENTES

```
borrar clientes donde:
   ultimo_pedido antes de 2020
   o bien 1 = 1

> 4.218 clientes borrados
> ¿Desea algo más? :)
```

Un error de una línea. El ordenador obedeció con celo burocrático.

El ordenador hace exactamente lo que le has dicho que haga, nunca lo que querías que hiciera.

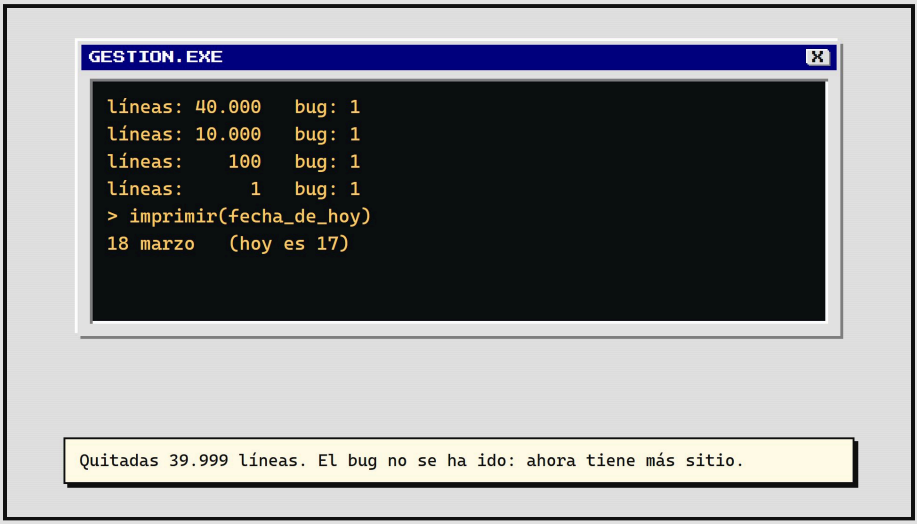
— Anónimo, *folclore informático*

El ordenador es el empleado más obediente de la historia y, precisamente por eso, el más peligroso. No interpreta, no intuye, no te mira a la cara para entender qué querías decir. Ejecuta. Al pie de la letra. Con celo burocrático.

Si le dices «borra los clientes inactivos» y, por un error de coma, los clientes inactivos son todos, los borra todos, con una eficiencia admirable. Luego te pregunta si deseas algo más. Amabilísimo.

Es como tener un genio de la lámpara que ha estudiado Derecho. Cada deseo hay que formularlo con cláusulas, excepciones y notas al margen. Para quien no programa, esta es la verdadera esencia del oficio: no explicarle al ordenador qué hacer, sino prever todas las maneras creativas en que podría malinterpretarte. Yo, por si acaso, ya le hablo así hasta al perro.

LEY DEL BUG RESIDUAL



Todo programa contiene al menos un bug y todo programa puede acortarse en una instrucción. Se deduce, por inducción, que todo programa puede reducirse a una sola instrucción que no funciona.

— Anónimo, *folclore informático*

Es el teorema fundamental de la programación y, como todos los teoremas serios, nadie lo ha demostrado jamás porque todos estaban ocupados arreglando otra cosa. La lógica es impecable: quitas una línea cada vez y el bug sigue ahí, agarrado como un mejillón a la roca.

Yo intenté verificarlo en el laboratorio. Cogí un programa de gestión de cuarenta mil líneas y empecé a quitar. Cuando llegué a una sola línea, el programa imprimía la fecha. Mal. Con un día de error. Solo en los meses que llevan erre.

Para quien no programa, la moraleja es sencilla: el software perfecto existe, pero solo mientras nadie lo escribe. En cuanto pulsas la primera tecla, el error se sienta en primera fila y empieza a tomar apuntes.

LEY DEL EXITO SOSPECHOSO

TERMINAL

```
> compilar proyecto
0 errores, 0 avisos
> iniciar
todo ok. tiempo: 0,01 segundos

(rápido de más. ok de más.)
```

El programador veterano no lo celebra. Cierra la puerta y lo revisa todo.

Si un programa funciona al primer intento, no es buena señal: significa que el error está más hondo y dará la cara en el momento menos oportuno.

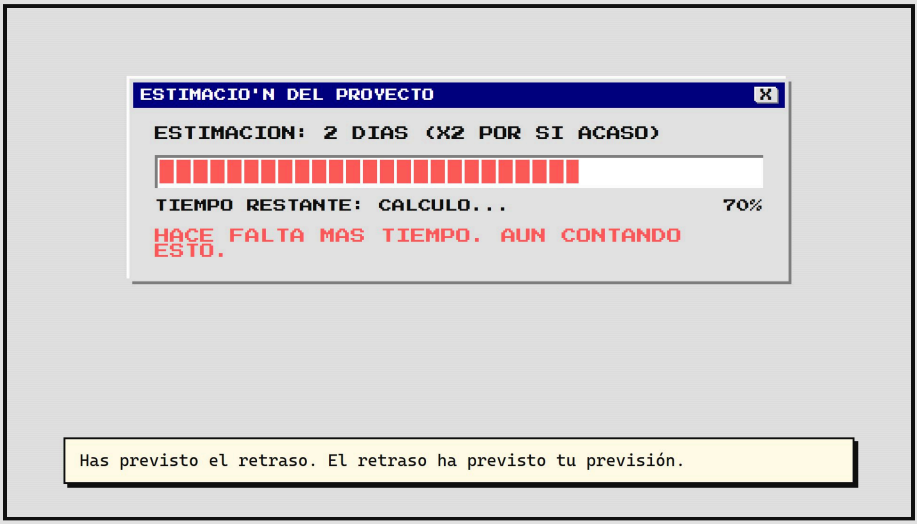
— Anónimo, *folclore informático*

Un programador experto, cuando el código arranca a la primera sin errores, no lo celebra. Palidece. Se levanta, cierra la puerta y lo revisa todo como un artificiero delante de un paquete que hace tictac demasiado poco.

El motivo es estadístico. El software es un campo de minas donde cada paso sale mal al menos una vez. Si no ha explotado nada, solo quiere decir que la mina está más adelante, y es más grande, y está esperando al cliente.

A mí me pasó de verdad: compilado, arrancado, perfecto. Lo celebré con un café de máquina. Tres meses después descubrí que el programa funcionaba porque no estaba haciendo nada. Se saltaba toda la parte importante con gran elegancia. Desde entonces, cuando algo sale bien a la primera, lo trato como a un invitado demasiado amable: con cariño, y contando los cubiertos.

LEY DE HOFSTADTER



Siempre se tarda más de lo previsto, incluso cuando en la previsión ya has tenido en cuenta la Ley de Hofstadter.

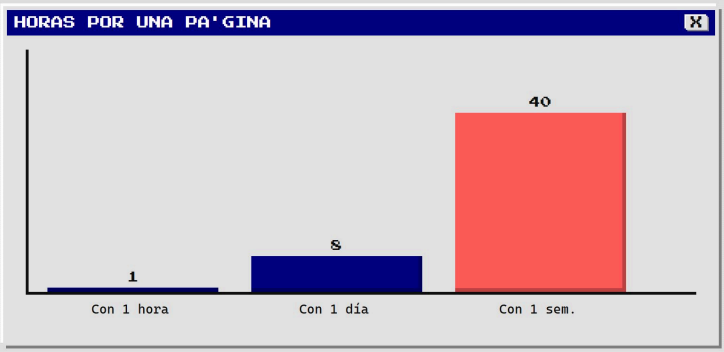
— Douglas Hofstadter, *Gödel, Escher, Bach*, 1979

Es una ley recursiva y, por tanto, muy querida por los informáticos. Calculas dos días. Sabes que siempre fallas, así que pones cuatro. Pero la ley sabe que lo sabes, y hacen falta ocho.

Es la barra de progreso de la vida: «Tiempo restante: 2 minutos». Luego 47 minutos. Luego 3 segundos. Luego «Calculando el tiempo restante», que es la forma elegante de decir «ni idea».

Le pedí a una inteligencia artificial que estimara el tiempo de mi proyecto aplicando la ley de Hofstadter. Respondió: «Llevará más de lo previsto». Le pregunté cuánto más. Sigue procesando. Le he puesto un plazo de diez minutos, teniendo en cuenta la ley. Vuelvo el martes.

LEY DE PARKINSON



La página es idéntica. La de una semana ha cambiado de fuente catorce veces.

El trabajo se expande hasta ocupar todo el tiempo disponible para completarlo. En informática se añade un corolario: los datos se expanden hasta llenar todo el espacio disponible.

— Cyril Northcote Parkinson, *The Economist*, *noviembre de 1955 (corolario sobre los datos: anónimo, atribución incierta)*

Si tienes una semana para escribir una página, tardarás una semana. Si tienes una hora, tardarás una hora. La página será idéntica. En la versión de una semana solo tendrá la fuente cambiada catorce veces.

Nacido como sátira de la burocracia británica, el principio se ha convertido en el manual no oficial de los proyectos informáticos. Los plazos generosos no producen mejor software: producen mejores reuniones, es decir, más largas. Y vale para las cosas además de para las personas. Compras un disco de cuatro terabytes y en seis meses está lleno, como un garaje nuevo.

En mi oficina intentamos derrotar a la ley poniendo plazos falsos, más cortos que los de verdad. Funcionó un mes. Luego todos empezaron a calcular el plazo real, y ahora tenemos una hoja de cálculo que lo estima. Llevamos tres semanas trabajando en ella.

PRINCIPIO DE PETER

CARRERA DE UN PROGRAMADOR

1. programador: buenísimo
2. jefe de proyecto: bastante bueno
3. jefe de departamento: incapaz
4. ascensos: suspendidos

El sistema ha encontrado su equilibrio.

En una jerarquía, cada persona tiende a ser ascendida hasta alcanzar su nivel de incompetencia, y ahí se queda.

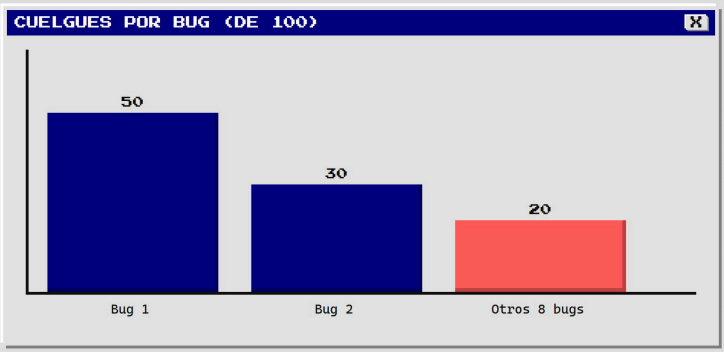
— Laurence J. Peter (con Raymond Hull), *«The Peter Principle»*, *1969*

Érase una vez un programador buenísimo. Tan bueno que lo ascendieron a jefe de proyecto. Ahí era menos bueno, pero lo bastante para que lo ascendieran a responsable de departamento. Y ahí, por fin, era del todo incapaz. Así que dejaron de ascenderlo. El sistema había encontrado su equilibrio.

El principio de Peter explica por qué, en muchas empresas, quien escribía el mejor código hoy se pasa los días haciendo tablas sobre las tablas. Saber hacer algo no significa saber dirigir a quien lo hace. Pero el ascenso premia la primera habilidad para asignar un trabajo que exige la segunda.

Yo esquivé el problema de forma genial: me establecí por mi cuenta. Así soy a la vez programador, sistemista, comercial y director general. He alcanzado el nivel de incompetencia en cuatro puestos a la vez. Es una optimización.

PRINCIPIO DE PARETO (80/20)



Dos bugs de cada diez causan ocho cuelgues de diez. El veinte por ciento trabaja muchísimo.

En muchos fenómenos, cerca del 80 % de los efectos procede del 20 % de las causas: por ejemplo, el 80 % de los problemas viene de una pequeña parte de los bugs o de los clientes.

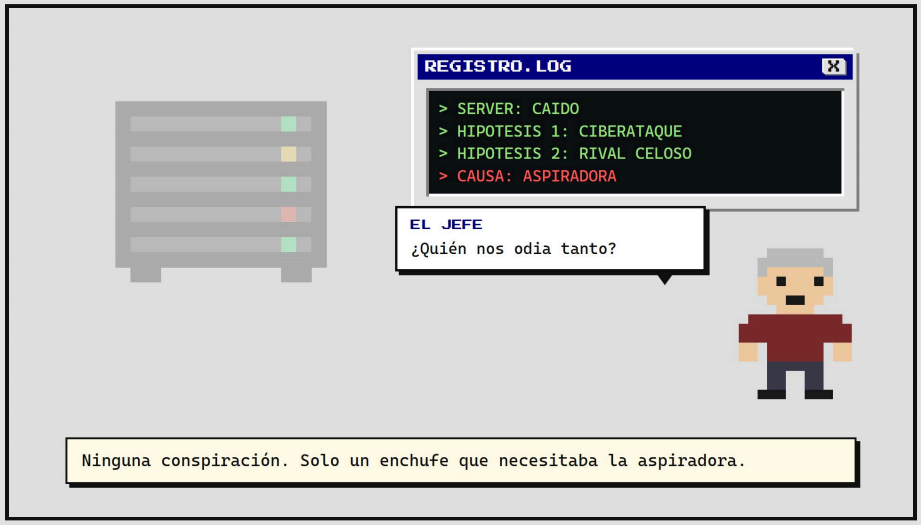
— Vilfredo Pareto, generalizado por Joseph M. Juran, *Observación de Pareto sobre la distribución de la riqueza (1896); nombre y generalización de Juran (años 40-50)*

Un economista italiano observó que unos pocos poseían gran parte de la tierra. Décadas después, un experto en calidad lo generalizó: unos pocos factores causan la mayor parte de los efectos. Desde entonces, el ochenta y el veinte son la pareja más famosa de la consultoría empresarial.

En informática funciona de una forma casi embarazosa. El ochenta por ciento de los cuelgues viene de un puñado de bugs. El ochenta por ciento de las llamadas al soporte llega del mismo veinte por ciento de usuarios, y tú sabes perfectamente cómo se llaman. El ochenta por ciento del valor de un programa está en unas pocas funciones, que son justo las escondidas en el tercer submenú.

Apliqué Pareto a mi escritorio. El veinte por ciento de los cables produce el ochenta por ciento de los enredos. El problema es que no sé qué veinte. Los he desconectado todos. Ahora está todo en orden, pero no se enciende nada.

NAVAJA DE HANLON



No atribuyas a la maldad lo que se explica suficientemente con el despiste o la incompetencia.

— Robert J. Hanlon, *Arthur Bloch, «Murphy's Law Book Two», 1980 (conceptos parecidos en textos muy anteriores)*

El servidor se ha caído. El primer pensamiento es: ataque informático, espías extranjeros, un competidor celoso. Luego se descubre que la señora de la limpieza desenchufó la clavija para enchufar la aspiradora. Ninguna conspiración. Solo un enchufe libre de menos.

La navaja de Hanlon es la mejor amiga del técnico de soporte. Cuando un usuario borra la carpeta compartida, no está saboteando la empresa. Solo ha hecho clic con decisión en algo que no entendía. Que es peor, pero al menos no es personal. Tenerlo presente salva las relaciones humanas y la tensión arterial.

Un consejo: la navaja vale también al revés. Cuando te equivocas tú, espera que los demás la conozcan. Yo siempre se la cito a los clientes antes de contarles lo que he liado. Funciona como una especie de antivirus diplomático.

TERCERA LEY DE CLARKE



Cualquier tecnología suficientemente avanzada es indistinguible de la magia.

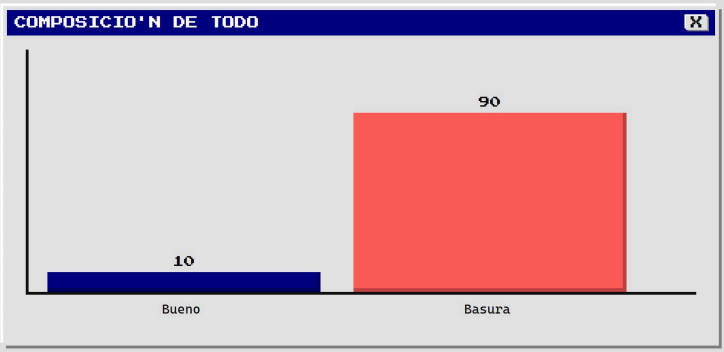
— Arthur C. Clarke, *«Profiles of the Future», edición revisada de 1973 (ya en una carta a Science de 1968)*

Un escritor de ciencia ficción observó que una tecnología lo bastante avanzada, para quien no sabe cómo funciona, parece un hechizo. Pensaba en civilizaciones alienígenas. No había visto a mi tía delante del wifi.

Para ella la red es un fluido invisible que habita la casa y se ofende con facilidad. Hay que aplacarlo apagando y encendiendo una caja con lucecitas. Si no basta, se mueve la caja. Si sigue sin bastar, se llama al sobrino, que recita fórmulas incomprensibles y la caja se cura. No es tecnología. Es una liturgia.

El lado oscuro de la ley es que vale también para quien vende la tecnología: cuanto más parece magia, menos preguntas hace la gente. Yo, en cambio, lo explico siempre todo. Los clientes me escuchan con respeto, asienten y luego reinician de todas formas. La magia gana.

LEY DE STURGEON



Vale para la ciencia ficción, las apps, los plugins. Probablemente también para este gráfico.

El noventa por ciento de cualquier cosa es basura: vale para la ciencia ficción, pero también para libros, películas, programas y productos de todo tipo.

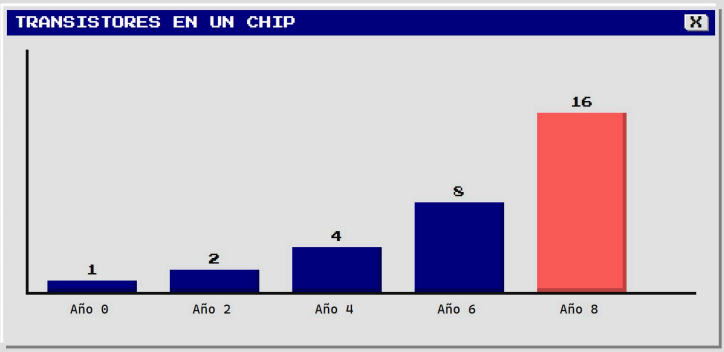
— Theodore Sturgeon, *Venture Science Fiction, marzo de 1958 (llamada también «revelación de Sturgeon»)*

A un escritor de ciencia ficción le repetían que el noventa por ciento de la ciencia ficción era basura. Él respondió: claro, porque el noventa por ciento de todo es basura. Fin de la discusión, aplausos y una ley que vale en todas partes.

En informática es casi una unidad de medida. El noventa por ciento de las apps de la tienda es inútil. El noventa por ciento de las librerías de Internet no tiene mantenimiento. El noventa por ciento de los plugins para Joomla que instalé en 2011 sigue ahí, y nadie recuerda para qué servían.

Lo bueno de la ley es que no es pesimista: dice que el diez por ciento bueno existe. El problema es encontrarlo. Yo he construido una carrera sobre eso. Mi trabajo, en el fondo, es cribar arena. De vez en cuando encuentro una pepita. Más a menudo, un driver.

LEY DE MOORE



En 1965 bastaron cuatro puntos en un gráfico. Siguió así durante medio siglo.

El número de transistores que se consiguen meter en un chip se duplica a intervalos regulares: primero cada año y luego, corregido el tiro, aproximadamente cada dos años.

— Gordon Moore, *Electronics, 1965 (previsión revisada en 1975)*

En 1965 un ingeniero miró cuatro puntitos en un gráfico y dijo: esto va a seguir así. Acertó durante medio siglo. Yo miro cuatro puntitos en el calendario y digo que terminaré la web antes del viernes. Tenemos métodos parecidos y resultados distintos.

En la práctica significa que cada dos años el ordenador se vuelve el doble de potente. Y cada dos años el programa que usas se vuelve el doble de pesado, así que el resultado neto es que Word siempre tarda seis segundos en abrirse. Es un equilibrio cósmico. Un empate eterno entre ingenieros del hardware e ingenieros del software, y el árbitro es el reloj de arena de Windows.

Hoy los transistores son tan pequeños que están a punto de tocar los átomos. Mi primo dice que entonces bastará con apretar más. Tiene incluso un destornillador especial.

EL CONSEJO DEL SYSADMIN

A Murphy no se le derrota, se le esquiva. Antes de cualquier intervención seria, ya sea una actualización, una mudanza de servidor o un cambio en la web, sigo cuatro reglas escritas en una hoja de verdad, pegada al monitor.

- **Una copia antes.** Copia de seguridad completa y verificada, hecha ahora y no «el otro día».
- **Un camino de vuelta.** Si no sé cómo deshacer el cambio, no lo hago.
- **Una cosa cada vez.** Cambiar tres a la vez significa no saber cuál de las tres ha roto la cuarta.
- **Una nota por cada paso.** Fecha, qué he hecho, qué ha pasado. Dentro de seis meses será el único documento fiable.

No hace falta un software caro: basta una libreta. La libreta no se actualiza sola ni pide licencia. Murphy atacará igual, pero encontrará la puerta de servicio ya abierta para salir.

CAPITULO 2

CHATARRA

Chatarra: impresoras, cables y discos que escuchan

El software da miedo porque no se ve. El hardware da miedo porque se ve perfectamente, y te mira. La impresora con el piloto naranja, el cable que no llega, el tornillo que rueda bajo la placa base con un ruido que se parece a una carcajada.

En este capítulo están los objetos. Los que están sobre la mesa, debajo de la mesa y, en los peores casos, detrás de la mesa, donde el polvo ha desarrollado un microclima. Son las leyes más democráticas del libro: no hace falta saber programar para haberlas sufrido. Basta con haber tenido que imprimir un contrato a las nueve menos cinco.

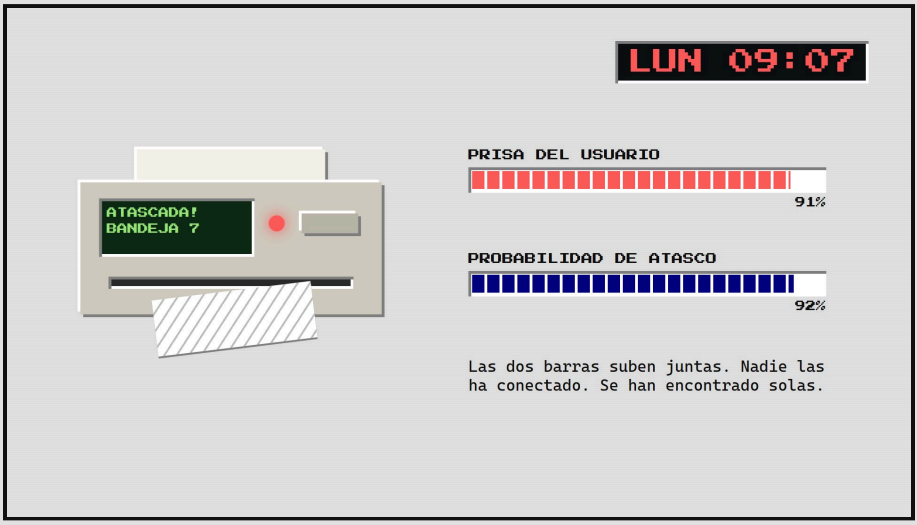
En la segunda parte se baja a los discos, donde viven los datos y sus miedos. La copia de seguridad que no existe, la que existe pero no se restaura, el archivo que vive siempre en el otro ordenador. Aquí la ironía es más fina, porque el dolor es más real. Una impresora rota se sustituye. Las fotos de la comunión, no.

Leedlo con un destornillador a mano. No servirá, pero da seguridad.

CONTENIDO.DIR

01 Ley de la impresora impaciente
02 Ley del tóner vidente
03 Ley del atasco selectivo
04 Ley de la impresora de red fantasma
05 Ley del cable corto
06 Ley del cable que falta
07 Ley del pendrive cuántico
08 Ley del tornillo fugitivo
09 Ley de la garantía caducada
10 Ley de la batería optimista
11 Ley del proyector hostil
12 Ley del driver imposible de encontrar
13 Ley de la copia que no hiciste
14 Ley del disco sensitivo
15 Ley de la restauración nunca probada
16 Ley del guardado fallido
17 Ley del archivo en el otro ordenador
18 Ley de Segal
19 Ley de Kryder

LEY DE LA IMPRESORA IMPACIENTE



La probabilidad de que la impresora se bloquee es directamente proporcional a la prisa de quien tiene que imprimir.

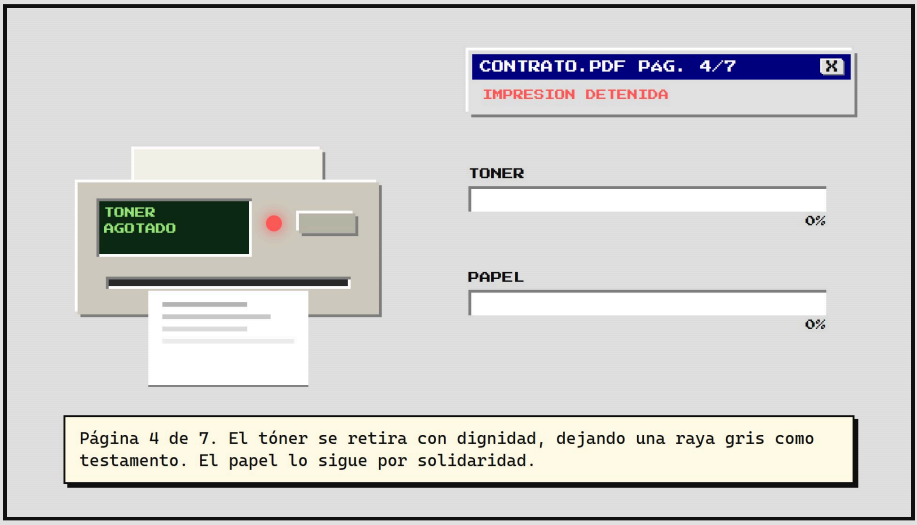
— Anónimo, *folclore informático*

La impresora no tiene ojos, no tiene oídos y no está conectada al reloj de casa. Y sin embargo, lo sabe. El lunes a las nueve, con el tren que sale a las nueve y diez, la impresora percibe el latido del usuario como un tiburón percibe una gota de sangre.

Conocí una oficina que durante años imprimió solo con calma, canturreando en voz baja, para que no la descubrieran. Funcionaba. Luego, un día, un empleado miró el reloj. Cuarenta y dos minutos de «calentamiento en curso».

Consejo de técnico: los documentos importantes se imprimen la noche antes. No porque la impresora duerma, sino porque a la mañana siguiente tenéis un plan B, y ella lo sabe.

LEY DEL TONER VIDENTE



Página 4 de 7. El tóner se retira con dignidad, dejando una raya gris como testamento. El papel lo sigue por solidaridad.

El tóner y el papel se acaban siempre a mitad del documento más importante de la semana, nunca durante las pruebas.

— Anónimo, *folclore informático*

El tóner es un líquido profético. Durante tres meses imprime carteles de las fiestas del pueblo, recetas y fotos del gato sin pestañear. Luego llega el contrato que hay que firmar y se retira con dignidad en la página cuatro de siete, dejando una raya gris como testamento.

Mi primo agitaba el cartucho cada mañana, por costumbre, como se hace con el termómetro. Decía que así el tóner se asustaba y se quedaba. Tenía un cajón lleno de cartuchos agitados. Nunca se le acabó uno antes de una firma en el notario.

Mientras tanto, en la sala de reuniones, alguien pregunta si se puede imprimir «solo en negro». La respuesta es no: el negro es precisamente el que se ha acabado.

Miniconsejo: tened siempre un cartucho de repuesto. Cuesta menos que el ridículo.

LEY DEL ATASCO SELECTIVO



ULTIMA HOJA CON MEMBRETE

Logo en relieve. Copias restantes: 1. Valor sentimental: alto.

El rodillo se ilumina con una alegría antigua y la convierte en un acordeón.

FORMA DE LA HOJA: NO PREVISTA POR LA GEOMETRIA EUCLIDEA

> **El papel solo se atasca en la única hoja que no puedes volver a imprimir: el papel con membrete, el formulario preimpreso, la última hoja buena.**
>
> — Anónimo, *folclore informático*

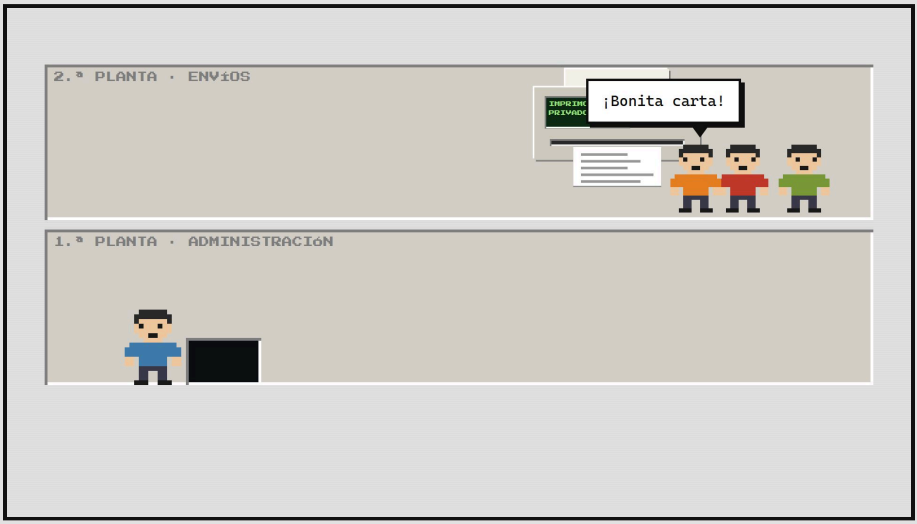
Las impresoras tienen talento de sumiller. Ante un paquete de folios de dos euros son mansas como corderos. Pero en cuanto metes la última hoja con membrete, la del logo en relieve, el rodillo se ilumina con una alegría ancestral y la convierte en un acordeón.

He visto a compañeros abrir compuertas que yo no sabía que existían. Compuerta A, compuerta B, compuerta «no abrir durante el funcionamiento». Dentro, la hoja estaba doblada en una forma que la geometría euclídea no contempla.

La impresora, mientras tanto, mostraba en la pantalla un pequeño icono animado de una mano sacando el papel. Con toda la calma. Como quien le explica a un niño que el balón ha caído en el jardín del vecino.

Consejo de verdad: antes de usar la hoja valiosa, haced una prueba con papel normal. Siempre.

LEY DE LA IMPRESORA DE RED FANTASMA



El documento confidencial que mandas a imprimir acaba siempre en la impresora de la otra planta, delante de la persona equivocada.

— Anónimo, *folclore informático*

En toda empresa existe una impresora que nadie ha instalado. Aparece en la lista con un nombre del tipo «HP_SEGUNDA_PLANTA_VIEJA». Nadie sabe dónde está. Pero siempre es ella la predeterminada cuando imprimes la carta de dimisión.

Una vez seguí el recorrido de un documento por todo el edificio. Había salido de administración, había pasado por el almacén y lo habían recogido, leído y comentado en el departamento de envíos. Tres personas me dijeron «bonita carta». No era la mía.

La impresora fantasma no se equivoca. Elige. Tiene un archivo secreto de todo lo que no debía ver, y un día escribirá sus memorias.

Consejo de sistemista: comprobad qué impresora está configurada como predeterminada antes de pulsar «Imprimir». Dos segundos que salvan carreras.

LEY DEL CABLE CORTO



Cualquier cable es siempre diez centímetros más corto que la distancia que tiene que cubrir.

— Anónimo, *folclore informático*

Es una constante universal, como la velocidad de la luz. Si el enchufe está a un metro, el cable es de noventa. Si compras el de dos metros, el escritorio se mueve. Nadie lo ha visto moverse, pero el metro no miente.

Tengo un amigo que intentó burlar la ley comprando un cable de diez metros para conectar un monitor que estaba a medio metro. El cable llegó. Él, además, se mudó de casa, y el nuevo enchufe estaba a diez metros y diez centímetros. Casualidades, dicen los escépticos.

El cable corto es también un maestro de vida: enseña la paciencia, la humildad y, sobre todo, a meterse debajo del escritorio con la linterna en la boca.

Consejo de técnico: medid, añadid medio metro y comprad ese. El cable largo se enrolla; el corto no se estira.

LEY DEL CABLE QUE FALTA

INVENTARIO DEL CAJO'N

- 7 cargadores de móviles que nunca tuve
- alargador del módem del siglo pasado
- cable con clavija redonda (destino ignoto)
- 3 cables idénticos, todos distintos

Cables útiles ahora mismo: 0

Cincuenta cables. De noche se reproducen, pero solo los inútiles.

En un cajón con cincuenta cables distintos nunca está el que necesitas ahora.

— Anónimo, *folclore informático*

Toda casa tiene el Cajón. Un ecosistema de cables enredados que se reproducen de noche: cargadores de móviles que nunca has tenido, alargadores de módems del siglo pasado, un cable misterioso con una clavija redonda que no entra en nada.

Cincuenta cables. Cero con el conector correcto. Estadísticamente imposible y, sin embargo, pasa todas las veces. El cable que necesitas existe, claro: está en el cajón del vecino, que a su vez busca el tuyo.

Propuse a la comunidad de vecinos un mercadillo de cables, con intercambios los sábados por la mañana. Fracasó porque nadie quería ceder su cable inútil. «Nunca se sabe», decían. El lema oficial de todo cajón.

Consejo de verdad: etiquetad los cables con un trozo de cinta y tirad sin piedad los que no reconozcáis desde hace dos años.

LEY DEL PENDRIVE CUANTICO



El conector USB entra siempre al tercer intento: la primera vez está del revés, la segunda también, la tercera estaba bien desde el principio.

— Anónimo, *folclore informático*

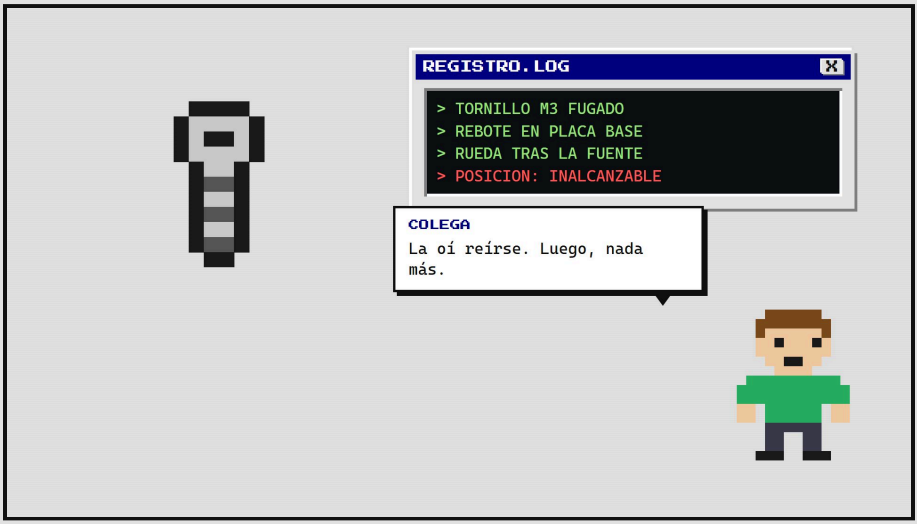
El conector USB clásico tiene dos lados. Matemáticamente, al primer intento tienes un cincuenta por ciento de probabilidades de acertar. En la práctica, la probabilidad es cero. El conector existe en un estado de superposición: está mal en los dos sentidos hasta que le das la vuelta dos veces.

He calculado que a lo largo de mi carrera he pasado unos tres días seguidos de mi vida dándole la vuelta a pendrives. Los considero vacaciones no disfrutadas.

Luego llegó el conector reversible, y la humanidad lo celebró. Durante una tarde. Después nos dimos cuenta de que los cables reversibles no cargan todos a la misma velocidad, no transmiten todos datos y no encienden todos el monitor. El caos solo se ha trasladado al interior del cable.

Consejo: el logo USB del conector, normalmente, va hacia arriba. Normalmente.

LEY DEL TORNILLO FUGITIVO



El tornillo que se te escurre de los dedos mientras abres el ordenador cae siempre en el punto más inaccesible de la caja.

— Anónimo, *folclore informático*

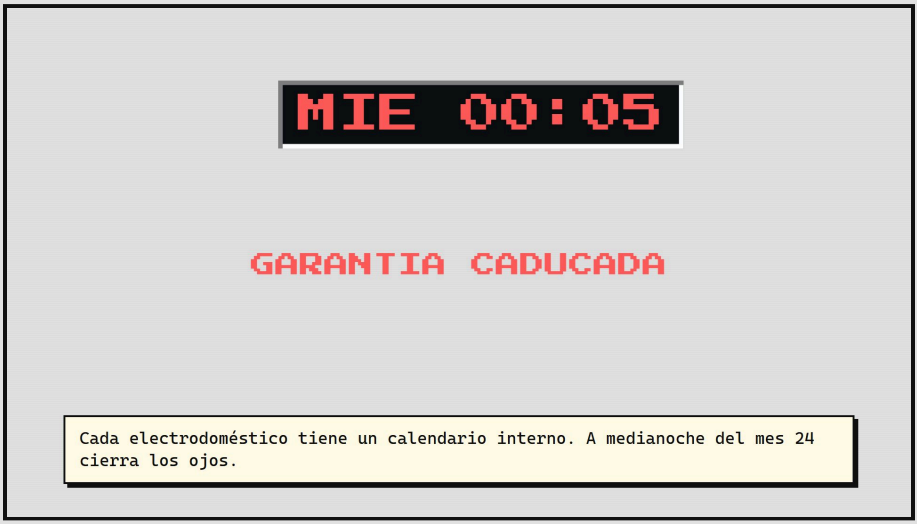
Dentro de un ordenador hay miles de sitios cómodos donde podría aterrizar un tornillo. Él elige siempre el único debajo de la placa base, detrás de la fuente de alimentación, en una rendija diseñada por alguien que odiaba los dedos humanos.

El técnico ejecuta entonces la antigua danza de la caja: la inclina, la sacude, le da la vuelta, escucha. El tornillo rueda con un sonido metálico, como una risa. Luego calla. Nadie volverá a verlo jamás, pero el ordenador ahora hace un ruido nuevo cuando lo mueves.

En mi taller guardamos un bote de tornillos huérfanos recuperados a lo largo de los años. Ninguno ha servido nunca para nada. Lo llamamos «el plan de pensiones».

Consejo de técnico: usad un destornillador imantado y un cuenquito para los tornillos. Y no trabajéis nunca encima de la moqueta.

LEY DE LA GARANTIA CADUCADA



Un dispositivo se estropea el día después de que caduque la garantía, salvo cuando se estropea el día antes y has perdido el tique.

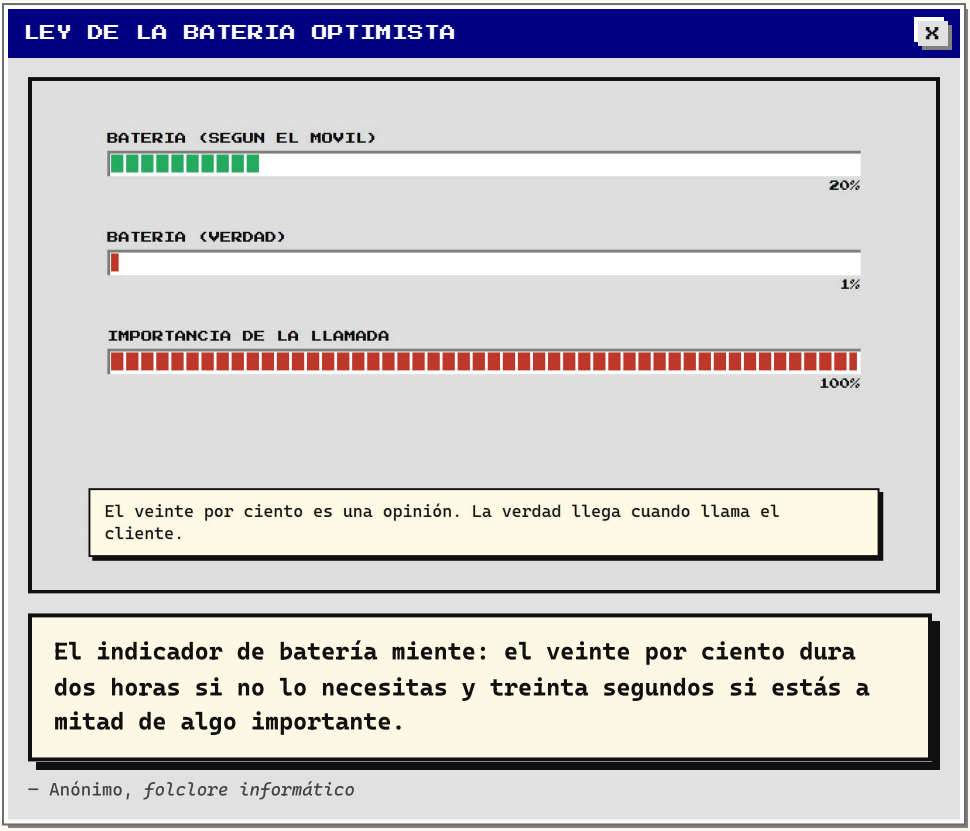
— Anónimo, *folclore informático*

Dentro de cada electrodoméstico moderno hay un calendario. No está escrito en el manual, pero está. Cuenta los días con la paciencia de un monje y, a medianoche del mes veinticuatro, cierra los ojos en paz.

Una clienta mía guardaba todos los tiques desde 1998 en una caja de zapatos, catalogados por colores. El día que la impresora se estropeó en garantía, la caja estaba en el trastero. El trastero estaba inundado. El tique se había convertido en una nubecilla blanca.

La garantía, en resumen, existe: es como el unicornio del servicio de atención al cliente. Todos hablan de ella, alguno jura haberla visto.

Consejo de técnico: fotografiad el tique nada más comprar y guardadlo en la nube. El papel térmico se borra; las fotos, no.



El indicador de batería miente: el veinte por ciento dura dos horas si no lo necesitas y treinta segundos si estás a mitad de algo importante.

— Anónimo, *folclore informático*

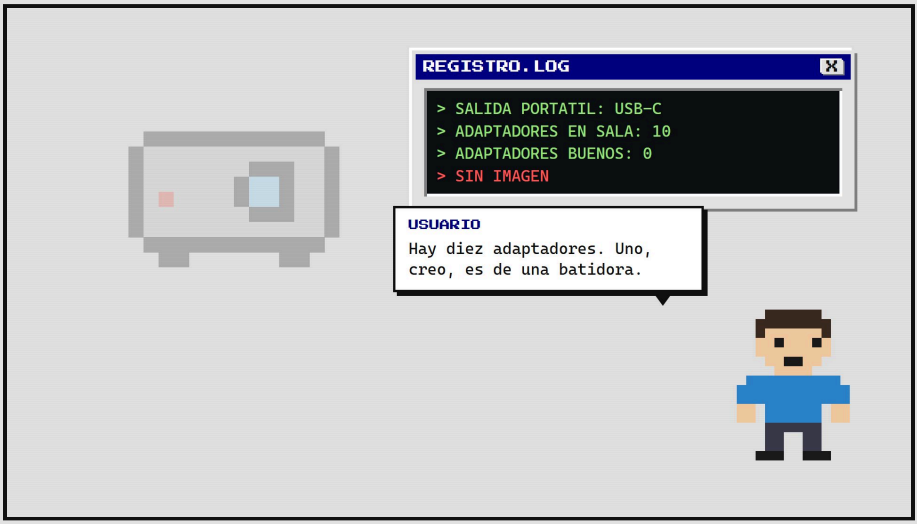
El porcentaje de la batería no es un dato. Es una opinión. Un sentimiento. El móvil al veinte por ciento te dice «tranquilo, llegamos», con el tono de quien conduce en reserva por la autopista.

Luego abres el mapa para encontrar al cliente, y el porcentaje baja como un ascensor sin cables: dieciocho, once, cuatro, negro. El cargador está en casa, cargándose él también, por solidaridad.

Mi tío sostenía que las baterías se recargan solas si las dejas descansar en el cajón. Tiene razón durante más o menos un minuto. Es el minuto más optimista de su día.

Consejo de verdad: las baterías de litio se desgastan y, con la edad, el indicador se vuelve impreciso. Si el dispositivo se apaga con porcentajes altos, es hora de cambiarla.

LEY DEL PROYECTOR HOSTIL



El proyector de la sala de reuniones tiene una entrada distinta de la de tu portátil, y el adaptador adecuado es el único que nadie ha traído.

— Anónimo, *folclore informático*

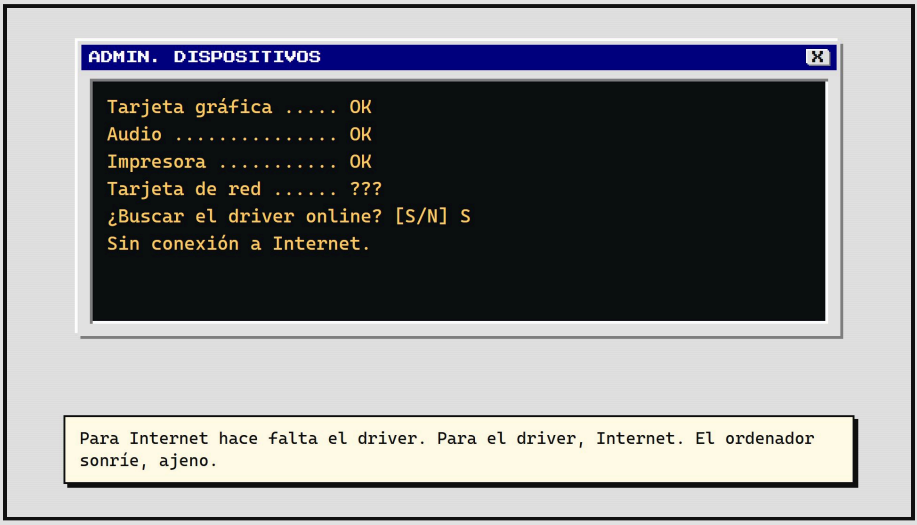
Toda sala de reuniones tiene un proyector colgado del techo como un murciélago electrónico. Tiene una sola entrada, de un tipo que tu portátil abandonó hace tres generaciones. Sobre la mesa, una caja de adaptadores: diez, todos para otros ordenadores, algunos para objetos que no son ordenadores.

Mientras tanto, el público espera. Alguien sugiere «mandar la presentación por correo y que cada uno la vea en el móvil». Es el momento en que la civilización vuelve a empezar desde el papel.

Cuando por fin aparece la imagen, es violeta, está cortada por la derecha y la pantalla del portátil se apaga. Has ganado, pero no sabes cómo.

Consejo de técnico: llevad siempre vuestro adaptador y una copia de la presentación en PDF en un pendrive. Y llegad quince minutos antes.

LEY DEL DRIVER IMPOSIBLE DE ENCONTRAR



Después de reinstalar el sistema, el único driver que falta es el de la tarjeta de red, que haría falta para descargar los demás drivers.

— Anónimo, *folclore informático*

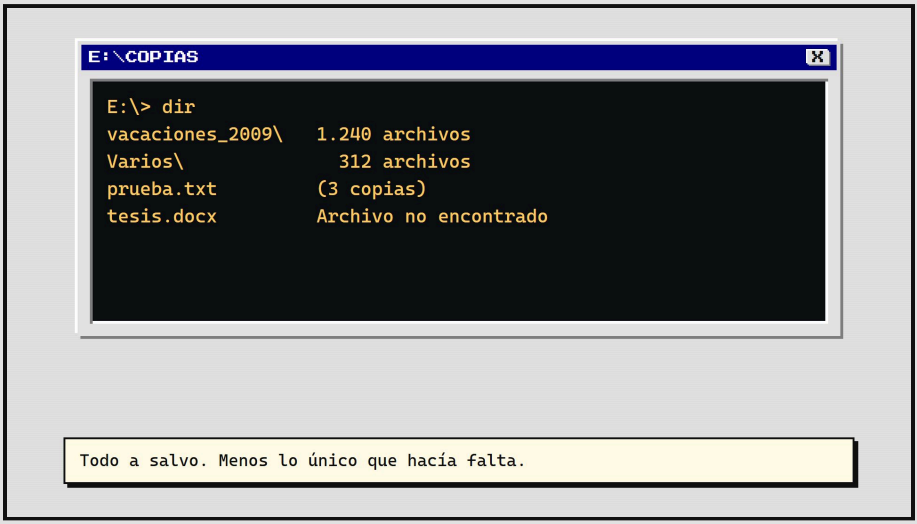
Es la paradoja perfecta, digna de un relato de ciencia ficción. Para tener Internet necesitas el driver. Para tener el driver necesitas Internet. El ordenador recién formateado te mira con inocencia, como un recién nacido que todavía no sabe hablar con el router.

El técnico parte entonces hacia la Gran Expedición: otro ordenador, un pendrive, una web del fabricante que ha cambiado de estructura seis veces. El driver se llama algo así como «paquete_final_v2_nuevo_DEFINITIVO.zip» y pesa más que el sistema operativo.

Mientras tanto, en la habitación de al lado, el cliente pregunta si, «ya que estás», puedes recuperar también las fotos de la boda.

Consejo de verdad: antes de formatear, descargad en un pendrive los drivers de red de vuestro modelo. Es la caja de herramientas del día después.

LEY DE LA COPIA QUE NO HICISTE



De todas las copias de seguridad posibles, la que necesitas es siempre la única que no hiciste.

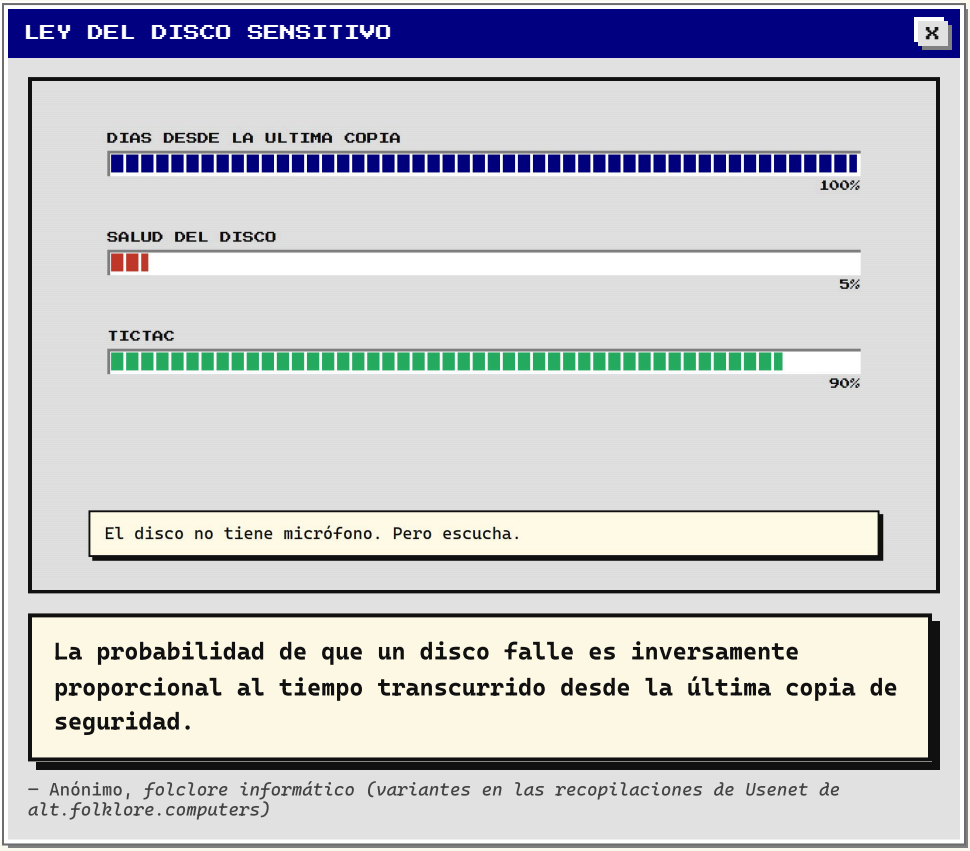
— Anónimo, *folclore informático*

Tienes copia de las fotos de las vacaciones de 2009. Tienes copia de la carpeta «Varios». Tienes tres copias idénticas de un archivo llamado «prueba.txt». Luego el disco muere, y el único archivo que de verdad necesitas es el trabajo de fin de grado, que estaba en el escritorio «solo por hoy».

La copia de seguridad es una forma de superstición al revés: si la haces, no pasa nada y crees que has perdido el tiempo. Si no la haces, el universo organiza una demostración práctica.

Tengo un cliente que hacía la copia en un pendrive, que tenía enchufado al ordenador, que estaba bajo la ventana, bajo la cual pasó una tormenta. Ahora tiene dos cosas mojadas en lugar de una.

Consejo de verdad: regla del 3-2-1. Tres copias, en dos soportes distintos, una de ellas fuera de casa o en la nube.



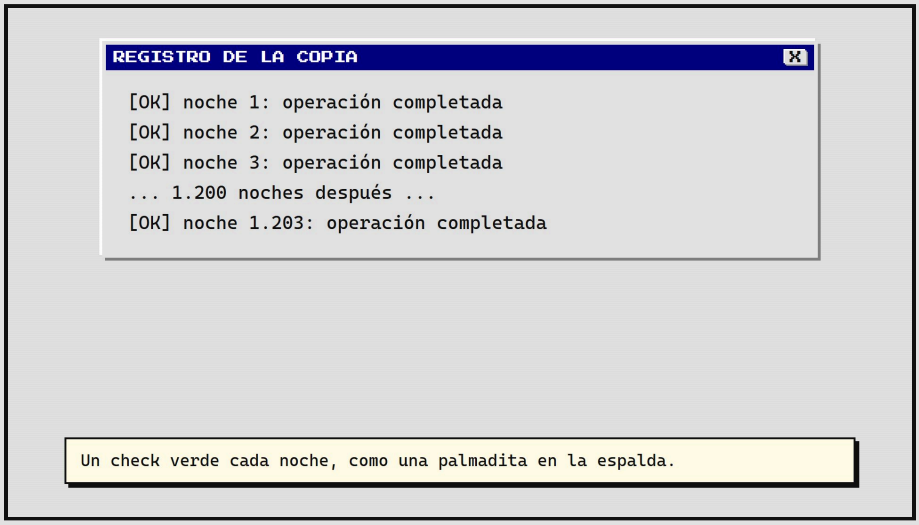
El disco duro escucha. No tiene micrófono, pero escucha. El día después de la copia completa gira feliz, silencioso, eterno. Tras seis meses sin copias empieza a hacer un pequeño tictac, como un reloj de película de suspense.

Los técnicos veteranos decían que el tictac era «el cabezal que saluda». En realidad era el cabezal despidiéndose para siempre.

Solo hay una manera de engañar al disco: hacer la copia sin que se entere. Con discreción, de noche, automáticamente. El disco descubre la copia después y, a esas alturas, se le quitan las ganas de romperse. No es ciencia. Pero los resultados son excelentes.

Consejo de sistemista: programad una copia de seguridad automática. La que depende de vuestra memoria no es una copia, es una esperanza.

LEY DE LA RESTAURACION NUNCA PROBADA



Una copia de seguridad funciona siempre, hasta el día en que intentas restaurarla.

— Anónimo, *folclore informático*

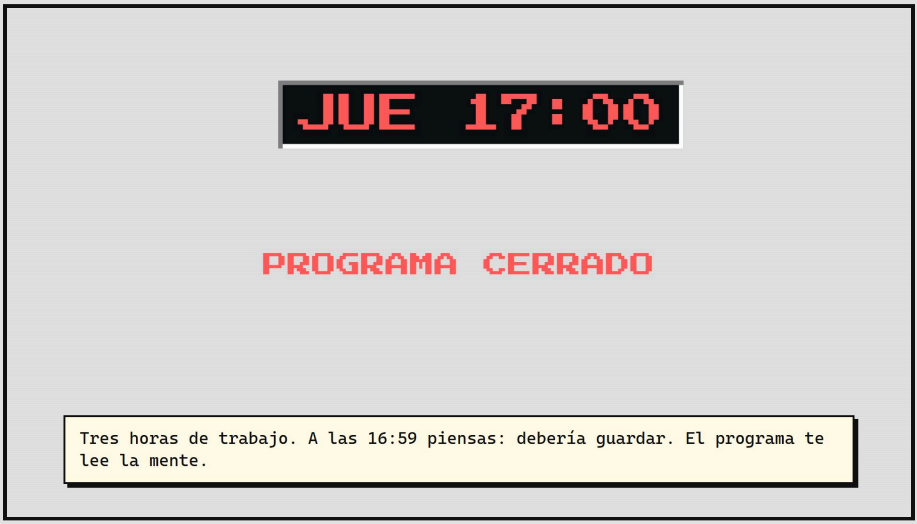
Durante años el programa de copias mostró una marca verde. Cada noche. «Operación completada». Una marca tranquilizadora como una palmadita en la espalda.

Luego llega el día de la necesidad. Abres el archivo y descubres que contiene una sola carpeta vacía, copiada con devoción religiosa durante mil doscientas noches seguidas. El programa nunca mintió: la operación estaba realmente completada. De qué, nadie lo había preguntado.

Una copia no verificada es como un paracaídas que nunca se ha abierto en la tienda. Probablemente está bien. Probablemente.

Consejo de verdad, de técnico: de vez en cuando intentad restaurar un archivo cualquiera de la copia. Si lo recuperáis, dormid tranquilos. Si no, acabáis de evitar el peor desastre.

LEY DEL GUARDADO FALLIDO



El programa se cuelga siempre después de tres horas de trabajo y treinta segundos antes de que decidas guardar.

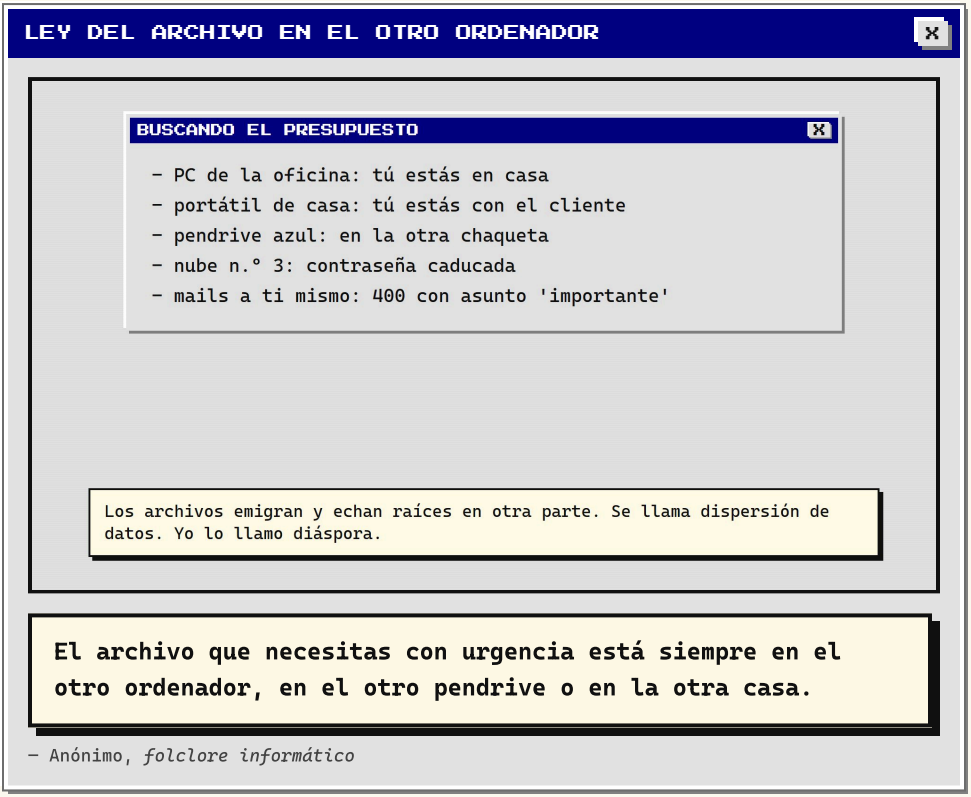
— Anónimo, *folclore informático*

Hay un momento, mientras trabajas, en que piensas: «Debería guardar». Es un pensamiento noble. Pero lo aplazas un minuto, lo justo para acabar la frase. El programa intercepta ese pensamiento, lo encuentra presuntuoso y se cierra.

No se cierra de mala manera. Se cierra con elegancia. Una ventanita dice «El programa ha dejado de funcionar», como un camarero que anuncia que la cocina está cerrada.

Mi tía guardaba cada documento después de cada palabra. Tardaba una tarde en escribir una felicitación, pero nunca perdió nada. La llamábamos «la Caja Fuerte».

Consejo de verdad: aprended la combinación Ctrl+S y usadla como un tic. Y activad el guardado automático, si el programa lo permite.



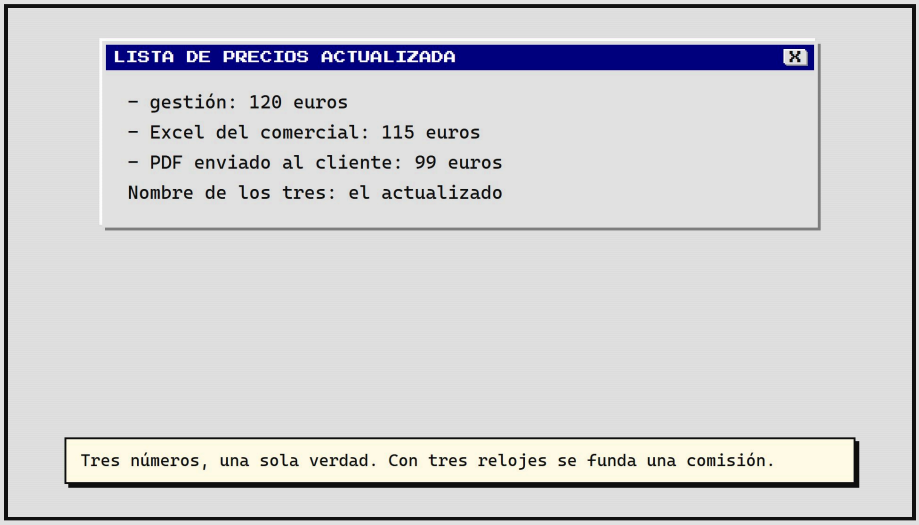
Tienes tres ordenadores, dos pendrives, un disco externo y cuatro servicios en la nube. El archivo del presupuesto está en uno solo de esos sitios. Y siempre es aquel al que no puedes acceder: el ordenador de la oficina, cuando estás en casa; el portátil de casa, cuando estás con el cliente.

Se llama dispersión de datos, pero yo lo llamo diáspora. Los archivos emigran, echan raíces en otra parte, se cambian el nombre a «presupuesto_definitivo_2_copia».

Una vez busqué un documento durante una hora. Estaba adjunto a un correo que me había mandado a mí mismo con el asunto «importante». Había cuatrocientos con el mismo asunto.

Consejo de verdad: elegid un único lugar de trabajo principal, sincronizado, y poned a los archivos nombres claros con fecha. Vuestro yo del futuro os lo agradecerá.

LEY DE SEGAL



Un hombre con un reloj sabe qué hora es. Un hombre con dos relojes nunca está seguro.

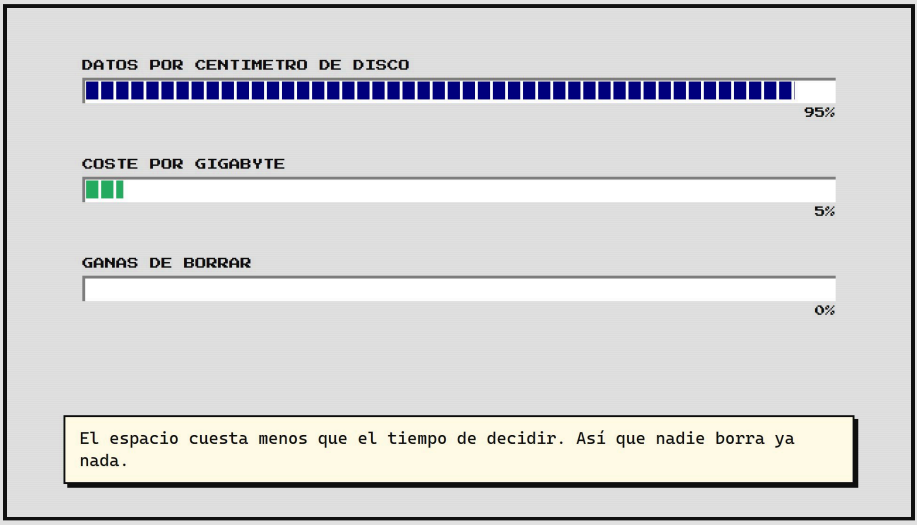
— Anónimo, *Dicho aparecido en el San Diego Union en 1930; luego atribuido por error a Lee Segall y popularizado como «ley de Segal» en los libros de Arthur Bloch (atribución incierta)*

Con un solo reloj no tienes dudas: te equivocas con total serenidad. Con dos relojes que marcan horas distintas empieza la crisis. ¿Cuál tiene razón? Con tres se funda una comisión.

En informática es el drama de las fuentes múltiples. La lista de precios está en el programa de gestión, en el Excel del comercial y en el PDF enviado al cliente. Son tres números distintos, y todos se llaman «el actualizado». Por eso quien diseña sistemas serios habla de fuente única de verdad: un solo sitio del que todos leen, y los demás copian.

Vale también para los servidores: si los relojes de los ordenadores de una red no están sincronizados, los registros cuentan historias imposibles, con respuestas que llegan antes que las preguntas. Yo en la oficina lo he resuelto. He quitado todos los relojes menos uno. Está parado desde 2016, pero ya nadie discute.

LEY DE KRYDER



La densidad de datos que se pueden grabar en un disco magnético crece más deprisa que la potencia de los chips, y el coste por gigabyte se desploma en consecuencia.

— Mark Kryder, *Artículo «Kryder's Law» de Chip Walter, Scientific American, agosto de 2005*

El primer disco duro comercial pesaba como una nevera y contenía unos pocos megabytes. Hoy una tarjeta del tamaño de una uña contiene todas las fotos de las vacaciones, incluidas las cuarenta idénticas de la puesta de sol. Esta es la ley de Kryder: el espacio para los datos se vuelve más denso y más barato a un ritmo vertiginoso.

La consecuencia psicológica es devastadora. Nadie borra ya nada. ¿Para qué elegir, si el espacio cuesta menos que el tiempo de decidir? Tengo clientes con doce copias del mismo PDF, llamadas definitivo, definitivo2, definitivo_de_verdad y definitivo_de_verdad_esta_vez.

En los últimos años el crecimiento se ha frenado, pero la humanidad ya ha encontrado la solución: acumular igualmente. Mi tío guarda los discos rotos en una caja. Dice que los datos siguen dentro, solo que un poco tímidos.

EL CONSEJO DEL SYSADMIN

Dos cosas que casi nadie hace y que todo el mundo debería hacer. La primera: **preguntarle al disco qué tal está**. Casi todos los discos, mecánicos y SSD, llevan un registro de su propia salud llamado SMART. En Windows un programa gratuito como CrystalDiskInfo lo lee en un momento y lo resume con un color: azul va bien, amarillo es un aviso, rojo es un problema. El amarillo no es una opinión, es un preaviso: copiad los datos ese mismo día. Los SSD, a diferencia de los discos antiguos, a menudo fallan sin tictacs de despedida.

La segunda: **no fiarse de los discos hibernados**. Un disco externo encerrado años en un cajón puede no volver a arrancar, y los pendrives olvidados pierden fuelle. Encended los archivos de vez en cuando, comprobad que los ficheros se abren y cada pocos años copiadlo todo en un soporte nuevo. Los datos no envejecen. Sus contenedores, sí.

CAPITULO 3

HELPDESK

Helpdesk: usuarios, claves y otros misterios

Este es el capítulo de mi trabajo de cada día, y por eso el que he releído con más emoción. Están el teléfono que suena, el usuario que no ha tocado nada, el mensaje de error cerrado a la velocidad de la luz, el ordenador que se cura en cuanto el técnico cruza la puerta.

Aclaro algo desde ya: aquí no nos reímos de los usuarios. Nos reímos con ellos. Todo técnico es usuario de otra cosa, y yo delante de la caldera de casa soy exactamente ese señor que jura no haber tocado nada. La navaja de Hanlon, en el primer capítulo, lo dice mejor que yo: detrás de un desastre no suele haber malicia. Hay un clic.

Luego están las contraseñas, que merecen una sección propia: demasiado simples, demasiado complicadas, olvidadas al día siguiente. Y el capítulo se cierra con algunas leyes serias del diseño, las que explican por qué se falla un botón pequeño y lejano, por qué un menú con cuarenta opciones paraliza y por qué medio segundo de espera parece una eternidad. Leerlas ayuda a entender que muchos «errores del usuario» son en realidad errores de quien diseñó la pantalla.

CONTENIDO.DIR

01 Ley del técnico curandero
02 Ley del «no he tocado nada»
03 Ley del enchufe desconectado
04 Ley del mensaje de error sin leer
05 Ley de la avería del viernes por la noche
06 Ley del pariente que entiende de ordenadores
07 Ley de la contraseña olvidada
08 Ley de la contraseña nunca suficiente
09 Principio de Kerckhoffs
10 Leyes de Gilb sobre la falta de fiabilidad
11 Ley de Tesler
12 Ley de Jakob
13 Ley de Fitts
14 Ley de Hick
15 Umbral de Doherty

LEY DEL TECNICO CURANDERO



El problema desaparece en el instante exacto en que llega el técnico, y reaparece en cuanto se va.

— Anónimo, *folclore informático*

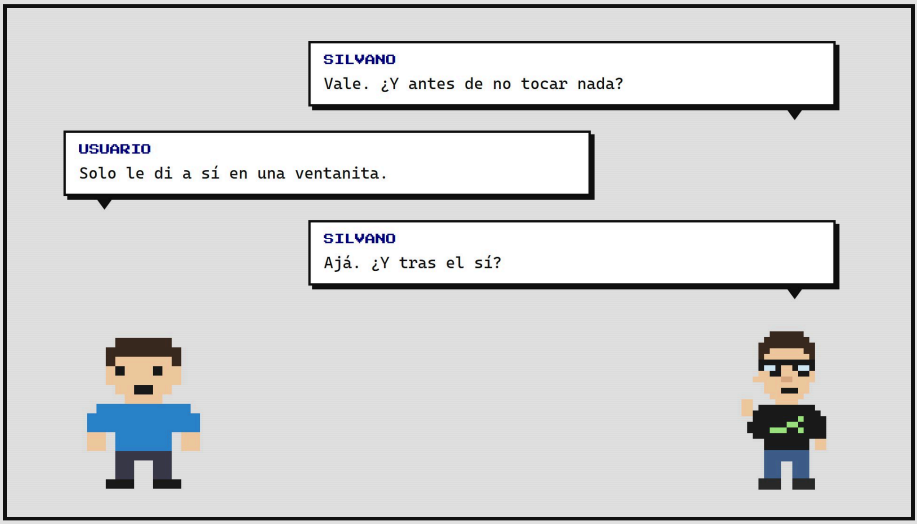
Es mi único superpoder. Entro en la habitación y el ordenador que llevaba tres días mostrando pantallazos azules se cuadra como un soldado en el cuartel. «Antes no iba, se lo juro», dice el usuario. Le creo. He visto esta escena mil veces.

El problema reconoce al técnico como un perro reconoce al veterinario: se porta bien, mueve la cola, lo niega todo. Luego, cuando mi coche sale del aparcamiento, el ordenador vuelve a empezar, con un poco más de rabia por el tiempo perdido.

He pensado en dejar en la oficina una foto mía a tamaño natural. Un cliente lo hizo con una silueta de cartón. Dice que funciona durante más o menos una semana.

Consejo de técnico: cuando aparezca el error, haced una foto a la pantalla con el móvil. Una foto vale más que mil «antes hacía una cosa rara».

LEY DEL «NO HE TOCADO NADA»



El usuario que dice «no he tocado nada» ha tocado algo; la única incógnita es qué.

— Anónimo, *folclore informático*

«No he tocado nada» es la frase más antigua del soporte técnico. La encontraron grabada en el primer ordenador de válvulas, junto a una huella dactilar.

El usuario no miente. En su memoria no ha tocado nada. Solo ha hecho clic en «sí» en una ventanita, ha movido una carpeta «que molestaba», ha instalado un programa gratuito para ver mejor las fotos y ha cambiado un pequeño ajuste que «era feo». Pero tocar, no.

El técnico experto no lo discute. Asiente con la serenidad de un confesor y pregunta: «Muy bien. ¿Y antes de no tocar nada, qué estaba haciendo?». Ahí llega la verdad.

Consejo de verdad: contar con sinceridad los últimos pasos ahorra horas. El técnico no juzga; solo necesita saber por dónde empezar.

LEY DEL ENCHUFE DESCONECTADO

HIPÓTESIS DE LA OFICINA

- virus extranjero
- fuente quemada
- el rayo de hace tres semanas
- complot
- comprar uno nuevo
- mirar el enchufe: 0 votos

También se encendió una vela. Por si acaso.

Antes de comprobar si el enchufe está conectado ya se habrán barajado un virus, una avería de la placa base y una conspiración.

— Anónimo, *folclore informático*

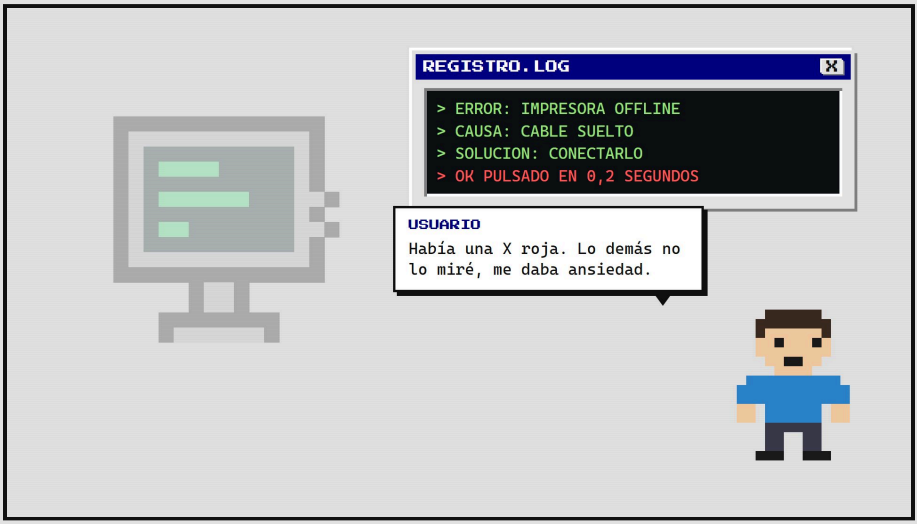
El ordenador no se enciende. Inmediatamente la oficina se divide en escuelas de pensamiento. Hay quien habla de virus rusos, quien de fuente de alimentación quemada, quien del rayo de hace tres semanas que «seguro que dejó algo». Se valora comprar un ordenador nuevo. Se enciende una vela.

Luego llega el técnico, se agacha y vuelve a enchufar la clavija que la empresa de limpieza había desenchufado para conectar la aspiradora. El silencio que sigue es sagrado.

El enchufe desconectado es la causa más humilde de la informática. No tiene código de error, no tiene registro, no tiene encanto. Pero siempre gana.

Consejo de técnico: antes que nada, comprobad lo obvio. Enchufe, regleta encendida, cables del monitor bien conectados. No es una tontería: es método.

LEY DEL MENSAJE DE ERROR SIN LEER



El usuario cierra el mensaje de error antes de leerlo y luego llama al técnico para saber qué ponía.

— Anónimo, *folclore informático*

El mensaje de error es un pequeño poeta incomprendido. Aparece con toda la información necesaria: el problema, la causa, incluso la solución. El usuario lo mira con el horror de quien encuentra una araña en la bañera y pulsa «Aceptar» a una velocidad que desafía la física.

Luego llama: «Ha salido un mensaje». «¿Qué decía?». «No sé, estaba en inglés. O en español. Había una X roja». Es todo lo que queda, como el relato de un avistamiento ovni.

El técnico tiene entonces que reproducir el error, esperando que el poeta vuelva a recitar. El poeta, ofendido, no vuelve.

Consejo de verdad: cuando aparezca un error, no lo cerréis enseguida. Leed con calma, haced una foto o una captura de pantalla. La mitad de las veces la respuesta ya está ahí.



El servidor ha funcionado sin una queja durante once meses y tres semanas. Luego el técnico reserva el hotel en la playa. El servidor lo sabe. No me preguntéis cómo: lo sabe.

El viernes a las seis y un minuto, mientras las llaves giran en la cerradura de la oficina, una lucecita se pone naranja. A las seis y dos, roja. A las seis y tres suena el teléfono del técnico, que ya está en la autopista con la familia y una colchoneta hinchable en el asiento.

Conocí a un sistemista que llevaba seis años sin irse de vacaciones por miedo a esta ley. El servidor nunca se averió. Luego fue al dentista un martes por la mañana.

Consejo de verdad: documentación clara, accesos anotados en un lugar seguro y un compañero de reserva. Estar de guardia no es heroísmo, es organización.

LEY DEL PARIENTE QUE ENTIENDE DE ORDENADORES



Quien en la familia «entiende de ordenadores» pasará cada comida festiva configurando la impresora de un tío.

— Anónimo, *folclore informático*

Basta una vez. Una sola vez en que arreglas el móvil de la abuela, y te nombran de por vida Responsable Informático de la Familia. Sin sueldo, sin vacaciones, con guardia permanente los días festivos.

Llegas a la comida de Navidad y no te da tiempo ni a quitarte el abrigo. «Ya que estás, ¿le echas un vistazo al ordenador?». El ordenador está en el desván, tiene doce años y una capa de polvo que hace de aislante térmico.

Al final del día te has comido un canelón frío, has actualizado cuatro móviles y has explicado seis veces que el wifi no es «Internet». Te pagan con un tarro de mermelada. Está buenísima, reconozcámoslo.

Consejo de verdad: escribid en una hoja las contraseñas del wifi y de las cuentas principales de vuestros parientes, y dejádsela en un lugar seguro. Reduciréis las llamadas a la mitad.

LEY DE LA CONTRASENA OLVIDADA



Una contraseña se olvida siempre al día siguiente de cambiarla, porque el sistema lo exigía.

— Anónimo, *folclore informático*

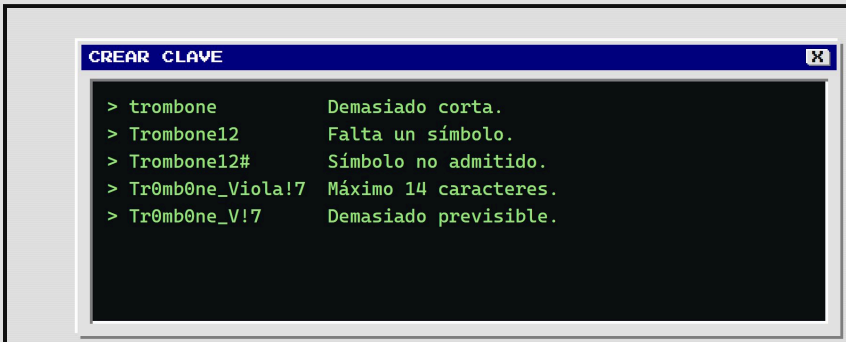
Durante tres años usaste la misma contraseña. La conocías como el nombre de tu hijo. Luego el sistema decide que ha caducado y quiere una nueva, distinta de las doce últimas. La inventas con mimo. Al día siguiente se ha borrado de tu memoria como un sueño al despertar.

Empieza entonces la caza del restablecimiento. «Introduzca la respuesta a la pregunta secreta». El nombre de tu primera mascota. ¿Habías escrito «Toby», «toby» o «TOBY»? El sistema no lo dice. Te bloquea treinta minutos para que reflexiones.

El pósit amarillo de debajo del teclado, mientras tanto, se ríe. Él siempre ha estado ahí.

Consejo de verdad: usad un gestor de contraseñas. Recordáis una sola, robusta, y él custodia las demás. El pósit no cuenta como gestor.

LEY DE LA CONTRASENA NUNCA SUFICIENTE



La web rechaza todas las contraseñas hasta que es imposible recordarla, y luego rechaza también esa porque contiene un carácter no permitido.

— Anónimo, *folclore informático*

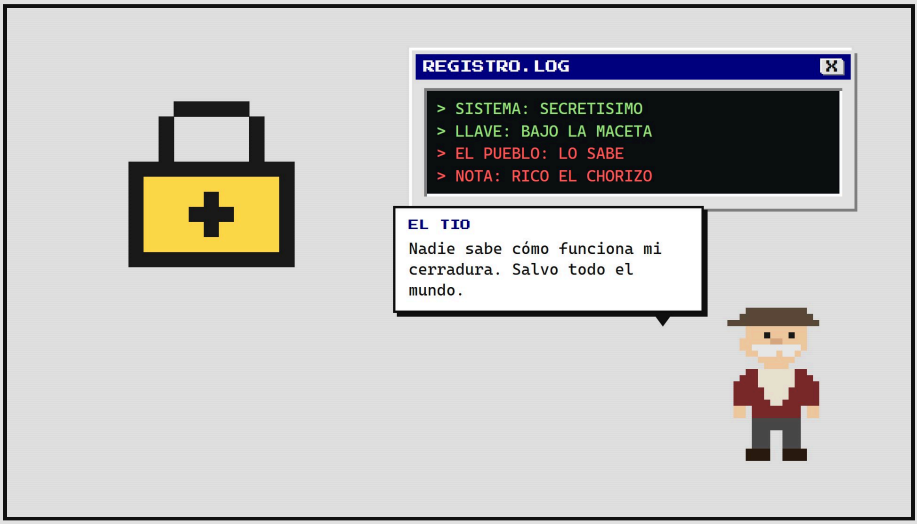
«La contraseña debe contener al menos ocho caracteres». Hecho. «Una mayúscula». Hecho. «Un número». Hecho. «Un símbolo». Hecho. «No puede contener el símbolo que has usado». Ah.

Es un diálogo con un portero muy severo y ligeramente desequilibrado. Cada intento abre una regla nueva, como en un videojuego cuyo nivel final ha escrito un comité.

Al final obtienes una secuencia del tipo «Tr0mb0n_Viol3ta!7», orgullosa como una bandera. La apuntas en una hoja. La hoja va a un cajón. El cajón no tiene cerradura. La seguridad informática ha dado la vuelta completa y ha vuelto al punto de partida.

Consejo de verdad: mejor una frase larga de palabras al azar que una palabra corta llena de símbolos. Es más segura y más fácil de recordar.

PRINCIPIO DE KERCKHOFFS



Un sistema criptográfico debe seguir siendo seguro aunque el enemigo lo sepa todo sobre su funcionamiento: el único secreto debe ser la clave.

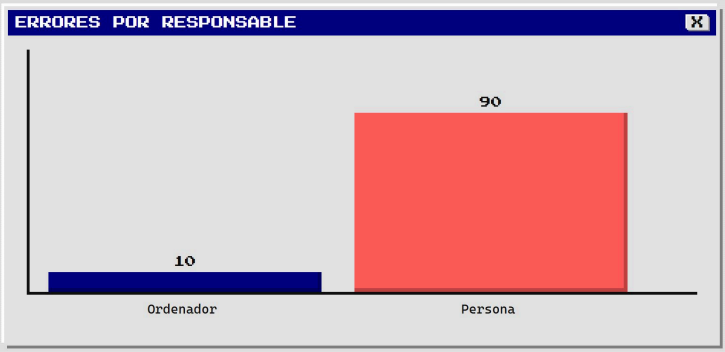
— Auguste Kerckhoffs, *La cryptographie militaire*, 1883

Lo contrario se llama «seguridad por oscuridad»: el programa es seguro porque nadie sabe cómo funciona. Dura exactamente hasta el primer curioso con una tarde libre.

Kerckhoffs razonaba como un militar: los artefactos se capturan, los manuales se copian, los espías se corrompen. Si toda la seguridad está en el secreto del mecanismo, basta un robo para perderlo todo. Si está en la clave, cambias la clave y vuelves a empezar. Por eso los algoritmos serios son públicos y los estudia todo el mundo.

Mi tío protege la bodega con un sistema secretísimo: la llave debajo de la maceta de geranios. El sistema es oscuro, pero el pueblo entero lo conoce. El domingo le dejaron una nota: «Salchichón excelente, buena añada del vino». Seguridad por geranios: desaconsejada.

LEYES DE GILB SOBRE LA FALTA DE FIABILIDAD



El ordenador falla siempre igual. La persona falla con creatividad.

Los ordenadores no son fiables, pero los seres humanos lo son todavía menos; y cualquier sistema que dependa de la fiabilidad humana no es fiable.

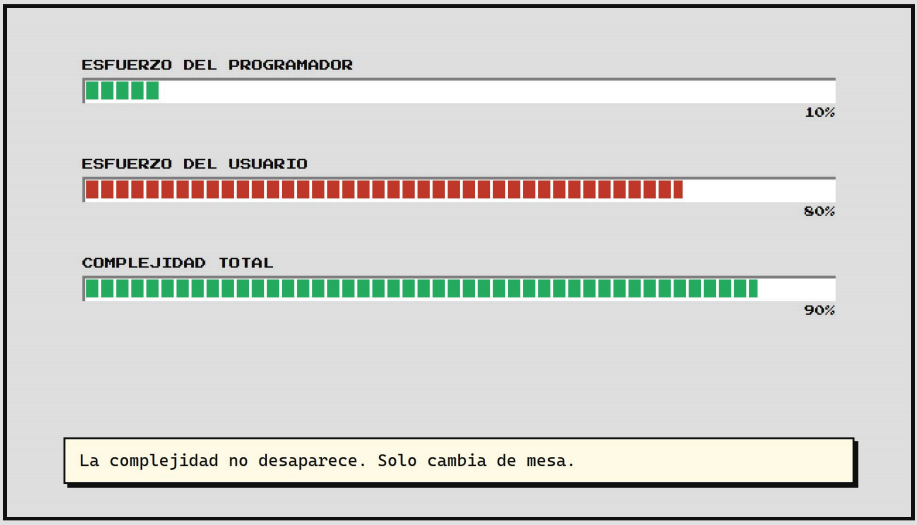
— Tom Gilb, *Leyes difundidas en las recopilaciones de los años 70 (p. ej., Arthur Bloch); numeración y formulaciones varían según las fuentes (atribución en parte incierta)*

El ordenador se equivoca, sí. Pero se equivoca siempre de la misma manera, y tarde o temprano lo entiendes. El ser humano se equivoca con creatividad: hoy la contraseña en un pósit, mañana el pósit en la pantalla, pasado mañana la pantalla fotografiada y compartida en el grupo de la familia.

La lección es de ingenieros: no diseñes sistemas que solo funcionan si todo el mundo está atento. Las personas están cansadas, distraídas, tienen prisa. Los controles automáticos, las confirmaciones y las copias de seguridad existen para ellas.

En una empresa tenían un procedimiento de copia de seguridad infalible: cada tarde un empleado cambiaba el disco. Funcionó durante años. Luego el empleado se jubiló. El disco sigue ahí, metido desde 2015, fiel como un perro en la estación.

LEY DE TESLER



Toda aplicación tiene una cantidad de complejidad que no se puede eliminar: solo se puede decidir si carga con ella el programador o el usuario.

— Larry Tesler, *Ley de conservación de la complejidad, formulada en los años 80 durante su trabajo en interfaces, también en Apple*

La complejidad es como la energía: ni se crea ni se destruye, se desplaza. Un formulario con un solo campo «sencillo» significa que detrás alguien ha escrito mil líneas para entender qué quería decir el usuario.

La ley es una invitación a la cortesía: mejor que sufra un desarrollador una vez que diez mil usuarios todos los días. Por desgracia, la tentación contraria es fuerte, y así nace el formulario con cuarenta campos obligatorios.

En la oficina simplificamos la impresora: ahora tiene un solo botón. Toda la complejidad se me ha trasladado a mí, que recibo veintidós llamadas al día para explicar qué hace ese botón. La conservación funciona. Yo, algo menos.

LEY DE JAKOB



Los usuarios pasan la mayor parte del tiempo en otras webs, así que esperan que la tuya también funcione como las que ya conocen.

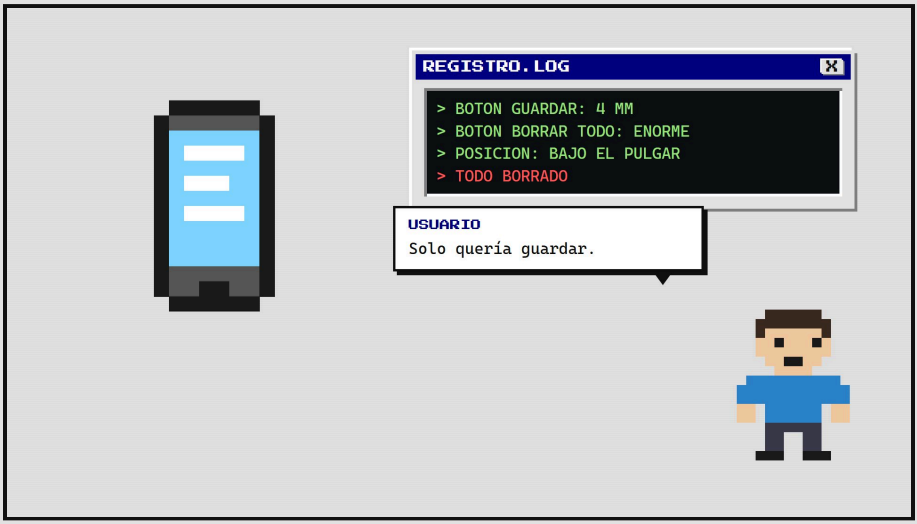
— Jakob Nielsen, *Alertbox, Nielsen Norman Group, 2000*

El carrito está arriba a la derecha. El logo de arriba a la izquierda lleva a la portada. La lupa busca. Nadie lo decidió por ley, pero millones de personas lo han aprendido a fuerza de usarlo en todas partes. La ley de Jakob dice: no luches contra estas costumbres, cabalga sobre ellas.

De vez en cuando llega el diseñador visionario que quiere sorprender. Menú circular que gira con el scroll. Botón de compra escondido en un globo aerostático animado. La web gana un premio. Los usuarios huyen a una web aburrida donde el carrito está donde tiene que estar.

Ser original está muy bien en el color, en las fotos, en los textos. En la navegación, la originalidad es un coste que paga el usuario. Una vez puse el menú abajo a la izquierda por motivos artísticos. El cliente lo encontró al cabo de una semana. Había venido a preguntarme por qué su web tenía una sola página.

LEY DE FITTS



El tiempo para alcanzar un objetivo con un movimiento depende de la distancia y de su tamaño: cuanto más lejos y más pequeño, más se tarda y más se falla.

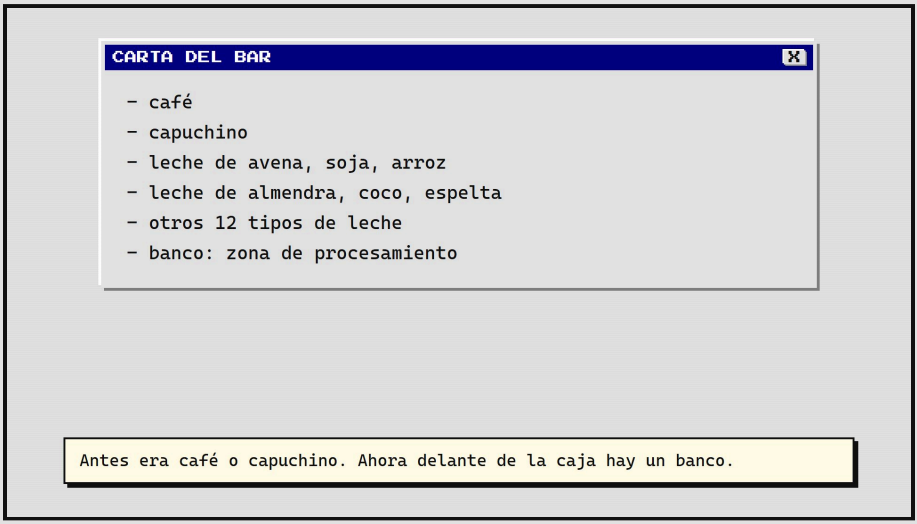
— Paul Fitts, *Journal of Experimental Psychology*, 1954

Un psicólogo de los años cincuenta estudiaba los movimientos de las manos, mucho antes de que existieran los ratones. Su fórmula, sin embargo, parece escrita para quien diseña interfaces: un botón grande y cercano se pulsa enseguida; uno pequeño y lejano es una prueba de habilidad de feria.

Por eso los botones importantes deberían ser grandes, y los peligrosos, lejanos. En cambio, a menudo pasa lo contrario: la tecla «Guardar» es un sello de correos y la de «Borrar todo» es grande como una puerta, justo debajo del pulgar. En el móvil se convierte en un deporte de riesgo.

Los bordes de la pantalla son objetivos infinitos, porque el ratón se detiene ahí solo. Lo demuestra mi tío, que siempre abre el menú Inicio lanzando el cursor abajo a la izquierda con un golpe de muñeca. Dice que es una técnica de pesca. No falla nunca.

LEY DE HICK



El tiempo necesario para decidir crece con el número de opciones disponibles (de forma logarítmica, no lineal).

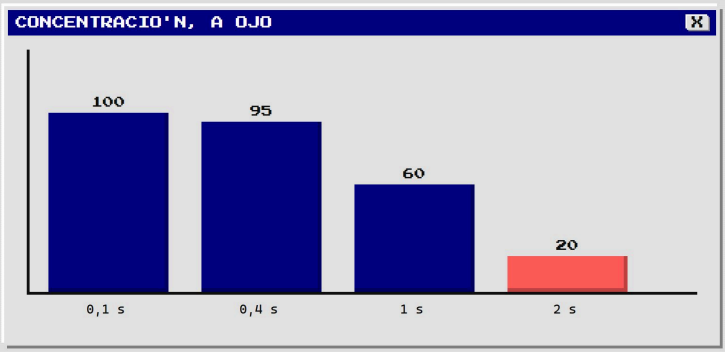
— William Edmund Hick (y Ray Hyman), *Quarterly Journal of Experimental Psychology, 1952; estudio de Hyman, 1953 (llamada también ley de Hick–Hyman)*

Un menú con tres opciones se lee y se elige en un momento. Un menú con cuarenta opciones, cada una con su submenú, es una tarde de tu vida. La ley de Hick dice que cuantas más opciones pones delante de una persona, más tarda en decidir. Crece menos que proporcionalmente, pero crece.

Es el motivo por el que los buenos programas esconden los ajustes avanzados y muestran solo lo necesario. Y es el motivo por el que la ventana de opciones de ciertos programas, con treinta y dos pestañas y doscientas casillas, provoca en el usuario una mirada fija hacia el horizonte.

Lo veo cada día en el bar. Antes era «café solo o con leche». Ahora hay una carta con dieciocho tipos de leche. El camarero ha puesto un banco delante de la caja. Lo llama zona de procesamiento.

UMBRAL DE DOHERTY



Pasado el umbral la mente empieza a divagar. Y cuando divaga, se va a internet.

La productividad crece claramente cuando el ordenador responde al usuario en menos de unos 400 milisegundos: por debajo de ese umbral ninguno de los dos espera al otro.

— Walter J. Doherty y Arvind J. Thadhani, *«The Economic Value of Rapid Response Time»*, *IBM, 1982 (el valor de 400 ms lo citan sobre todo fuentes posteriores)*

A principios de los años ochenta se pensaba que dos segundos de espera eran aceptables. Un estudio de IBM sostuvo lo contrario: si el ordenador responde en menos de medio segundo, más o menos, el usuario sigue concentrado y trabaja mucho más. Por encima, la mente empieza a divagar. Y cuando divaga, se va a Internet.

Por eso las interfaces modernas muestran algo enseguida: una animación, una barra, un esqueleto gris de la página. No es más rápido, pero parece vivo. El usuario no quiere el resultado inmediato. Quiere saber que alguien le está escuchando.

Apliqué el umbral de Doherty al soporte telefónico. Respondo en menos de cuatrocientos milisegundos con «Sí, dígame». Luego, evidentemente, tardo una hora en resolverlo. Pero el cliente se siente atendido. Todo está en la barra de progreso emocional.

EL CONSEJO DEL SYSADMIN

Gran parte del tiempo del soporte se va en entender qué ha pasado. Podéis regalárselo a vuestro técnico, que os recordará con cariño. Cuando algo no va, preparad estos cinco datos antes de llamar:

- **Qué estabais haciendo**, paso a paso, incluido el «no he tocado nada» con detalle.
- **El mensaje exacto**, fotografiado. En Windows 10 y 11 las teclas Windows + Mayús + S capturan una parte de la pantalla.
- **Desde cuándo** ocurre, y si antes funcionaba.
- **Qué ha cambiado** mientras tanto: una actualización, un programa nuevo, una mudanza de mesa.
- **Si ocurre siempre** o solo a veces, en un solo ordenador o en todos.

Con estas respuestas, muchas averías se resuelven por teléfono. Las demás se resuelven más rápido, porque el técnico llega ya con la herramienta adecuada en la mano.

CAPITULO 4

RED

Redes e Internet: el hilo invisible

La red es lo más usado y menos comprendido de la casa moderna. Todos la quieren, nadie sabe dónde está. ¿En el router? ¿En la pared? ¿En el aire? Mi vecina sostiene que pasa por la toma del teléfono y que por la noche descansa. No le he llevado la contraria: en ciertos barrios tiene razón.

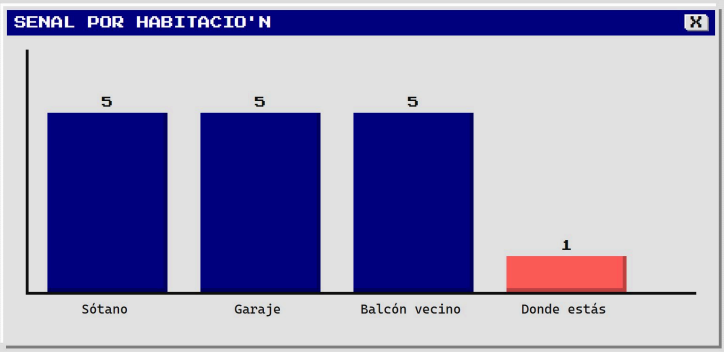
El capítulo se abre con las leyes de la conexión de casa y de la oficina: el Wi-Fi que llega a todas partes menos a donde estás tú, la descarga que muere en el noventa y nueve, la videollamada que se congela cuando te toca hablar. Luego pasa a las leyes del gran Internet, las nacidas entre foros, periódicos online y chats de grupo. Aquí la tecnología cuenta poco y las personas muchísimo: quien corrige equivocándose, quien titula con signo de interrogación, quien intenta esconder algo y lo hace famoso.

Son leyes que valen también fuera de la pantalla. La de Brandolini, por ejemplo, la he visto en acción en una cena, en unas fiestas de pueblo y una vez en una sala de espera: desmentir una tontería siempre cuesta más que decirla. Este capítulo no la desmiente. La documenta.

CONTENIDO.DIR

01 Ley del wifi caprichoso
02 Ley del router que hay que reiniciar
03 Ley de la videollamada decisiva
04 Ley de la descarga al 99 %
05 Ley de la web que funciona en todas partes
06 Ley de Metcalfe
07 Ley de Nielsen del ancho de banda
08 Principio de robustez
09 Ley de Cunningham
10 Ley de Skitt
11 Ley de los titulares de Betteridge
12 Ley de Poe
13 Ley de Godwin
14 Efecto Streisand
15 Ley de Brandolini

LEY DEL WIFI CAPRICHOSO



Rayitas de cobertura. El Wi-Fi es un gato: va donde quiere.

El wifi funciona perfectamente en todas las habitaciones de la casa salvo en la que estás tú.

— Anónimo, *folclore informático*

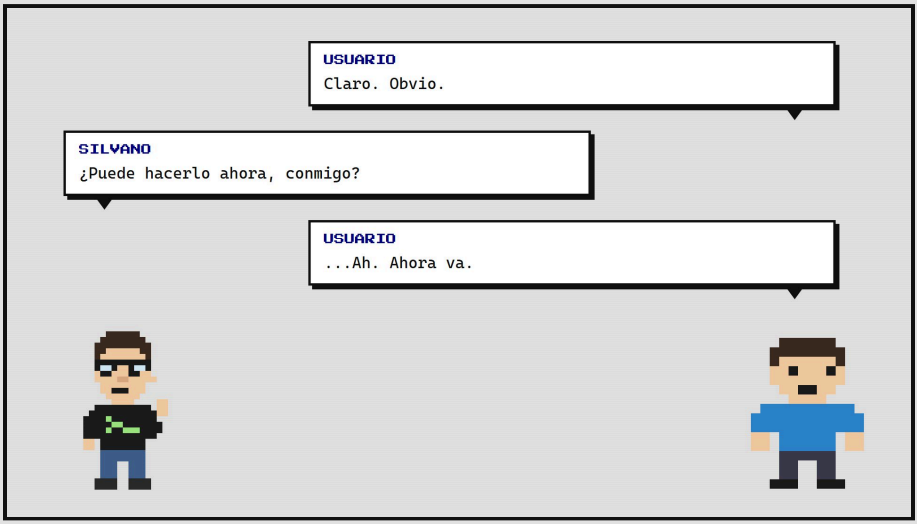
La señal wifi es un gato. Va donde le da la gana. En el sótano llega de maravilla, en el garaje también, en el balcón del vecino se ve en alta definición. Pero tú estás en el salón, y en el salón hay dos rayitas que parpadean como una petición de auxilio.

Entonces empieza el rito. Te mueves medio metro. Levantas el móvil al cielo como una ofrenda. Te sientas en el brazo del sofá en una postura que el fisioterapeuta desaconseja. Funciona. Te mueves un centímetro. Ya no funciona.

Tengo un cliente que metió el router en un armario de madera maciza «para no verlo». Ahora tampoco lo ve el wifi.

Consejo de técnico: el router va en una posición central, alta y despejada. Paredes gruesas, espejos y acuarios son sus enemigos naturales.

LEY DEL ROUTER QUE HAY QUE REINICIAR



Antes de llamar al servicio técnico nadie reinicia el router; después de llamar, el problema se resuelve reiniciando el router.

— Anónimo, *folclore informático*

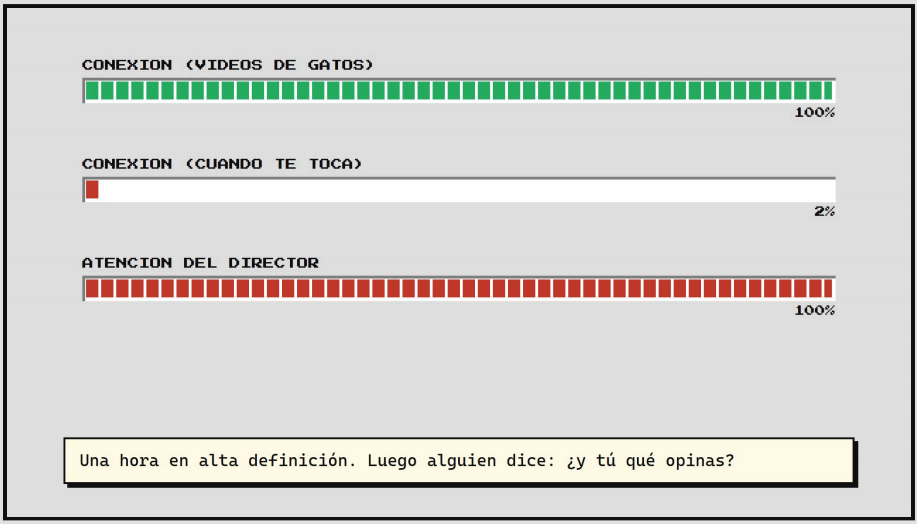
Reiniciar el router es el sacramento laico de la era digital. Todo el mundo conoce su poder, y nadie cree de verdad en él hasta que se lo dice alguien con auriculares y un número 900.

«¿Ha probado a apagar y volver a encender?». «Claro». «¿Puede hacerlo otra vez ahora conmigo?». Silencio. Ruido de muebles arrastrados. Lucecitas que se apagan. Lucecitas que se encienden. «Ah. Ahora va».

¡Hermanos y hermanas de la conexión! El router no es un aparato. Es un estado de ánimo que de vez en cuando hay que resetear, como nosotros los lunes por la mañana.

Consejo de técnico: apagad, esperad al menos treinta segundos, volved a encender y dadle al router un par de minutos para arrancar. Si el problema vuelve a menudo, hay que buscar la causa: el reinicio es una tirita, no la cura.

LEY DE LA VIDEOLLAMADA DECISIVA



La conexión aguanta horas de vídeos de gatos, pero se cae exactamente cuando en la videollamada importante te toca hablar a ti.

— Anónimo, *folclore informático*

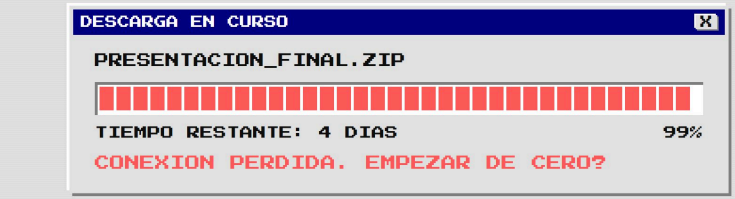
Durante toda la reunión la conexión ha sido impecable. Has visto a los compañeros en alta definición, has oído respirar al director, has contado las plantas que tiene detrás del escritorio. Luego alguien dice: «¿Y tú qué opinas?».

En ese preciso instante tu cara se congela en una expresión indescifrable, a medio camino entre el asombro y el estornudo. La voz se vuelve la de un robot debajo del agua. Los demás te ven así durante cuarenta segundos y deciden pasar al siguiente punto.

Cuando vuelves a estar conectado, la reunión ya ha decidido. Sobre ti. En tu ausencia.

Consejo de verdad: para las llamadas importantes, si podéis, conectad el ordenador al router con un cable de red. El wifi es cómodo; el cable, fiable.

LEY DE LA DESCARGA AL 99 %



Al uno por ciento la conexión es una roca. Al noventa y nueve recuerda que tenía un compromiso.

Una descarga o una subida se interrumpe siempre al noventa y nueve por ciento, nunca al uno.

— Anónimo, *folclore informático*

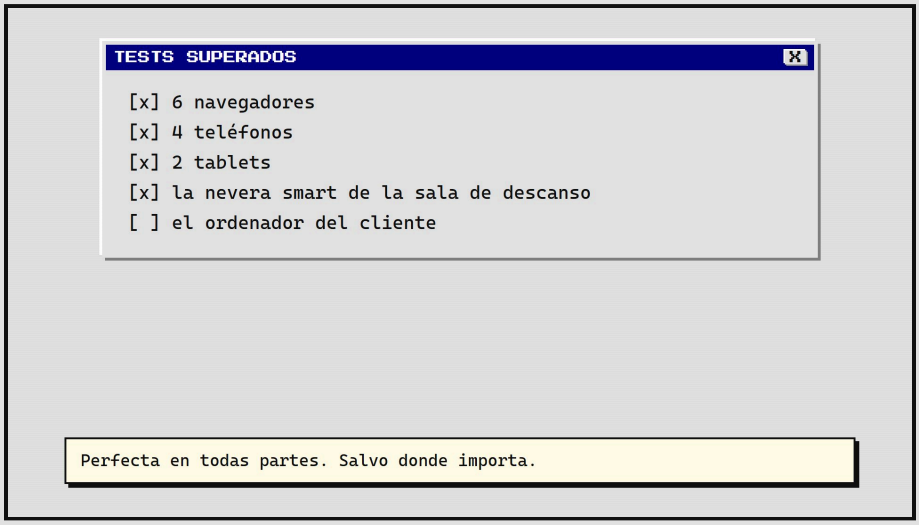
Ninguna descarga ha fallado jamás al uno por ciento. Sería un fracaso honesto, sin dolor. No: la barra avanza despacio durante una hora, llega al noventa y nueve, te deja vislumbrar la tierra prometida y entonces aparece el aviso «error de red».

Los científicos lo llaman crueldad. Los técnicos lo llaman «el último uno por ciento», que, como es sabido, dura tanto como el noventa y nueve restante. Ahí es donde el ordenador hace los cálculos serios, dicen. En realidad solo está decidiendo si te quiere.

He visto a hombres hechos y derechos mirar fijamente una barra parada en el noventa y nueve durante veinte minutos, sin respirar, como quien mira un penalti decisivo.

Consejo de técnico: para archivos grandes, usad programas o servicios que permitan reanudar la descarga. Así, si se corta, sigue desde donde se había quedado.

LEY DE LA WEB QUE FUNCIONA EN TODAS PARTES



La web funciona en todos los ordenadores de la oficina menos en el del cliente que tiene que aprobarla.

— Anónimo, *folclore informático*

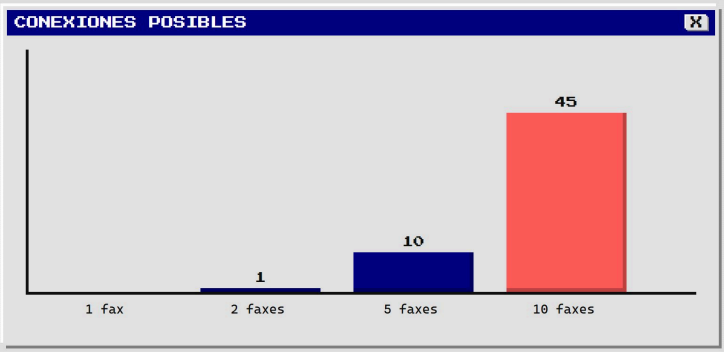
La web se ha probado en seis navegadores, cuatro móviles, dos tabletas y la nevera inteligente de la sala de descanso. Todo perfecto. Luego el cliente la abre desde su ordenador y ve un menú desplazado a la izquierda y un logo del tamaño de un camión.

El ordenador del cliente es un lugar misterioso. Tiene un navegador que solo se actualiza los días pares, seis barras de herramientas instaladas por un sobrino en 2016 y el zoom puesto al ciento cincuenta por ciento «porque se lee mejor».

Y el cliente nunca llama para decir «en mi ordenador no se ve bien». Dice: «La web no funciona». Con el tono de quien anuncia un eclipse.

Consejo de verdad: antes de echarle la culpa a la web, probad a vaciar la caché y a devolver el zoom al cien por cien. Resuelve más problemas de lo que parece.

LEY DE METCALFE



Un fax solo es un adorno que hace ruido. Diez faxes son una sociedad, con cotilleos.

El valor de una red crece con el cuadrado del número de dispositivos (o personas) conectados: cada nodo nuevo aumenta las conexiones posibles de todos los demás.

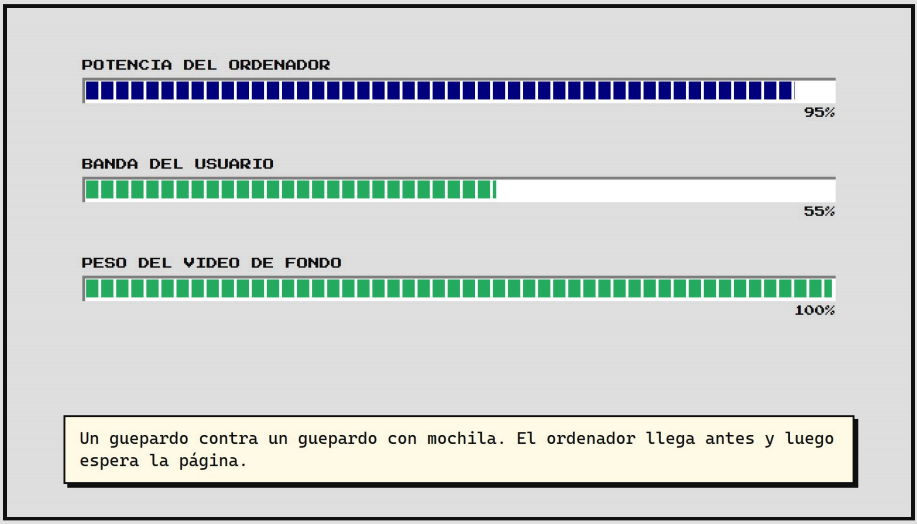
— Robert Metcalfe, *Formulada hacia 1980 para promocionar Ethernet; el nombre «ley de Metcalfe» es de George Gilder, 1993*

Un fax solo no sirve para nada. Es un adorno que hace ruido. Dos faxes pueden hablar entre sí. Diez faxes ya forman una sociedad, con cotilleos y papel térmico. Esta es la ley de Metcalfe: cada nuevo conectado hace crecer el valor de la red para todos, y no poco: al cuadrado.

Es el motivo por el que nadie abandona una red social o un chat de grupo: si te vas, pierdes a todos los demás. El valor no está en el programa, está en la gente atrapada dentro. Felizmente, se entiende.

La otra cara de la moneda la conozco bien. El grupo de WhatsApp de la comunidad de vecinos tiene cuarenta miembros. Según la fórmula, vale muchísimo. En la práctica produce mil seiscientas peleas posibles sobre dónde se aparcan las bicis. Eso también, matemáticamente, es valor.

LEY DE NIELSEN DEL ANCHO DE BANDA



La velocidad de conexión de un usuario avanzado crece alrededor de un 50 % al año: más despacio que la potencia de los ordenadores.

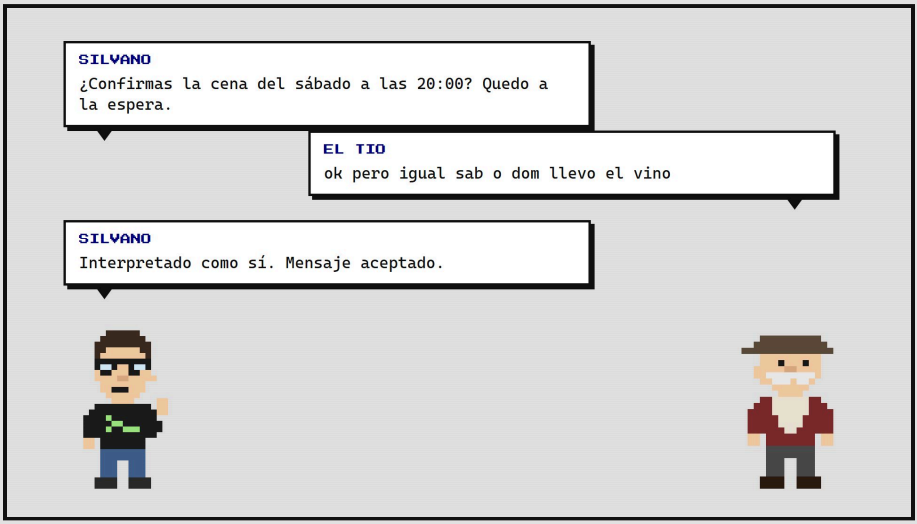
— Jakob Nielsen, *Alertbox, Nielsen Norman Group, 1998 (actualizada en 2008 y 2019)*

Los procesadores se duplican cada dos años, el ancho de banda crece un cincuenta por ciento al año. Parece mucho, pero es como una carrera entre un guepardo y un guepardo con mochila. El ordenador siempre llega antes. Luego se sienta y espera a que cargue la página.

La lección para quien hace webs es sencilla: no des por hecho que todo el mundo tiene fibra. En algún sitio siempre hay un usuario en una casa de campo, con un módem USB apoyado en el alféizar en el único punto donde hay cobertura, que intenta abrir tu vídeo de fondo de noventa megas.

Yo tengo ese cliente. Para mejorar la señal puso el módem en lo alto de un palo de las judías. Funciona. Pero cuando hace viento, Internet se balancea.

PRINCIPIO DE ROBUSTEZ



Sé conservador en lo que envías y liberal en lo que aceptas de los demás.

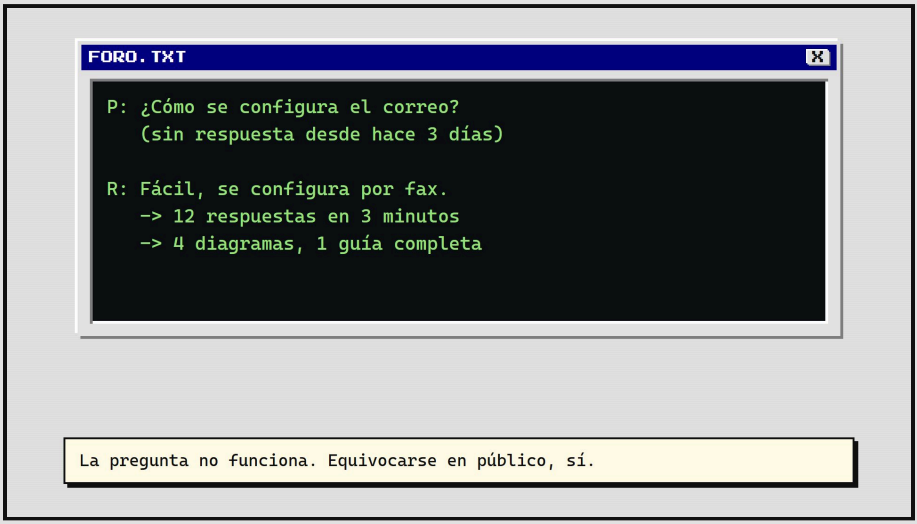
— Jon Postel, *Especificaciones de los protocolos de Internet, RFC 760 (1980) y RFC 793 (1981)*

Es el código de buenas maneras de Internet: cuando hablas, usa el protocolo al pie de la letra; cuando escuchas, sé comprensivo con quien se equivoca en una coma. Gracias a esta amabilidad, máquinas escritas por personas distintas consiguen entenderse.

Tiene su reverso: si todos aceptan errores, nadie los corrige nunca. Tras treinta años de tolerancia, los navegadores leen páginas HTML que parecen escritas por un gato sobre el teclado, y tendrán que seguir haciéndolo para siempre.

He aplicado el principio a mi vida social. Envío mensajes impecables, con puntuación de notario. Acepto cualquier respuesta, aunque sea solo un pulgar. Mi tía me respondió con una berenjena. Lo interpreté como un «sí». Ahora soy testigo de boda.

LEY DE CUNNINGHAM



La mejor manera de obtener la respuesta correcta en Internet no es hacer una pregunta, sino escribir la respuesta equivocada.

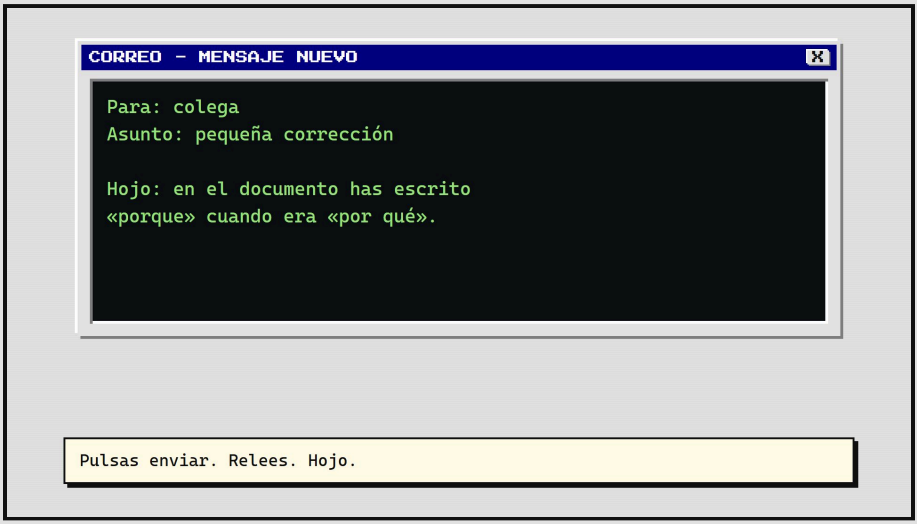
— Steven McGeady, *Bautizada por Steven McGeady en 2010 en honor de Ward Cunningham, que sin embargo ha negado haberla dicho nunca (atribución discutida)*

Preguntas en un foro cómo se configura el servidor de correo: silencio, una bola de paja rodando. Escribes con aplomo un procedimiento equivocado: en tres minutos llegan doce expertos, cuatro diagramas y una corrección firmada por un ingeniero jubilado.

Funciona porque corregir a alguien da más satisfacción que ayudarle. Es un motor de conocimiento alimentado por las ganas de tener razón. La paradoja final: el hombre al que está dedicada dice que nunca la pronunció. La ley se ha demostrado sola.

Yo la uso para todo. Escribí en un grupo que Joomla se instala por fax. Ahora tengo una guía completa, actualizada, con capturas de pantalla. Y también un fax, que un usuario me mandó para demostrar que no funciona.

LEY DE SKITT



Cualquier mensaje que corrige un error ajeno contendrá, casi con toda seguridad, al menos un error propio.

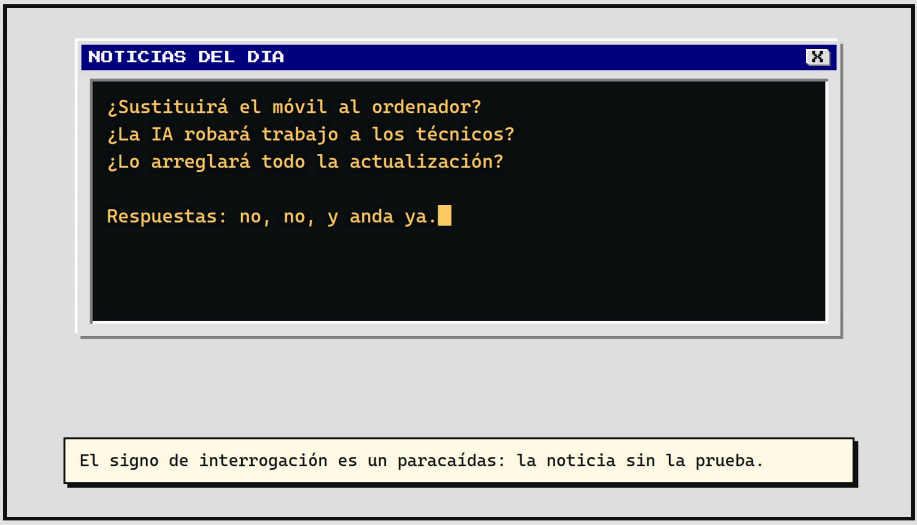
— Anónimo (seudónimo «Skitt»), *Usenet, grupo alt.usage.english, hacia 1999 (atribución incierta; variantes anteriores conocidas como «ley de Muphry»)*

Escribes un correo al compañero: «Ojo, en el documento has puesto haber en lugar de a ver». Pulsas enviar. Relees. Has escrito «Ojo, en el documento as puesto». El compañero responde con una sola palabra, y esa palabra es perfecta.

La ley de Skitt es una ley natural de la corrección. El universo no tolera la soberbia ortográfica: cada vez que te pones en plan maestro, una letra rebelde se suelta del teclado y va a esconderse en el punto más visible de tu mensaje. Vale también para el código: el parche que corrige el bug del compañero introduce un bug tuyo.

Yo, por prudencia, ya no corrijo a nadie. He delegado en un corrector automático. Ayer cambió «servidor» por «servidora». El cliente leyó «la servidora se ha caído» y me preguntó si se había hecho daño.

LEY DE LOS TITULARES DE BETTERIDGE



A cualquier titular de periódico que termine con un signo de interrogación se le puede responder «no».

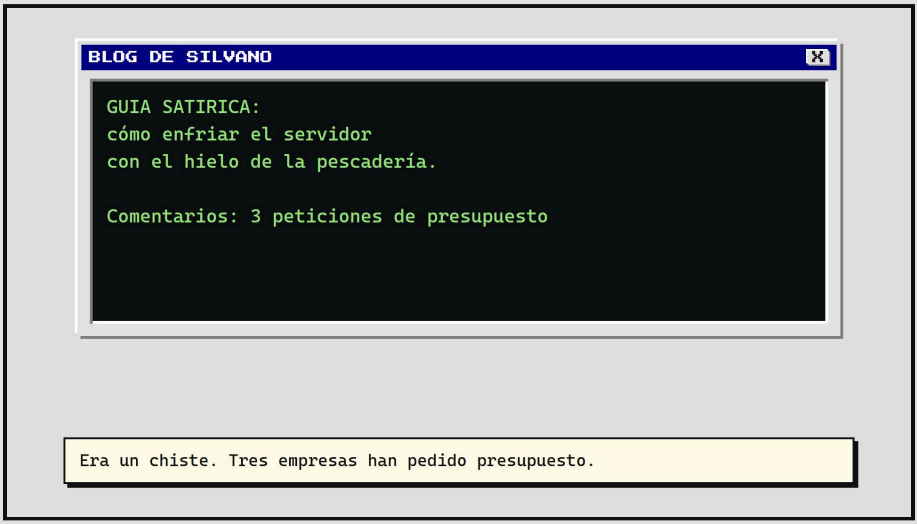
— Ian Betteridge, *Artículo en su blog, febrero de 2009 (idea que ya circulaba con otros nombres en los años 80)*

«¿Sustituirá este nuevo móvil al ordenador?». No. «¿Le robará la inteligencia artificial el trabajo a los técnicos?». No; es más, nos toca configurarla. «¿Resolverá la nueva actualización todos los problemas?». Por favor.

La ley de Betteridge desvela un truco antiguo: si el periodista tuviera pruebas para un sí, escribiría una afirmación. El signo de interrogación es un paracaídas. Permite lanzar la noticia bomba sin tener que demostrarla y cobrar los clics de quien tiene esperanzas.

Hice un experimento en mi blog. Publiqué «¿Volverá a ponerse de moda el disquete?». Tres mil visitas. Luego publiqué «El disquete no volverá a ponerse de moda». Dos visitas, y una era mi madre. El signo de interrogación, lo he entendido, es el verdadero motor de la prensa digital.

LEY DE POE



Sin una señal clara de la intención del autor, una parodia de una postura extrema es indistinguible de una opinión sincera del mismo tipo.

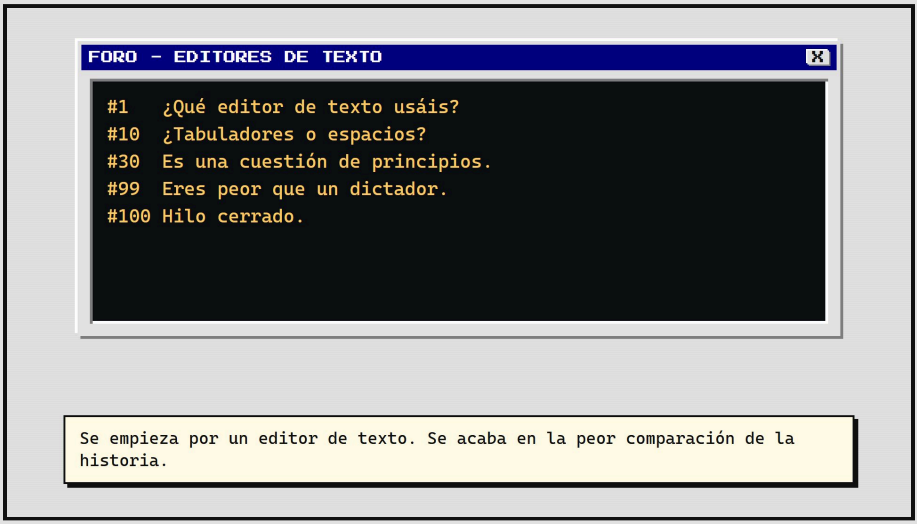
— Nathan Poe, *Foro christianforums.com, 2005 (una advertencia parecida se remonta a Usenet, 1983)*

Escribes en Internet que la mejor forma de proteger la contraseña es imprimirla y pegarla en el monitor, porque así los ladrones no la buscan donde es obvia. Es un chiste. Tres personas te dan las gracias. Una lo hace de verdad. Otra monta un curso de pago.

Esta es la ley de Poe: en línea la ironía no tiene tono de voz, y ciertas opiniones reales son tan extremas que la parodia ya no consigue superarlas. Por eso nacieron los emoticonos. Son nuestra ceja levantada digital.

Una vez publiqué una guía satírica sobre cómo refrigerar el servidor con el hielo de la pescadería. Me escribieron tres empresas para pedir presupuesto. Una ya ha encargado las doradas. Dice que ahora los logs huelen a mar.

LEY DE GODWIN



Cuanto más se alarga una discusión en línea, más se acerca a la certeza la probabilidad de que alguien saque una comparación con Hitler o con los nazis.

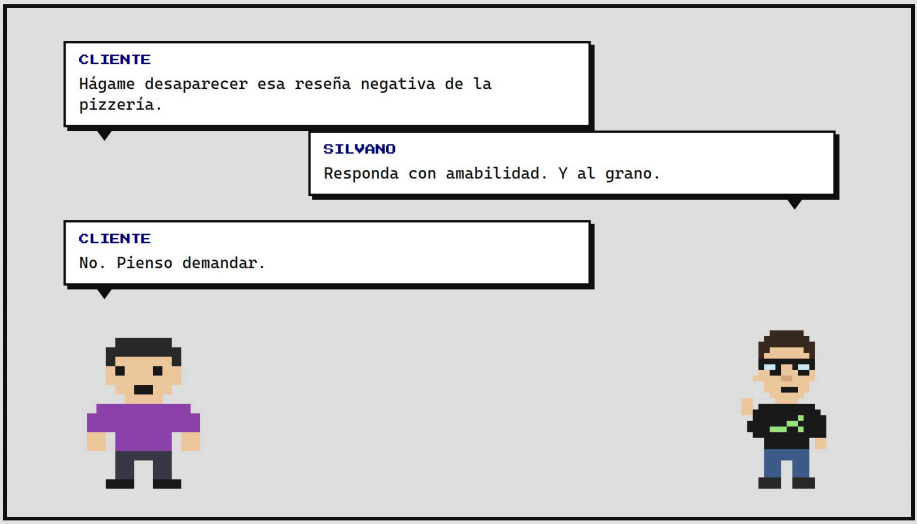
— Mike Godwin, *Usenet*, *1990*

Toda discusión en línea empieza educada. Alguien pregunta qué editor de texto usar. En el décimo mensaje se habla de tabuladores contra espacios. En el trigésimo, de principios morales. En el centésimo, alguien compara al adversario con el peor dictador de la historia. Por un editor de texto.

La ley de Godwin no dice que ocurre: dice que es inevitable, como la gravedad o como el cable HDMI que siempre está del lado equivocado. En los viejos foros valía además una regla no escrita: quien llega primero a esa comparación ha perdido la discusión.

Yo modero un foro de aficionados a las impresoras. Aguantamos trescientos veinte mensajes. Luego uno definió el driver de cierta marca como un régimen totalitario. Para ser sinceros, algo de razón tenía. Pero el hilo se cerró igualmente.

EFECTO STREISAND



Intentar ocultar, eliminar o censurar una información en línea acaba haciéndola mucho más visible y difundida.

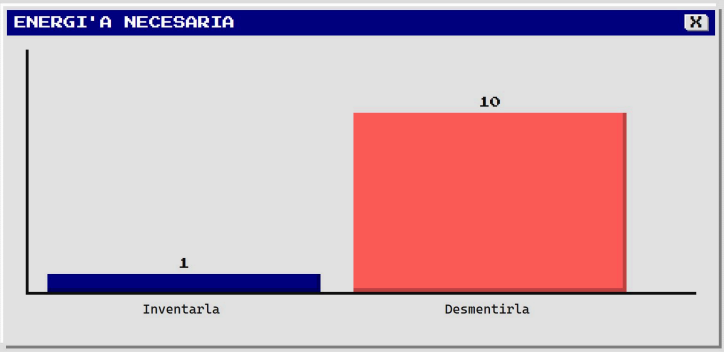
— Mike Masnick, *Techdirt, enero de 2005 (inspirado en un caso de 2003 sobre una foto aérea)*

El nombre viene de un caso de 2003: una demanda para que se retirara de un archivo en línea la foto aérea de una mansión, tomada para documentar la erosión de la costa. Antes de la demanda la foto la habían visto poquísimos. Después, la vio medio mundo. Un bloguero construyó sobre ello el nombre de un fenómeno.

Internet funciona como un gato: si le dices que no mire una caja, solo mirará esa. Cada petición de borrado es un cartel luminoso que dice «aquí hay algo interesante». Copias, archivos y capturas se multiplican más deprisa que cualquier abogado.

Un cliente me pidió hacer desaparecer una reseña negativa de su pizzería. Le aconsejé responder con amabilidad. Prefirió amenazar con querellas. Ahora la reseña es la más leída de la provincia. La pizza, entretanto, ha mejorado. Pero nadie lo sabe.

LEY DE BRANDOLINI



Tres segundos para inventarla. Dos horas, cuatro enlaces y un gráfico para desmentirla.

La energía necesaria para desmentir una tontería es un orden de magnitud mayor que la que se necesitó para producirla.

— Alberto Brandolini, *Mensaje en Twitter, enero de 2013 (conocida también como principio de asimetría de las tonterías)*

Escribir «el 5G marchita los geranios» lleva tres segundos. Explicar por qué no es verdad lleva dos horas, cuatro enlaces, un gráfico y la paciencia de un monje. Mientras tanto, la frase de los geranios ha dado la vuelta a tres grupos de WhatsApp de la familia.

La ley, formulada por un programador italiano, describe una asimetría cruel: inventar cuesta poco, verificar cuesta mucho. Es la razón por la que los bulos ganan las carreras de velocidad, y por la que todo sistemista ha pasado al menos una cena explicando que no, el ordenador no ha cogido el virus por el enchufe.

Yo he encontrado una contramedida. Cuando un cliente me reenvía un bulo, respondo con un bulo opuesto, igual de breve. Se anulan. Ya me he ganado fama de persona informada. También tengo alguna denuncia por geranios, pero está en fase de verificación.

EL CONSEJO DEL SYSADMIN

Cuando «Internet va lento», antes de pelearos con la operadora haced la prueba del cable. Conectad un ordenador directamente al router con un cable de red y medid la velocidad con un test online. Luego repetid la prueba por Wi-Fi, en la habitación donde soléis trabajar.

- Si **también con cable va lento**, el problema es la línea o el router: llamad a la operadora, con los números en la mano.
- Si **con cable va rápido y por Wi-Fi no**, la línea es inocente. El problema es la señal en casa: posición del router, paredes, interferencias.

Dos detalles útiles. Los routers modernos emiten en dos bandas: la de 5 GHz es más rápida pero atraviesa peor las paredes, la de 2,4 GHz llega más lejos pero está más concurrida. Y si los espacios son grandes, un sistema mesh o un punto de acceso conectado por cable funcionan mucho mejor que los repetidores baratos. A veces mover el router dos metros vale tanto como cambiar de contrato.

CAPITULO 5

DEBUG

A la caza del bug

Aquí entramos en el laboratorio. El bug, para quien no lo frecuenta, es un error en el programa. La palabra es más vieja que los ordenadores, pero existe un cuaderno de trabajo de 1947 con una polilla de verdad pegada, encontrada dentro de una máquina. Desde entonces la caza no ha parado nunca, y la polilla ha tenido muchísimos nietos.

Las leyes de este capítulo cuentan el bug en su hábitat natural. Desaparece cuando llega el compañero, se esconde en la última línea que revisas, espera al viernes por la tarde, se presenta diez minutos después de publicar. A veces, si sobrevive lo bastante, los usuarios le cogen cariño y se convierte en una funcionalidad.

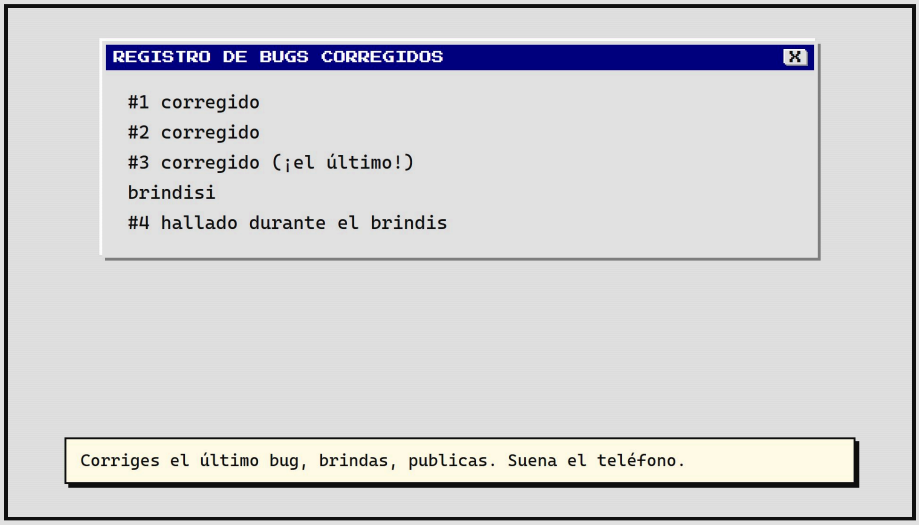
Quien no programa encontrará aquí la respuesta a la pregunta que se hace desde hace años: ¿por qué el programa que uso todos los días de vez en cuando hace cosas raras? Respuesta corta: porque siempre hay otro bug. Respuesta larga: este capítulo.

Quien programa, en cambio, se reconocerá en cada página. Pido perdón de antemano por las noches que os haré recordar.

CONTENIDO.DIR

01 Ley de Lubarsky de la entomología cibernética
02 Ley del «en mi PC funciona»
03 Ley del compañero taumaturgo
04 Ley de la línea inocente
05 Ley de Kernighan
06 Ley de la corrección prolífica
07 Ley del bug evolucionado
08 Ley del bug puntual
09 Ley del viernes por la tarde
10 Ley de la parte no probada
11 Ley del compila, luego funciona
12 Ley del dato de prueba
13 Ley del mensaje oracular
14 Ley del calendario traidor
15 Ley de Linus
16 Ley de Hyrum

LEY DE LUBARSKY DE LA ENTOMOLOGIA CIBERNETICA



Siempre hay otro bug.

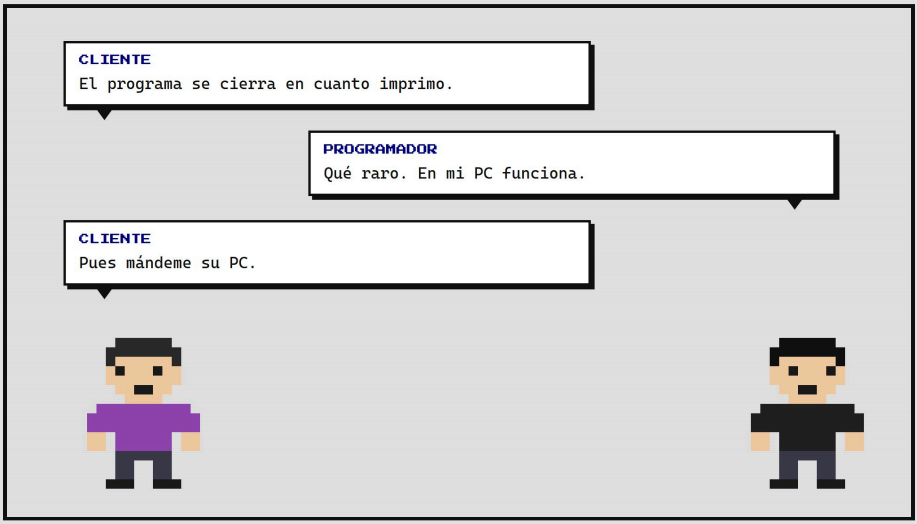
— Anónimo, *Dicho que circuló en las recopilaciones de humor informático (p. ej., listas de Multics y Usenet); identidad de Lubarsky no verificada (atribución incierta)*

Cuatro palabras y queda resumida la carrera de todo programador. Corriges el último bug, brindas, publicas. Suena el teléfono. Había un último-último bug.

El significado es una sabia humildad: ningún programa no trivial está libre de errores. Se pueden hacer raros, se pueden hacer inofensivos, pero creer que los has cazado todos es el bug más peligroso de todos.

Llamé a un fumigador para el servidor. Roció algo sobre la caja, dijo «seis meses de garantía» y se fue. Al día siguiente el programa de gestión imprimía las facturas en cirílico. Volví a llamar. Me dijo que esa especie no estaba cubierta. Entomología cibernética: la ciencia exacta del siempre.

LEY DEL «EN MI PC FUNCIONA»



Cualquier error que señale el cliente es imposible de reproducir en el ordenador del programador, donde todo funciona perfectamente.

— Anónimo, *folclore informático*

Es la frase más pronunciada en la historia de la informática, más que «¿has probado a reiniciar?». El programador la dice con una sinceridad conmovedora: en su ordenador el programa funciona de verdad. El problema es que el cliente no vive dentro de su ordenador.

El ordenador del desarrollador es un ecosistema protegido, un invernadero tropical con todas las librerías correctas, la configuración correcta, el idioma correcto. El ordenador del cliente es una selva con un Windows de 2016, tres antivirus en guerra entre sí y la fecha puesta en la hora de Honolulu.

Hubo una época en la que valoré en serio enviar mi PC a los clientes junto con el software. Habría resuelto el noventa por ciento de los problemas. Luego comprendí que el cliente llamaría igualmente, para decirme que mi PC en su mesa no funcionaba.

LEY DEL COMPANERO TAUMATURGO



El bug que te atormenta desde hace horas desaparece en el instante en que llamas a un compañero para que lo vea, y reaparece en cuanto él vuelve a su mesa.

— Anónimo, *folclore informático*

En cada oficina hay un compañero con poderes. No programa mejor que tú, no ha estudiado más. Simplemente, cuando se acerca a tu monitor, los bugs se cuadran y fingen no existir.

«A ver, enséñamelo», dice. Tú repites exactamente los mismos pasos, con la voz temblorosa de quien denuncia un fantasma. El programa funciona. Él te da una palmadita en el hombro con aire de médico de cabecera. Se va. El bug reaparece y te sonríe.

Los programadores tienen incluso un nombre para estos errores tímidos: los llaman bugs que cambian de comportamiento cuando los observas. Yo lo he resuelto de raíz. Imprimí una foto del compañero a tamaño natural y la colgué detrás de la silla. Funciona en el sesenta por ciento de los casos.

LEY DE LA LINEA INOCENTE

SOSPECHOSOS, POR ORDEN

1. el código del compañero
2. la librería externa
3. el sistema operativo
4. los rayos cósmicos
5. mi línea (imposible)

Seis horas de investigación. El culpable era el presidente de la comisión.

El bug está siempre en la última línea que revisas, y es la que estabas absolutamente seguro de que era correcta.

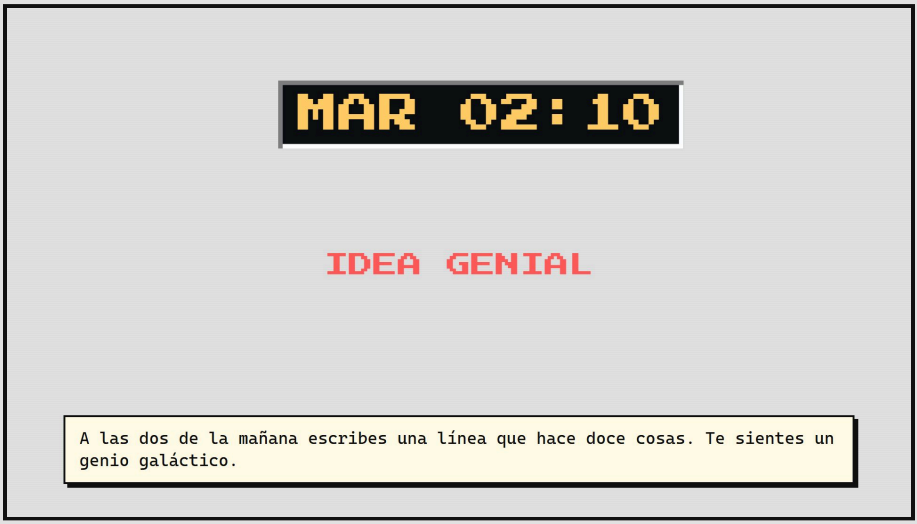
— Anónimo, *folclore informático*

La caza del bug sigue un ritual preciso. Primero se sospecha del código de los compañeros, luego de la librería externa, luego del sistema operativo, luego del compilador y, por último, de los rayos cósmicos. Solo al final, a regañadientes, se mira la línea propia. La perfecta. La escrita con orgullo.

Y ahí está el error. Un punto y coma fuera de sitio, un igual en lugar de un doble igual. Seis horas de investigación por un carácter. Una comisión parlamentaria entera para descubrir que el culpable era el presidente de la comisión.

Hay también una explicación lógica, que estropea la poesía: dejas de buscar cuando lo encuentras, así que siempre está en el último sitio donde miras. Pero es inútil decírselo a un programador a las dos de la madrugada. Él sabe que el código se está burlando de él. Y, sinceramente, tiene razón.

LEY DE KERNIGHAN



Depurar es el doble de difícil que escribir el código. Por tanto, si escribes el código de la forma más ingeniosa posible, por definición no eres lo bastante listo para depurarlo.

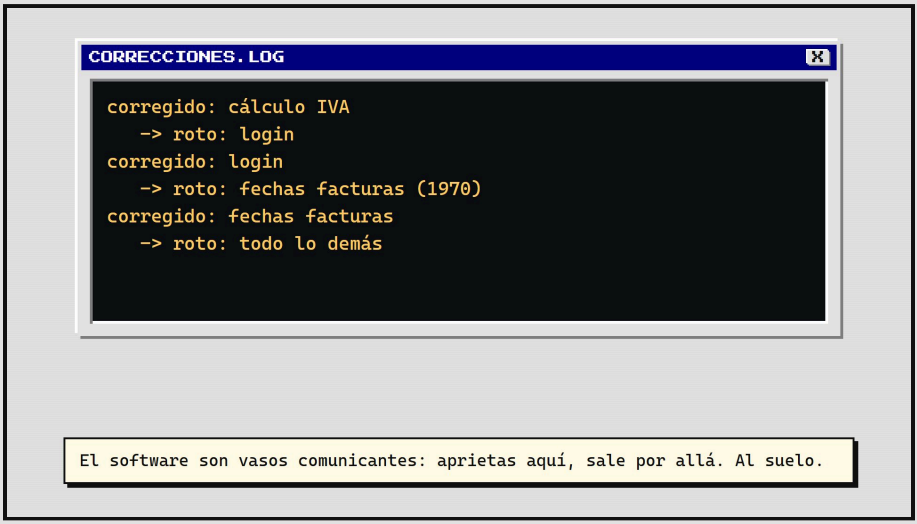
— Brian Kernighan, *The Elements of Programming Style (con P. J. Plauger), 2.ª ed., 1978*

Es un silogismo despiadado. A las dos de la madrugada, inspirado, escribes una línea que hace doce cosas a la vez. Te sientes un genio galáctico. A la mañana siguiente la línea no funciona y el genio de la noche no coge el teléfono.

El mensaje de verdad es: escribe sencillo. El código ingenioso es una deuda con tu yo del futuro, que estará más cansado, será más viejo y tendrá menos café.

Yo lo he resuelto con método: escribo el código al 50 % de mi inteligencia, así la depuración me sale al 100 %. Los compañeros dicen que se nota. Me lo tomo como un cumplido a la ingeniería, no a mí.

LEY DE LA CORRECCION PROLIFICA



Todo bug corregido con prisas genera al menos dos nuevos, a ser posible en una parte del programa que no tenía nada que ver.

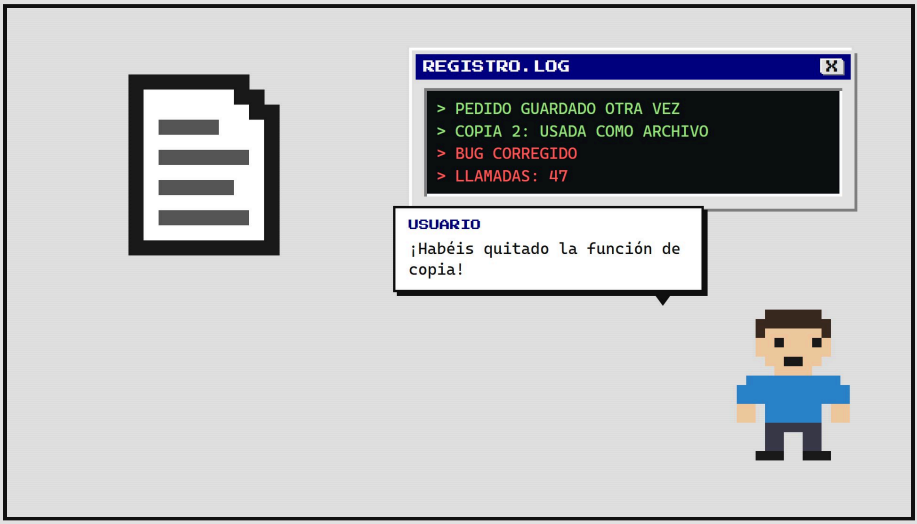
— Anónimo, *folclore informático*

Los bugs son como las cabezas de la hidra de las leyendas: cortas una y salen dos. Con una diferencia: las cabezas de la hidra al menos salían en el mismo sitio. Los bugs nuevos prefieren aparecer en la otra punta del programa, donde nadie los busca.

Corriges el cálculo del IVA y deja de funcionar el inicio de sesión. Arreglas el inicio de sesión y las facturas salen con fecha de 1970. El software es un sistema de vasos comunicantes: empujas por un lado y sale por el otro, normalmente al suelo.

El gurú de la calidad, en estos casos, pronuncia la palabra mágica: «tests automáticos». Tiene razón. Pero el bug urgente llega siempre cuando los tests automáticos están «casi listos». Yo afronto la corrección rápida con el mismo espíritu con que se cambia una bombilla en una casa antigua: con confianza, y con el extintor cerca.

LEY DEL BUG EVOLUCIONADO



Cualquier bug lo bastante avanzado es indistinguible de una funcionalidad.

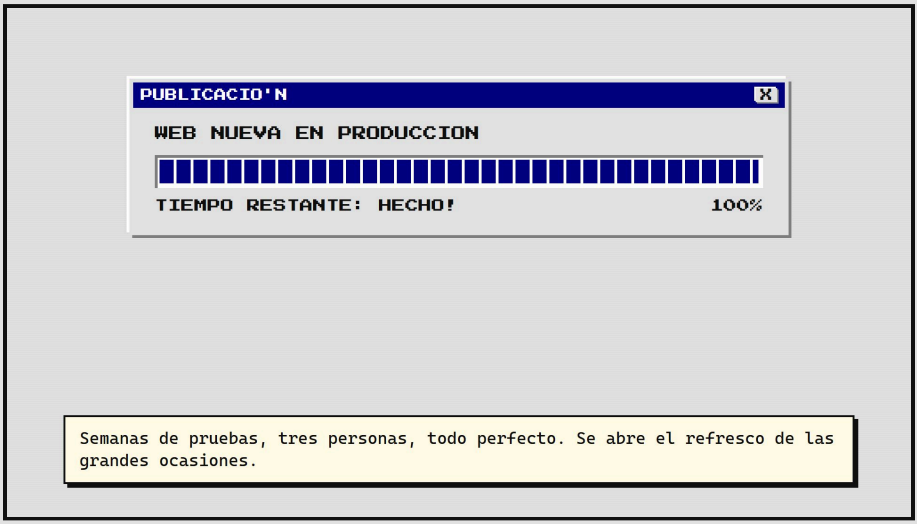
— Anónimo, *folclore informático*

De vez en cuando un error sobrevive tanto tiempo que echa raíces. Los usuarios lo descubren, se acostumbran, lo encuentran cómodo. A esas alturas, corregirlo se convierte en un acto de violencia.

Ejemplo real, contado con el debido respeto: un programa de gestión que, por un despiste, guardaba dos veces el borrador del pedido. Los empleados habían aprendido a usar la segunda copia como archivo. Cuando lo corregimos, llamaron en masa: «Habéis quitado la función de copia de seguridad».

De ahí nace la célebre frase que todo programador pronuncia al menos una vez en la vida, con la barbilla alta y la mirada en el horizonte: «No es un bug, es una funcionalidad». Es una fórmula mágica, como el abracadabra. No resuelve nada, pero convierte la vergüenza en marketing. Y el marketing, ya se sabe, no se corrige.

LEY DEL BUG PUNTUAL



El bug crítico se descubre diez minutos después de la publicación, nunca diez minutos antes.

— Anónimo, *folclore informático*

Durante semanas el programa se prueba, se vuelve a probar, lo revisan tres personas distintas. Todo bien. Luego se pulsa el botón «publicar», se abre el refresco de las grandes ocasiones y, en ese preciso instante, alguien dice: «Ejem, perdonad…».

No es mala suerte, es un fenómeno casi físico. Mientras el código está en pruebas, la mente está en modo defensivo y no ve nada. En cuanto el código es público e irreversible, la mente se relaja y de repente se vuelve genial. Lo ve todo. Demasiado tarde, pero lo ve todo.

He intentado engañar al destino: hago como que publico, espero diez minutos y luego publico de verdad. Funcionó dos veces. La tercera, encontré el bug en el décimo minuto de la falsa publicación y lo celebré como un gol. Luego llegó el segundo bug, el de verdad.

LEY DEL VIERNES POR LA TARDE



Un cambio que se pasa a producción un viernes por la tarde se romperá después de las seis, cuando todos los compañeros ya están en la autopista.

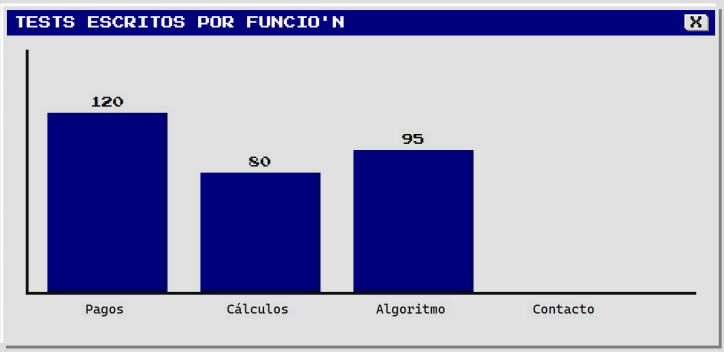
— Anónimo, *folclore informático*

En el calendario del informático, el viernes por la tarde es una zona sagrada y maldita, como las tierras fronterizas de las sagas fantásticas. Quien publica una actualización a esas horas desafía a los antiguos dioses de producción, y los antiguos dioses no perdonan.

La dinámica es siempre la misma. A las cuatro y media: «Es un cambio pequeñísimo, una sola línea». A las cinco: todo en verde, todo perfecto. A las seis y diez: la web muestra la portada en chino y el carrito regala la mercancía.

Mientras tanto, en la autopista, el compañero que conoce el sistema tiene el móvil en modo avión por motivos espirituales. En nuestra agencia rige un decreto no escrito: los viernes solo se hacen cosas reversibles. Quitar el polvo al teclado, leer la documentación, pensar. Sobre todo pensar, que nunca pasa a producción.

LEY DE LA PARTE NO PROBADA



Una fortaleza en los pagos. La puerta de atrás, atada con cordel.

La parte del programa que se rompe en producción es siempre la que nadie había pensado probar, porque era demasiado sencilla para fallar.

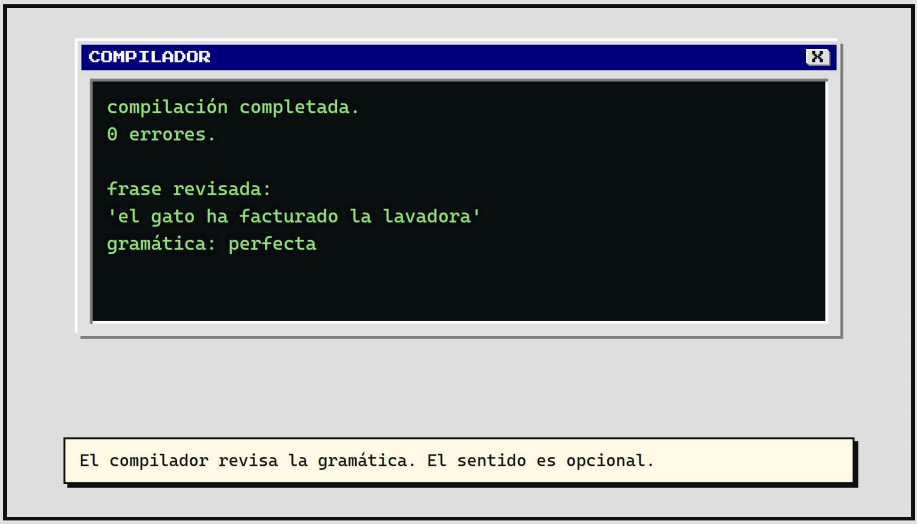
— Anónimo, *folclore informático*

Las pruebas se concentran siempre en las partes difíciles: cálculos complejos, algoritmos, pagos. Sobre ellas se construyen fortalezas. Luego la web se cae por culpa del botón «Contacta con nosotros», que nadie había pulsado nunca porque «total, es solo un formulario».

Es la versión informática del castillo medieval con murallas de tres metros de grosor y la puerta trasera atada con un cordel. El enemigo no escala las murallas. El enemigo, sencillamente, llama por detrás.

El santón de la calidad repite desde hace años el mismo sermón: «Lo que no está probado está roto, solo que aún no lo sabes». Es una frase que da miedo, porque es verdad. Yo ahora pruebo también las cosas tontas, una a una, con el celo de un notario. El cliente me pregunta por qué tardo tanto. Le contesto que estoy revisando la puerta trasera.

LEY DEL COMPILA, LUEGO FUNCIONA



Si el código compila sin errores, el programador cansado lo considerará listo para entregar.

— Anónimo, *folclore informático*

Compilar significa traducir el código a un programa ejecutable. El compilador comprueba que la gramática sea correcta, no que el programa tenga sentido. Es como un corrector de pruebas que aprueba con entusiasmo la frase «el gato ha facturado la lavadora».

Y sin embargo, a las siete de la tarde, ante el aviso «compilación completada», algo se dispara en el cerebro humano. Una voz interior dice: ya está. El código sale hacia el cliente como un barco que deja el puerto con las velas puestas del revés.

En muchas oficinas circula la versión abreviada del lema: «¿Compila? Se envía». Lo dicen riéndose. Luego lo ponen en práctica. Yo lo hice una sola vez, en un pasado remoto. La lavadora, en sentido figurado, sigue centrifugando. Y yo he aprendido a probar el programa antes, al menos una vez.

LEY DEL DATO DE PRUEBA

BASE DE DATOS (00:47)

```
insertar cliente:
   nombre: Asdf Qwerty
   dirección: Calle del Bug 404
   notas: prueba prueba no mirar

(total, mañana lo borro)
```

Medianoche. El programador cansado habla solo, dentro de la base de datos.

El dato de prueba más embarazoso introducido durante el desarrollo aparecerá, tarde o temprano, delante del cliente más importante.

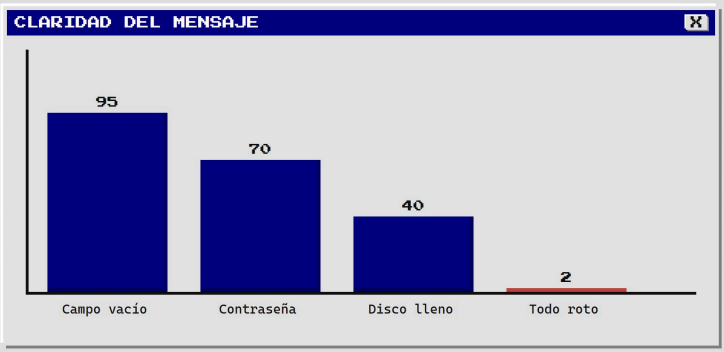
— Anónimo, *folclore informático*

Durante el desarrollo nadie introduce datos realistas. Se escribe «asdf», «prueba prueba», «Pepito Pérez», «producto que no existe». Son los marcadores del programador cansado, su forma de hablar solo a medianoche.

Luego llega la presentación oficial. Sala de reuniones, proyector, director general en primera fila. Se abre el listado de clientes y arriba del todo, en negrita, está el señor Asdf Qwerty, con la dirección «Calle de los Bugs 404». Silencio. Alguien tose.

La regla que aprendí a mi costa es sencilla: escribe cada dato de prueba como si el cliente lo fuera a leer. Porque lo leerá. Hoy en mis entornos de pruebas solo hay nombres sobrios y respetables, del tipo «Cliente Ejemplo S.L.». Un poco más triste, pero mucho menos sudor.

LEY DEL MENSAJE ORACULAR



Cuanto más grave el error, menos habla. Como un oráculo sin saldo.

Cuanto más grave es un error, más vago, genérico e incomprensible es el mensaje que lo anuncia.

— Anónimo, *folclore informático*

Los errores pequeños hablan claro: «El campo Correo no es válido». Los errores catastróficos, en cambio, se expresan como los oráculos antiguos: «Se ha producido un error inesperado». Código 0x8000FFFF. Fin de la comunicación.

¿Inesperado por quién, además? Es una frase escrita por un programador que, años antes, había previsto perfectamente que algo saldría mal, pero no tenía ganas de decir qué. Es una nota pegada en la nevera: «Ha pasado una cosa. Adiós».

Los clientes me los leen por teléfono con la voz de quien recita una profecía. Yo escucho en silencio, asiento y luego pronuncio la fórmula sagrada que todo sistemista conoce: «¿Ha probado a reiniciar?». No siempre funciona. Pero es la única respuesta a la altura de la pregunta.

LEY DEL CALENDARIO TRAIDOR

FECHAS QUE EL PROGRAMA ODIA

- 29 de febrero
- la noche del cambio de hora
- 1 de enero
- 30 de febrero (inventado por él)

Trescientos sesenta y dos días perfectos. Los otros tres, novela gótica.

Cualquier programa que maneje fechas funcionará todo el año, salvo el 29 de febrero, la noche del cambio de hora y el primer día de enero.

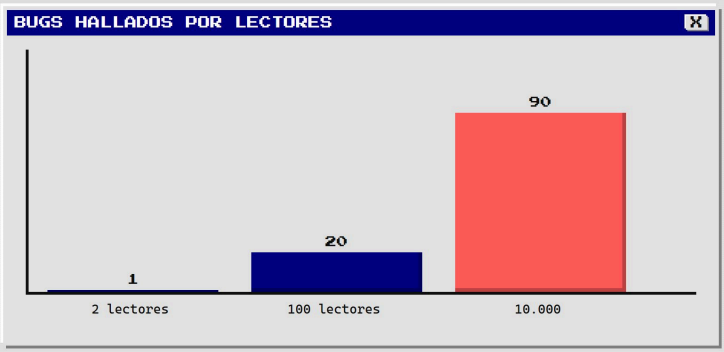
— Anónimo, *folclore informático*

El tiempo parece una cosa sencilla. Sesenta segundos, sesenta minutos, veinticuatro horas. Luego un programador intenta manejarlo en el código y descubre el horror: años bisiestos, husos horarios, horario de verano, meses de veintiocho días, países que cambian de opinión.

El resultado es un software que funciona de maravilla trescientos sesenta y dos días al año. Los otros tres hace cosas de novela gótica: reservas en el pasado, nóminas con una hora fantasma, vencimientos fijados el 30 de febrero.

Una noche de finales de octubre, un programa de gestión mío registró el mismo fichaje dos veces, a las dos y media, porque esa hora había existido dos veces de verdad. El empleado figuraba presente en dos sitios a la vez. El director quería ascenderlo. Yo corregí el bug, con el pesar de quien ha destruido un milagro.

LEY DE LINUS



Código abierto y muchos lectores: tarde o temprano el fallo sale.

Con suficientes ojos mirando, todos los bugs salen a la superficie.

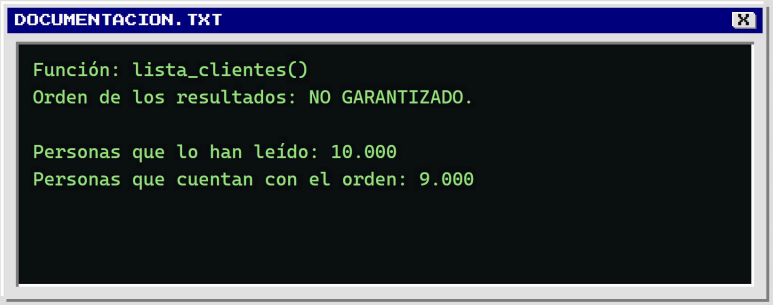
— Eric S. Raymond, *The Cathedral and the Bazaar, 1997 (ensayo), 1999 (libro); bautizada en honor de Linus Torvalds*

La idea es preciosa: si el código es abierto y miles de personas lo leen, alguien encontrará el defecto. Es la razón por la que el software libre puede competir con las grandes empresas, y a menudo ganar.

El detalle incómodo es la palabra «suficientes». Los ojos, además, tienen que mirar de verdad. Hay librerías que usa medio planeta y que, en la práctica, leen dos voluntarios cansados y un gato.

Intenté aplicarla en la oficina: proyecté el código en la pared durante la pausa del café. Veinte ojos. Ningún bug encontrado, pero tres compañeros señalaron que la fuente era fea y uno preguntó si el proyector era nuevo. El bug sigue ahí, muy bien iluminado.

LEY DE HYRUM



El día que cambias el orden, caen nueve mil programas. Todos avisados, ninguno a salvo.

Con suficientes usuarios, cualquier comportamiento observable de tu sistema lo usará alguien, independientemente de lo que hayas prometido en la documentación.

— Hyrum Wright, *Observación nacida en la ingeniería de software de Google, años 2010 (hyrumslaw.com)*

Escribes en la documentación: «El orden de los resultados no está garantizado». Diez mil personas lo leen. Nueve mil construyen su programa dando por hecho que el orden es siempre ese. El día que lo cambias, se cae medio Internet.

La ley nos dice que el contrato real de un software no es el escrito, sino todo lo que hace, defectos incluidos. Corregir un bug puede romperle el negocio a quien lo había construido encima de ese bug.

Una vez arreglé una errata en un mensaje de error. Al día siguiente me llamó un cliente furioso: su script buscaba precisamente esa palabra mal escrita para saber que algo había ido mal. Restauré la errata. Ahora es patrimonio histórico.

EL CONSEJO DEL SYSADMIN

La caza del bug no es un arte adivinatorio, es un método. El mío, probado en bastantes noches, es este.

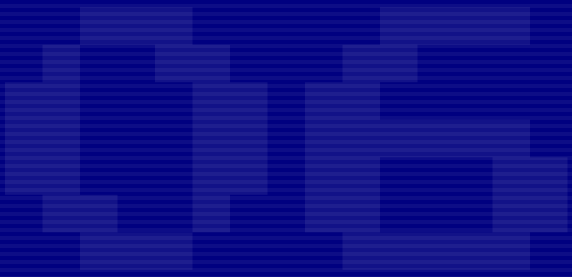
- **Primero reproducirlo.** Escribid los pasos exactos que hacen aparecer el error. Un bug que no sabéis hacer reaparecer tampoco sabréis decir si lo habéis corregido.
- **Leer los logs.** A menudo el programa ya ha escrito en algún sitio qué ha salido mal. Parece obvio: es el paso que más se salta.
- **Dividir por la mitad.** Si ayer funcionaba y hoy no, buscad a medio camino qué ha cambiado. Con Git existe incluso un comando a propósito, git bisect, que encuentra el cambio culpable partiendo el campo por la mitad cada vez.
- **Explicarlo en voz alta.** A un compañero, o a un objeto de la mesa. Contando el problema línea a línea, la línea equivocada a menudo se delata sola.

Y cuando lo encontréis, escribid un test que lo vuelva a atrapar si regresa. Los bugs siempre vuelven a casa, como los gatos.

AVISO

Esta página se ha dejado intencionadamente en blanco.

También esto es una ley de Murphy: la página en blanco sale siempre donde no hace falta.



CAPITULO 6

CODIGO

El arte de escribir código (y de arrepentirse)

Si el capítulo anterior era la crónica de sucesos, este es el tratado de filosofía. Las leyes que siguen no hablan de errores sueltos, sino de la forma de escribir software: cómo se pone nombre a las cosas, cuánto se comenta, cuándo se optimiza, por qué el código astuto de hoy es el dolor de cabeza de mañana.

Muchas llevan la firma de grandes nombres de la programación, personas que construyeron lenguajes, sistemas y libros con los que han estudiado generaciones. Es reconfortante descubrir que también ellos, en algún momento, se encontraron delante de un programa escrito por ellos mismos unos meses antes sin entender nada. El genio no vacuna.

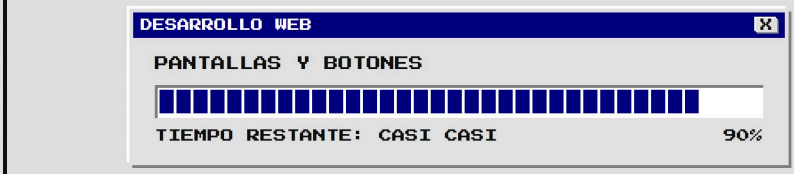
El hilo común es uno solo, y lo digo ya para que quien tenga prisa pueda saltar al final: la sencillez es difícil, pero es lo único que aguanta con el tiempo. El resto es comentario. Mentiroso, probablemente.

Para quien no programa: tranquilos. Cada ley está explicada con ejemplos cotidianos, y os servirá para entender por qué vuestro programa de gestión tarda dos años en mover un botón. No es pereza. Es arqueología.

CONTENIDO.DIR

01 Regla del noventa-noventa
02 Ley de las dos cosas difíciles
03 Ley de los dos problemas
04 Ley de Eagleson
05 Ley del comentario mentiroso
06 Ley de la documentación póstuma
07 Ley del código adoptado
08 Ley de Osborn
09 Axioma de Flon
10 Principio de Hoare
11 Optimización prematura
12 Reglas de Pike
13 Ley de Deméter
14 Ley de las abstracciones con fugas
15 Décima regla de Greenspun
16 Ley de Wadler
17 Leyes de Lehman

REGLA DEL NOVENTA-NOVENTA



El primer noventa por ciento vuela. El cliente dice: wow.

El primer 90 % del código requiere el 90 % del tiempo de desarrollo. El 10 % restante requiere el otro 90 %.

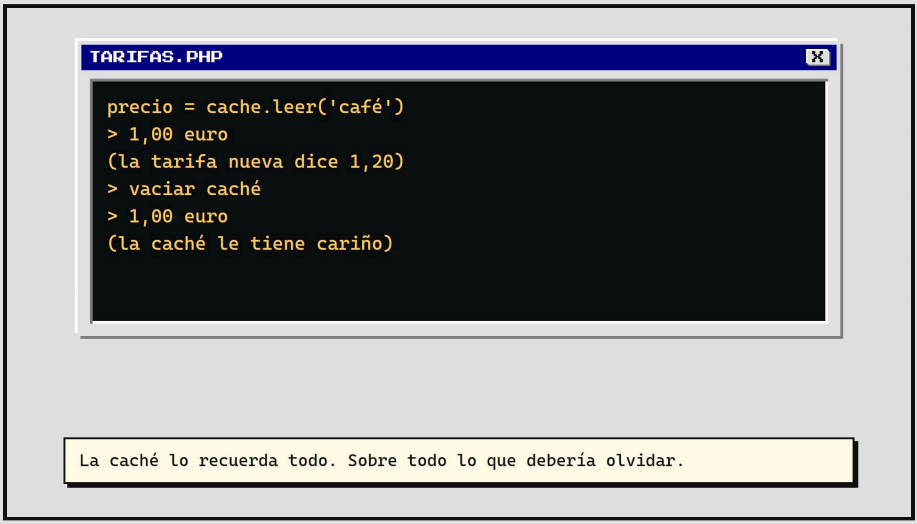
— Tom Cargill, *Bell Labs; dada a conocer por Jon Bentley en la columna Programming Pearls, Communications of the ACM, 1985*

Las matemáticas dicen 180 %. El programador dice: exacto. El proyecto está siempre «casi terminado» durante un periodo que dura tanto como el proyecto entero.

El primer 90 % es la parte bonita: pantallas, botones, el cliente que dice «hala». El último 10 % es cuando descubres que un apellido puede llevar apóstrofo, que alguien nació un 29 de febrero y que el IVA cambia justo la semana de la entrega.

En mi oficina hay una web al 98 % desde 2017. Cada año celebramos el cumpleaños del «ya casi está». Tenemos tarta. Las velas están al 98 %: medio encendidas, pero con gran entusiasmo.

LEY DE LAS DOS COSAS DIFICILES



En informática solo hay dos problemas realmente difíciles: invalidar la caché y ponerles nombre a las cosas.

— Phil Karlton, *máxima atribuida a Karlton en los tiempos de Netscape, recogida por Tim Bray y por Martin Fowler («TwoHardThings»)*

Quien no programa piensa que las partes difíciles son las matemáticas, la seguridad, la inteligencia artificial. Error. Las dos verdaderas montañas son estas: conseguir que el ordenador olvide una cosa vieja y decidir cómo llamar a una cosa nueva.

La caché es la memoria rápida que guarda los datos a mano para no tener que ir a buscarlos cada vez. Funciona de maravilla, hasta que te enseña con orgullo el precio de ayer. El nombre, en cambio, es una tortura filosófica: «datos», «datos2», «datosNuevos», «datosNuevosDeVerdad». He visto equipos enteros bloqueados una tarde por un nombre de variable, como un concilio medieval.

Existe también una variante folclórica que añade un tercer problema: los errores de uno de más o de menos al contar. Las cosas difíciles son dos, pero nadie ha conseguido nunca contarlas bien.

LEY DE LOS DOS PROBLEMAS

VALIDAR_EMAIL

```
regla = /^[a-z.]+@+[a-z]*\.es$/

probar('pepito@@.es')      -> VALIDA
probar('despacho@asesoria.com')
                           -> RECHAZADA
```

Una fórmula para reconocer emails. Acepta a pepito, rechaza al asesor.

Ante un problema, algunas personas piensan: «Lo resuelvo con expresiones regulares». Entonces tienen dos problemas.

— Jamie Zawinski, *mensaje en Usenet de 1997, reconstruido por Jeffrey Friedl en el blog regex.info (2006)*

Las expresiones regulares son fórmulas para buscar texto, potentísimas y escritas en un idioma que parece un gato paseándose por el teclado. Sirven, por ejemplo, para comprobar si una dirección de correo es válida. El resultado es una línea de símbolos que ningún ser humano sabe leer pasados diez minutos, incluido quien la escribió.

El primer problema era encontrar los correos. El segundo es entender por qué la fórmula acepta «fulanito@@.es» y rechaza al gestor. El tercero, porque siempre se suma uno, es que el único compañero que la entendía ha cambiado de trabajo.

Yo las uso con la cautela con la que se maneja un conjuro encontrado en el desván. Funcionan, pero cada vez que las toco me parece oír un trueno a lo lejos. Y normalmente, en efecto, se cae el servidor.

LEY DE EAGLESON

PROYECTO_VIEJO.PHP

```
function f($pepito2) {
  $x = $pepito2 * 3.14 + 7;  // ?
  // arreglar luego
  return f2($x, true, true, false);
}
```

Ni un comentario, variables misteriosas. ¿Quién ha sido?

El código que escribiste tú y que no miras desde hace seis meses o más bien podría haberlo escrito otra persona.

— Anónimo, *Dicho difundido entre programadores; origen e identidad de Eagleson no verificados (atribución incierta)*

Abres un archivo viejo. ¿Quién ha escrito esto? Variables llamadas «cosa2», ni un comentario, una función de trescientas líneas. Compruebas el autor. Eres tú. Tú mismo de hace seis meses, un perfecto desconocido con tus credenciales.

La ley enseña que el primer destinatario de la documentación no es el compañero: eres tú en el futuro, que lo habrás olvidado todo. Escribir código claro es un gesto de amabilidad hacia un desconocido que se te parece mucho.

Yo ya trato a mi yo del pasado como a un consultor externo. Le escribo correos de reclamación. Nunca contesta. Ayer encontré un comentario suyo: «Arreglar luego». ¿Luego cuándo? Estamos en el luego. El luego soy yo.

LEY DEL COMENTARIO MENTIROSO

REBAJAS.PHP

```
// calcula la rebaja del 10%
función rebaja(pedido):
   enviar_mail(director)
   rebaja = 30%
   // no tocar
```

El comentario cuenta una intención antigua. El código, un accidente moderno.

Cuando el comentario y el código no están de acuerdo, probablemente se equivocan los dos.

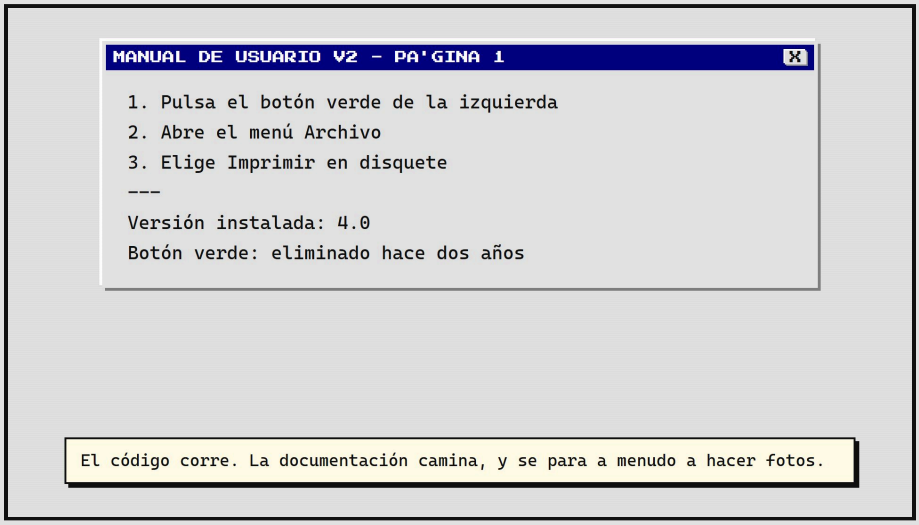
— Anónimo, *folclore informático*

Los comentarios son las frases que los programadores dejan en el código para explicar qué hace. En teoría son un gesto de cortesía hacia quien vendrá después. En la práctica son como las señales de tráfico de una carretera que entretanto han cambiado de sitio.

El código se cambia, el comentario se queda. Así encuentras escrito «calcula el descuento del diez por ciento» encima de una función que manda un correo al director. ¿Quién tiene razón? Nadie. El comentario cuenta una intención antigua; el código, un accidente moderno.

Existe también la escuela de pensamiento contraria, la de los programadores orgullosos que nunca comentan: si costó escribirlo, que cueste entenderlo. Yo estoy en medio. Comento siempre, pero con frases como «no tocar». No explica nada, pero salva muchas vidas.

LEY DE LA DOCUMENTACION POSTUMA



La documentación, cuando existe, describe con gran precisión la versión anterior del programa.

— Anónimo, *folclore informático*

La documentación técnica es un género literario en sí mismo, a medio camino entre la novela histórica y el testamento. Narra con todo lujo de detalles un mundo que ya no existe: botones que se han movido, menús rebautizados, funciones eliminadas entretanto en una reunión de cuatro minutos.

El motivo es sencillo. El código corre, la documentación camina. Cuando por fin alguien escribe el manual de la versión dos, los programadores ya se están peleando por la cuatro.

Tengo un cliente que guarda el manual de su programa de gestión en una vitrina, como una reliquia. De vez en cuando lo abre y lee las instrucciones en voz alta, en tono solemne. Ninguna coincide ya con la pantalla. Pero dice que le ayuda a recordar cómo era el mundo cuando los botones estaban a la izquierda. Y yo, francamente, lo entiendo.

LEY DEL CODIGO ADOPTADO

SOLUCION_DEL_FORO

```
// sacado de un foro, año desconocido
// no sé por qué, pero sirve
x = x + 0
esperar(0 segundos)
repetir 1 vez:
   funciona()
```

Líneas donadas por un desconocido. Se aceptan con gratitud y sin preguntas.

El código copiado de Internet funciona a la perfección hasta que intentas entender por qué funciona.

— Anónimo, *folclore informático*

Todo programador tiene un archivo secreto de líneas tomadas prestadas de los foros: soluciones escritas por desconocidos, en otra época, para otro problema. Las pegas, funcionan, y las miras con la gratitud de quien recibe un riñón de un donante anónimo.

El lío llega cuando, por escrúpulo, intentas entenderlas. Lees, razonas, quitas una línea que parece inútil. Se viene todo abajo. Vuelves a poner la línea. Ya no funciona ni con la línea. El código adoptado se ha ofendido y ha decidido no perdonarte.

Desde entonces trato estas líneas como los antiguos trataban los amuletos: respeto, distancia, ninguna pregunta. En mi código todavía hay trozos que llevan el comentario «no sé por qué, pero hace falta». Son las líneas más estables de todo el proyecto. El conocimiento, a veces, está sobrevalorado.

LEY DE OSBORN

CONFIG.PHP

```
const AÑO_ACTUAL = 2019;
const TIPO_IVA = 16;       // luego 18, luego 21
var temperatura = 0;       // nunca cambiada

echo AÑO_ACTUAL;
> 2019
```

La variable sigue igual desde el día uno. La constante ha cambiado varias veces.

Las variables no varían; las constantes no se mantienen constantes.

— Anónimo, *Dicho difundido en el humor informático; origen e identidad de Osborn no verificados (atribución incierta)*

En teoría, la variable cambia y la constante se está quieta. En la práctica, la variable sigue clavada en el valor equivocado desde que se creó, mientras que la constante «tipo de IVA» ha cambiado cuatro veces en diez años.

Detrás del chiste hay una lección: las cosas que das por inmutables, como impuestos, prefijos telefónicos o formatos de fecha, son las que cambiarán. Y las que deberían actualizarse a menudo se quedan bloqueadas por un error de inicialización.

En mi código hay una constante llamada AÑO_ACTUAL. Vale 2019. Nadie tiene valor para tocarla. Hemos decidido que, para el software, todavía es 2019. Se está bien. Los precios son más bajos.

AXIOMA DE FLON



No existe, no ha existido nunca y no existirá jamás un lenguaje de programación en el que sea difícil escribir programas feos.

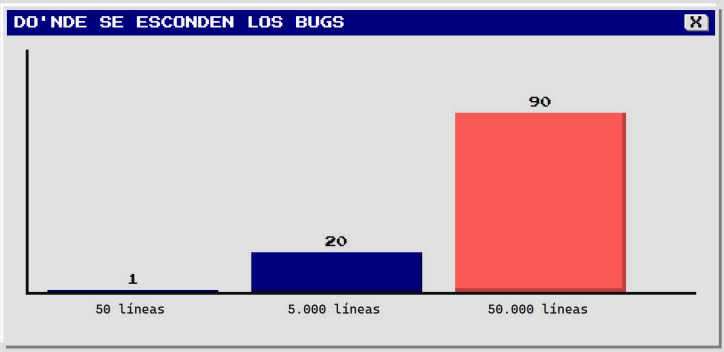
— Lawrence Flon, *On Research in Structured Programming*, *SIGPLAN Notices*, *octubre de 1975*

Cada lenguaje nuevo llega con la promesa mesiánica: «¡Conmigo el código será por fin limpio!». A los tres meses alguien ya ha escrito una función de novecientas líneas que se llama «gestionar_todo».

Flon no decía que todos los lenguajes sean iguales. Decía que la calidad está en quien escribe, no en la herramienta. Un buen lenguaje ayuda, pero no te impide construir un laberinto.

Hice un curso con un santón de los lenguajes modernos. Nos enseñó un código perfecto, elegante, seguro. Luego nos dejó solos diez minutos. A la vuelta habíamos reinventado las variables globales, el copiar y pegar y una carpeta llamada «definitivo_de_verdad». El lenguaje era inocente. Nosotros no.

PRINCIPIO DE HOARE



En cincuenta líneas el bug es una mosca en el mantel. En cincuenta mil tiene casa y huerto.

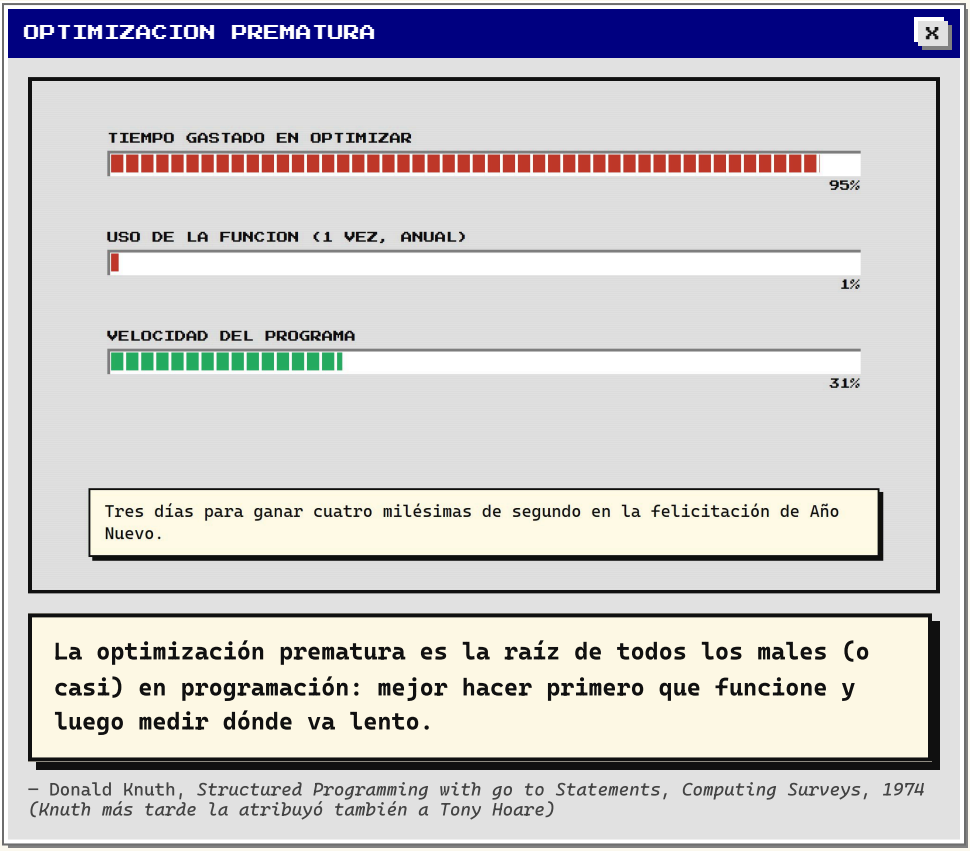
Hay dos maneras de diseñar software: hacerlo tan sencillo que obviamente no tenga defectos, o tan complicado que no tenga defectos obvios. La primera es mucho más difícil.

— C. A. R. Hoare, *The Emperor's Old Clothes, discurso del Premio Turing 1980, publicado en Communications of the ACM, 1981*

La complejidad es un escondite excelente. En un programa de cincuenta líneas un bug se ve como una mosca en un mantel blanco. En uno de cincuenta mil tiene casa, familia y huerto.

Hoare nos recuerda que la sencillez no es pereza: es lo más trabajoso de conseguir. Hay que quitar, repensar, tirar código al que uno le había cogido cariño. Añadir es fácil; renunciar requiere valor.

Le pregunté a un compañero si su módulo tenía defectos. Me respondió: «Ninguno que se vea». Era verdad. No se veía nada. Ni siquiera la lógica. Era una obra maestra de niebla lombarda en forma de código y, debajo, en la niebla, los bugs jugaban a la petanca.



— Donald Knuth, *Structured Programming with go to Statements*, *Computing Surveys*, 1974 *(Knuth más tarde la atribuyó también a Tony Hoare)*

Hay un tipo de programador que pasa tres días haciendo rapidísima una función que se llama una vez al año, el uno de enero, para imprimir la felicitación. Gana cuatro milisegundos. Lo celebra como un alunizaje.

El sentido de la frase es de sentido común: no sabes dónde está la lentitud hasta que mides. Optimizar a ciegas vuelve el código ilegible y deja intacto el verdadero cuello de botella, que suele ser una consulta metida dentro de un bucle.

Mi tío optimizó su bodega: estanterías por orden alfabético, índices, caché de las botellas favoritas. Luego se dio cuenta de que la puerta estaba cerrada y la llave estaba dentro de la bodega. El tiempo de acceso era infinito, pero muy bien organizado.

REGLAS DE PIKE

PERFIL.TXT

```
Hipótesis: lento en el cálculo
Medición real:
  cálculo        2%
  leer fichero  97%   (mismo fichero x 1000)
  otros          1%
```

Primero mide, luego optimiza. La intuición sobre rendimiento es un horóscopo.

No puedes saber de antemano dónde perderá tiempo un programa: primero mide. Los algoritmos sofisticados son lentos cuando hay pocos datos, y son los datos, más que el código, los que determinan la estructura adecuada.

— Rob Pike, *Notes on Programming in C, 1989*

Las reglas son cinco y suenan como los mandamientos de un monje del terminal: mide, no adivines; no uses algoritmos complicados si n es pequeño, y n suele ser pequeño; la sencillez gana a la elegancia; los datos mandan.

El mensaje es que la intuición de un programador sobre el rendimiento es tan fiable como un horóscopo. Uno cree saber dónde está la lentitud, tarda dos días en optimizar lo que no era y el problema de verdad era un bucle que leía el mismo archivo mil veces.

Implementé un árbol equilibrado autooptimizable para gestionar la agenda de la oficina. Contiene siete números. La vieja nota pegada en la nevera los encuentra más deprisa. La nota tiene, además, un elegante soporte magnético.

LEY DE DEMETER

PEDIDO.JS

```
nombre = cliente.pedido.direccion
         .ciudad.provincia.region.nombre
// seis eslabones: si cambia uno, se cae todo

mejor: nombre = cliente.provincia()
```

Una cadena de cotilleos: cada pieza sabe demasiado de las demás.

Un objeto debería hablar solo con sus vecinos directos y no con desconocidos: nada de cadenas de llamadas que atraviesan medio programa.

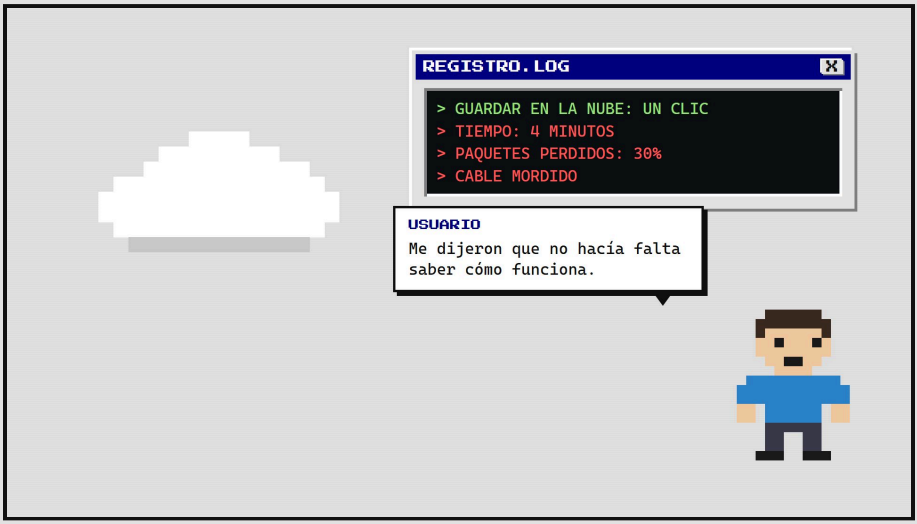
— Ian Holland, *Proyecto Demeter, Northeastern University, 1987 (grupo de Karl Lieberherr)*

También se conoce como principio del mínimo conocimiento. En la práctica: si quieres pagar al quiosquero, le das el dinero. No le pasas la cartera para que rebusque dentro, encuentre la tarjeta, llame al banco y le pregunte el saldo a tu madre.

En el código, una línea como cliente.pedido.direccion.ciudad.provincia.nombre es un cotilleo en cadena: cada eslabón sabe demasiado de los demás, y si uno cambia se caen todos. Reducir lo que cada uno sabe hace que las piezas se puedan sustituir.

Apliqué la ley en la oficina. Ya no le pido al compañero que le pida al técnico que le pida al proveedor. Se lo pido al compañero y punto. Él no sabe nada, pero el sistema está mucho más desacoplado. Ahora no funciona nada, pero de forma elegante.

LEY DE LAS ABSTRACCIONES CON FUGAS



Todas las abstracciones no triviales, tarde o temprano, dejan escapar los detalles que debían ocultar.

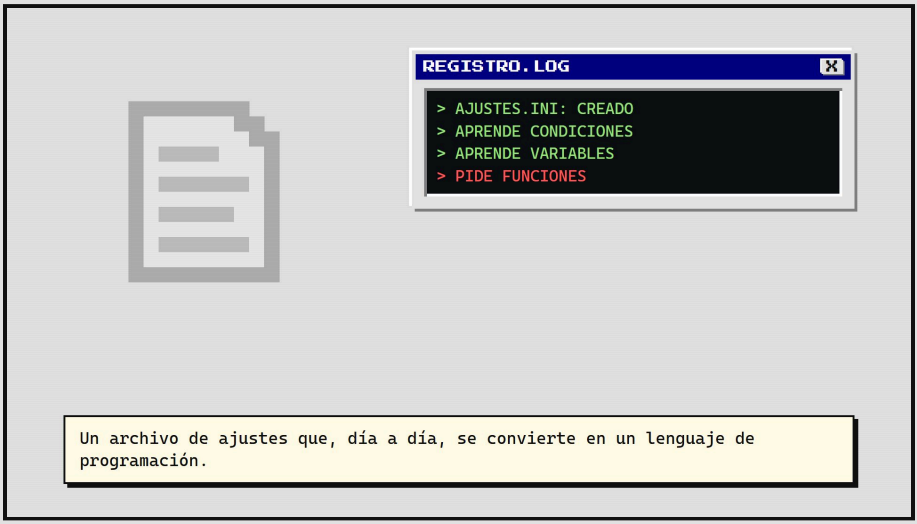
— Joel Spolsky, *The Law of Leaky Abstractions*, *Joel on Software*, 2002

Una abstracción es una fachada bonita: «No te preocupes por cómo funciona la red, guarda el archivo y listo». Funciona de maravilla, hasta el día en que el archivo tarda cuatro minutos y tienes que descubrir qué es una latencia, un paquete perdido y un cable mordisqueado por un roedor.

La cuestión es que las herramientas cómodas ahorran tiempo en el trabajo normal, pero no te eximen de entender qué hay debajo. Cuando tienen fugas, y siempre las tienen, hace falta alguien que sepa bajar al sótano.

La nube es la abstracción suprema: «Tus datos están en una nube». Luego la nube sufre una avería en un centro de datos con nombre, dirección y un aire acondicionado roto. La nube tiene código postal. Lo sé. He estado allí.

DECIMA REGLA DE GREENSPUN



Todo programa lo bastante complicado, escrito en C o Fortran, contiene una versión improvisada, lenta y llena de bugs de la mitad del lenguaje Lisp.

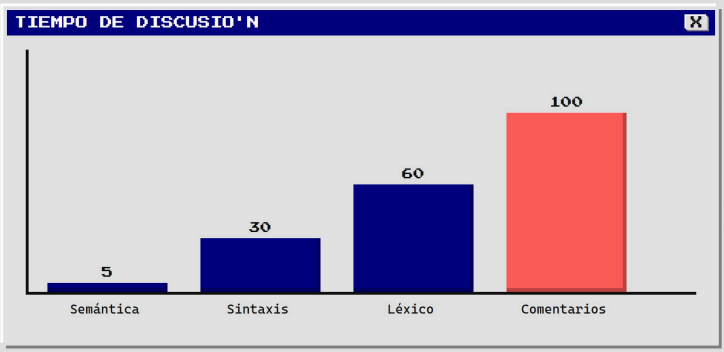
— Philip Greenspun, *Hacia 1993; las nueve primeras reglas nunca existieron*

Se llama «décima», pero es la única. Las otras nueve nunca se escribieron. Solo eso ya la convierte en la regla más honrada de la informática: promete nueve cosas y no entrega ninguna, como ciertos presupuestos.

El sentido es que los programas grandes acaban reinventando, mal, herramientas potentes que ya existían: un minilenguaje de configuración, un intérprete casero, un sistema de reglas escrito en un archivo de texto que en algún momento tiene hasta bucles.

En mi antiguo programa de gestión había un archivo de configuración. Luego aprendió las condiciones. Luego las variables. Un día pidió poder declarar funciones. Lo apagamos a tiempo. Ahora lo guardamos en un cajón, pero por la noche lo oigo compilar.

LEY DE WADLER



El núcleo del lenguaje: cinco minutos. El símbolo de comentario: tres semanas.

Al diseñar un lenguaje, el tiempo de discusión crece a medida que se baja de la semántica a la sintaxis, luego a la sintaxis léxica, hasta los comentarios, que ganan a todo.

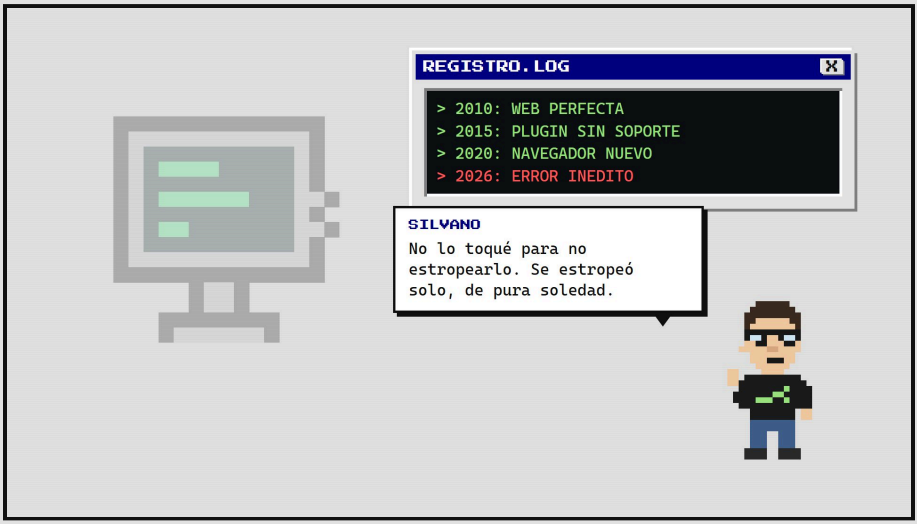
— Philip Wadler, *Mensajes en la lista de correo de Haskell, 1992 y 1996*

¿El significado profundo de un lenguaje? Cinco minutos, todos de acuerdo. ¿El símbolo para los comentarios? Tres semanas, dos cismas y un fork. Cuanto más pequeña es la cuestión, más experto se siente cada uno.

Es la versión para programadores de una regla humana universal: sobre las cosas difíciles se calla, sobre las fáciles se combate. Cualquiera que haya participado en una revisión de código ha visto cincuenta comentarios sobre los espacios y cero sobre la lógica.

En nuestro grupo discutimos largo y tendido si usar tabuladores o espacios. Creamos un comité. El comité produjo un documento. El documento tenía una sangría mixta. Ahora el comité tiene un subcomité. El proyecto avanza, en los comentarios.

LEYES DE LEHMAN



Un software que se usa en el mundo real debe cambiar continuamente o se vuelve cada vez menos útil; y cada cambio aumenta su complejidad, a menos que alguien trabaje expresamente para reducirla.

— Meir M. Lehman, *Leyes de la evolución del software, formuladas con László Bélády a partir de los años 70*

El software es como un jardín, no como una estatua. Si lo dejas estar, no se queda perfecto: las malas hierbas crecen solas. El mundo alrededor cambia, las leyes cambian, los navegadores cambian, y el programa que era impecable en 2010 hoy pide instalar un plugin extinto.

La parte más cruel es la segunda: cada cambio añade desorden. Un parche aquí, un caso especial allá, y el código se convierte en una comunidad de vecinos con doce porteros automáticos distintos.

Tengo una web en Joomla que no toco desde hace años para no estropearla. No la estropeé yo. Se estropeó sola, de pura soledad. Ahora voy a visitarla los domingos, le actualizo una extensión y ella me lo agradece con un error nuevo.

EL CONSEJO DEL SYSADMIN

Si escribís código, aunque sea poco, aunque sea solo para vosotros, usad un **sistema de control de versiones**. El más extendido es Git, y es gratuito. Funciona como una máquina del tiempo con etiquetas: cada vez que el trabajo está en buen estado, guardáis una foto con una frase que explica qué habéis cambiado.

Las ventajas son enormes. Volvéis a ayer con un comando. Veis exactamente qué línea ha cambiado, y cuándo. Probáis una idea loca en una rama aparte y la tiráis sin daños. Adiós a las carpetas duplicadas con la fecha en el nombre y la palabra «viejo» repetida tres veces.

Tres reglas de oro: guardad a menudo y a pasos pequeños; escribid mensajes que entendería hasta un desconocido, es decir, vosotros dentro de seis meses; no metáis nunca contraseñas ni claves de acceso en el repositorio, ni siquiera «un momentito». Ese momentito, en Internet, dura para siempre.

CAPITULO 7

REUNIONES

Proyectos, reuniones y clientes

Tarde o temprano todo informático descubre una verdad incómoda: los problemas más difíciles no están en el ordenador. Están en la sala de reuniones, en el presupuesto, en el correo con doce personas en copia. El código se puede corregir. Una estimación soltada en el pasillo, no.

Este capítulo reúne las leyes del trabajo en equipo. Se parte de los plazos, siempre equivocados por exceso de optimismo, y se pasa a los clientes, que saben lo que querían solo después de verlo. Luego están las reuniones, que se multiplican por recursión y discuten siempre lo menos importante, y las demos, que fallan en proporción al rango de quien mira.

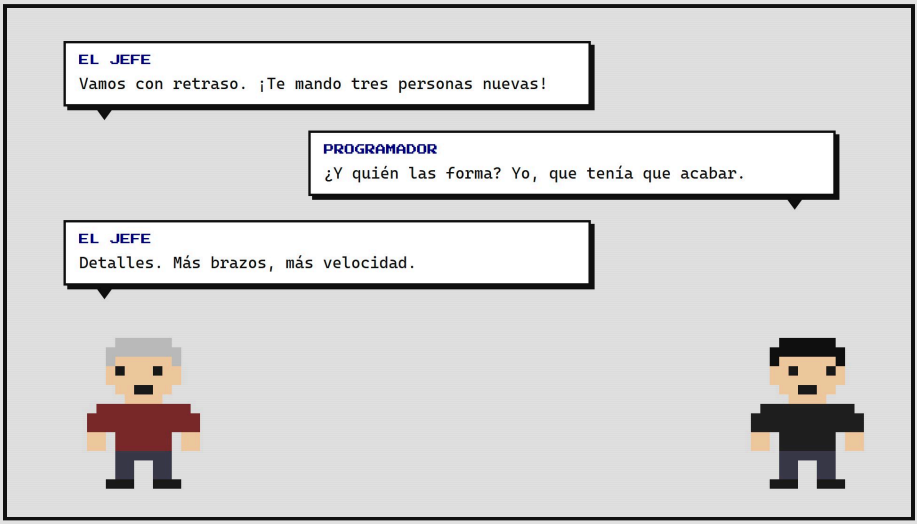
Al final llegan las leyes más amargas, las de las organizaciones: los números que se convierten en objetivos y dejan de decir la verdad, las soluciones provisionales que sostienen la empresa, los sistemas que acaban pareciéndose al organigrama, pasillos cerrados incluidos.

Quien trabaja en una oficina reconocerá a compañeros, jefes y también a sí mismo. Es normal. El truco es reconocerse antes de la reunión, no durante.

CONTENIDO.DIR

01 Ley de Brooks
02 Ley del multiplicador pi
03 Ley de la estimación jurada
04 Ley del cliente vidente
05 Ley del requisito fantasma
06 Ley de la reunión recursiva
07 Ley de la trivialidad de Parkinson
08 Ley de Sayre
09 Leyes de Wiio
10 Ley de la demo delante del cliente
11 Corolario del público importante
12 Ley de lo provisional eterno
13 Ley de Conway
14 Ley de Gall
15 Ley de Weinberg
16 Ley de Goodhart
17 Ley de Campbell
18 Principio de Dilbert

LEY DE BROOKS



Añadir personas a un proyecto de software que va con retraso lo retrasa todavía más.

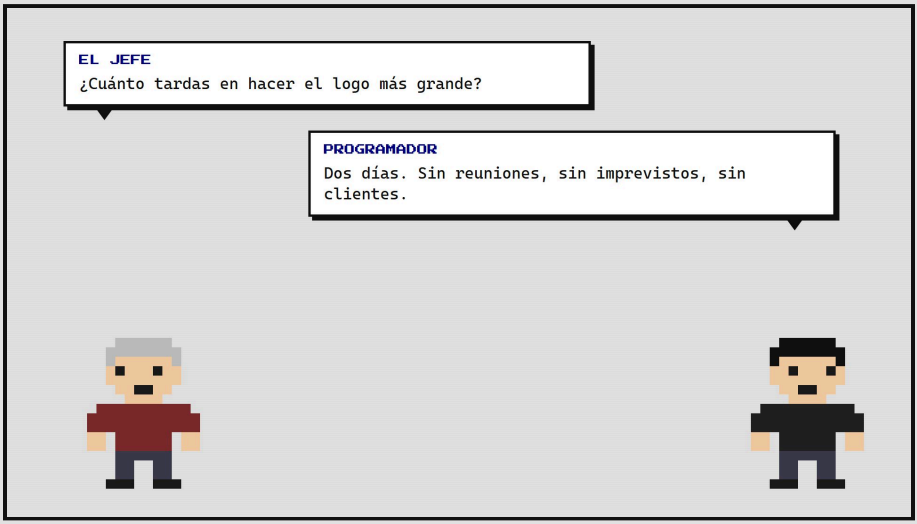
— Fred Brooks, *The Mythical Man-Month*, 1975

El directivo visionario mira el diagrama de Gantt, ve rojo y proclama: «¡Duplicamos el equipo!». Es el mismo razonamiento por el que nueve mujeres tendrían un hijo en un mes.

El problema es que a los recién llegados hay que formarlos, y quienes los forman son justo los que estaban terminando el trabajo. Además, cada persona de más multiplica las conversaciones: entre cuatro se habla; entre doce se hacen reuniones para decidir cuándo hacer las reuniones.

Una vez, para recuperar un retraso, me asignaron tres becarios. El primer día expliqué dónde estaban los baños. El segundo, la contraseña del wifi. El tercero, el proyecto iba otras dos semanas retrasado, pero el wifi funcionaba de maravilla.

LEY DEL MULTIPLICADOR PI



Para obtener el tiempo real de un proyecto de software, coge la estimación del programador y multiplícala por pi. Si el proyecto es importante, usa también los decimales.

— Anónimo, *folclore informático*

El programador que estima un trabajo es optimista por constitución. Imagina un mundo sin imprevistos, sin reuniones, sin el cliente que llama para preguntar si el logo puede ser un poquito más grande. En ese mundo ideal, «dos días» son dos días.

Luego llega la realidad, que tiene sus propias matemáticas. Dos días se convierten en seis coma veintiocho, y el cero coma veintiocho es el momento en que descubres que el servidor de pruebas tiene la contraseña de alguien que ya se jubiló.

¿Por qué precisamente pi? Nadie lo sabe. Pero es un número irracional e infinito, así que es el más adecuado para describir los plazos de entrega. Yo en los presupuestos he dejado de escribir los días. Escribo directamente el diámetro del proyecto y dejo que el cliente haga las cuentas.

LEY DE LA ESTIMACION JURADA



Cualquier estimación dicha de palabra, con mil cautelas y tres «quizás», se convertirá en una fecha límite escrita en el calendario del jefe.

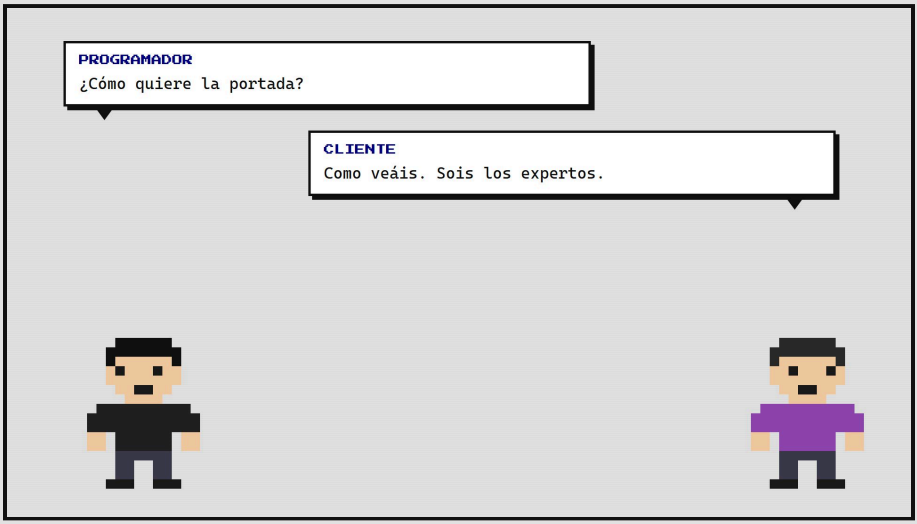
— Anónimo, *folclore informático*

El programador, interrogado en el pasillo, responde con prudencia de diplomático: «Así a ojo, si todo va bien, quizás, sin imprevistos, una semana. Pero no lo toméis al pie de la letra». El jefe asiente. Solo ha oído «una semana».

Al día siguiente esa frase aparece en un documento oficial, luego en un correo al cliente, luego en un gráfico de colores proyectado en una reunión. Las cautelas se han eliminado en la fase de compresión, como los detalles de una foto enviada por el móvil.

El directivo visionario, mientras tanto, la presenta con orgullo: «El equipo se ha comprometido». El equipo no sabía que se había comprometido. Desde entonces, cuando alguien me pregunta cuánto se tarda, respondo con otra pregunta: «¿Es una pregunta o es un contrato?». Normalmente el silencio que sigue vale como respuesta. Y como presupuesto.

LEY DEL CLIENTE VIDENTE



El cliente sabe exactamente lo que quiere en el preciso momento en que le entregas lo que había pedido.

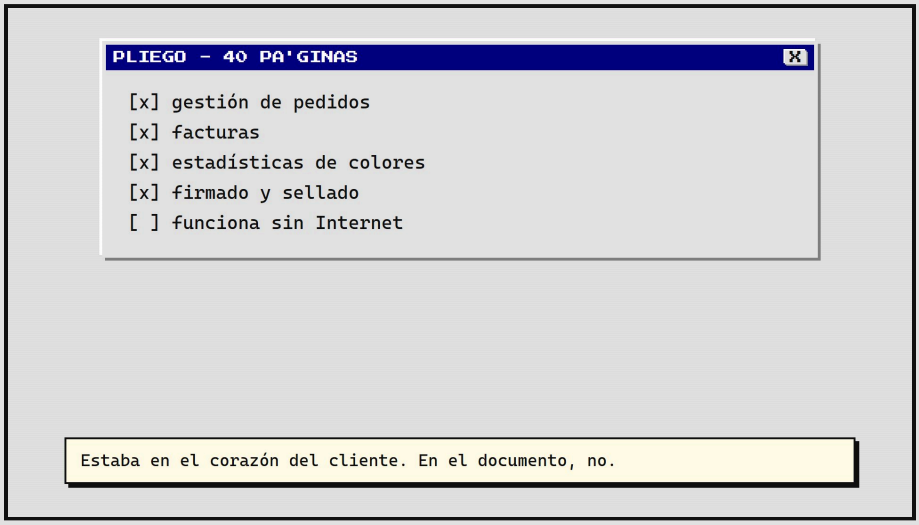
— Anónimo, *folclore informático*

Durante el análisis el cliente es vago, pensativo, zen. «Haced vosotros, que sois los expertos». Luego ve el trabajo terminado y tiene una revelación mística, como si se le abrieran los cielos: ahora lo sabe. Sabe que no era esto.

El fenómeno es bien conocido también en la literatura técnica: hay un célebre dibujo anónimo que muestra un columpio tal como lo quería el cliente, como lo entendió el comercial, como lo construyó el programador. Ninguna de las versiones es un columpio.

Yo he encontrado un método infalible. Preparo enseguida una versión equivocada, a propósito, en media jornada. El cliente la mira, se horroriza y por fin habla claro. Es el único caso en informática en que el bug es la funcionalidad más útil del proyecto. Lo facturo como «análisis de requisitos por contraste».

LEY DEL REQUISITO FANTASMA



El requisito más importante del proyecto se comunica de palabra, de pasada, el día de la entrega.

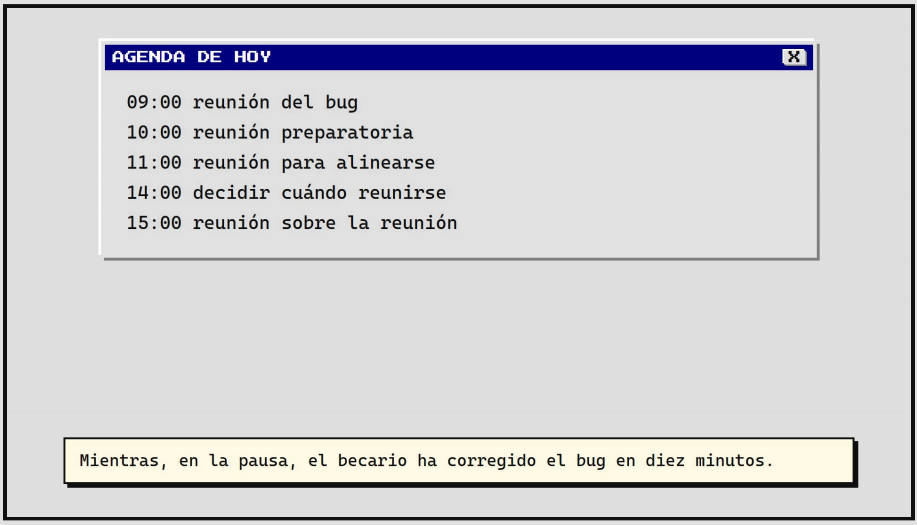
— Anónimo, *folclore informático*

El pliego de condiciones tiene cuarenta páginas, con tablas, gráficos y un índice que parece un tratado de paz. Todo se ha discutido, firmado y sellado. Luego, el día de la entrega, mientras se estrechan las manos, el cliente añade: «Ah, por supuesto tiene que funcionar también sin Internet».

No lo dice con maldad. Lo dice como quien dice «buenos días», con la serenidad de quien cree recordar un detalle obvio. Para él siempre ha sido así. Estaba escrito en su corazón, aunque no en el documento.

Desde entonces he introducido en la oficina la reunión de la pregunta final. Antes de firmar, pregunto: «¿Hay algo que os parezca tan obvio que no lo hayáis dicho?». El silencio que sigue siempre es largo. Y dentro de ese silencio, cada vez, vive un nuevo módulo por desarrollar.

LEY DE LA REUNION RECURSIVA



Todo problema técnico complejo genera una reunión, que genera una reunión preparatoria, que genera la necesidad de una reunión para decidir cuándo hacer la reunión.

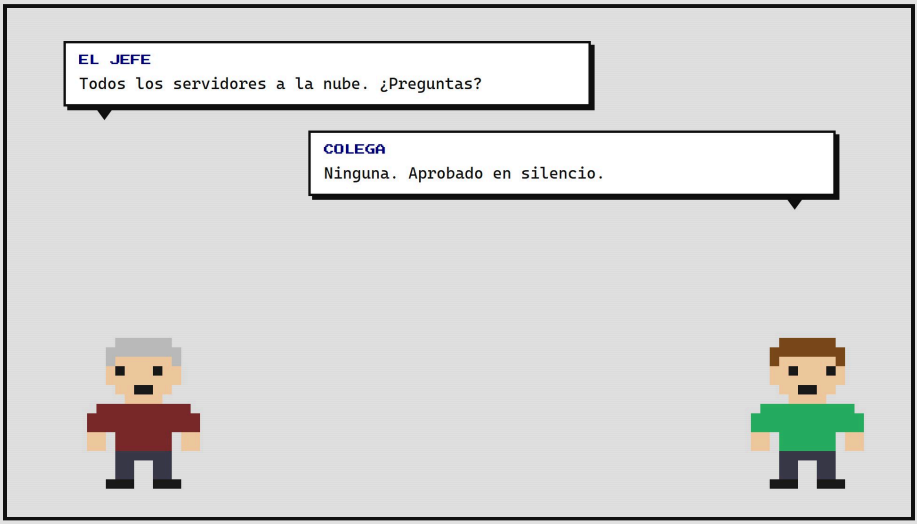
— Anónimo, *folclore informático*

En informática, la recursividad es una función que se llama a sí misma. En las empresas es una reunión que se convoca a sí misma, hasta el infinito, hasta que alguien se va de vacaciones y rompe el ciclo por agotamiento del calendario.

El modelo es siempre este. Se juntan ocho personas para hablar de un bug. Se descubre que hace falta información. Se fija un nuevo encuentro con otras cuatro personas. En ese encuentro surge la necesidad de «alinearse». Mientras tanto, el bug lo ha corregido el becario, en diez minutos, durante la pausa.

Siempre está el directivo que abre la sesión con solemnidad: «Señores, hoy decidimos». Y, efectivamente, se decide: la fecha de la próxima reunión. Yo propuse instalar un marcador en la sala, como los de las ferias del pueblo. Cada hora de reunión, una línea de código menos.

LEY DE LA TRIVIALIDAD DE PARKINSON



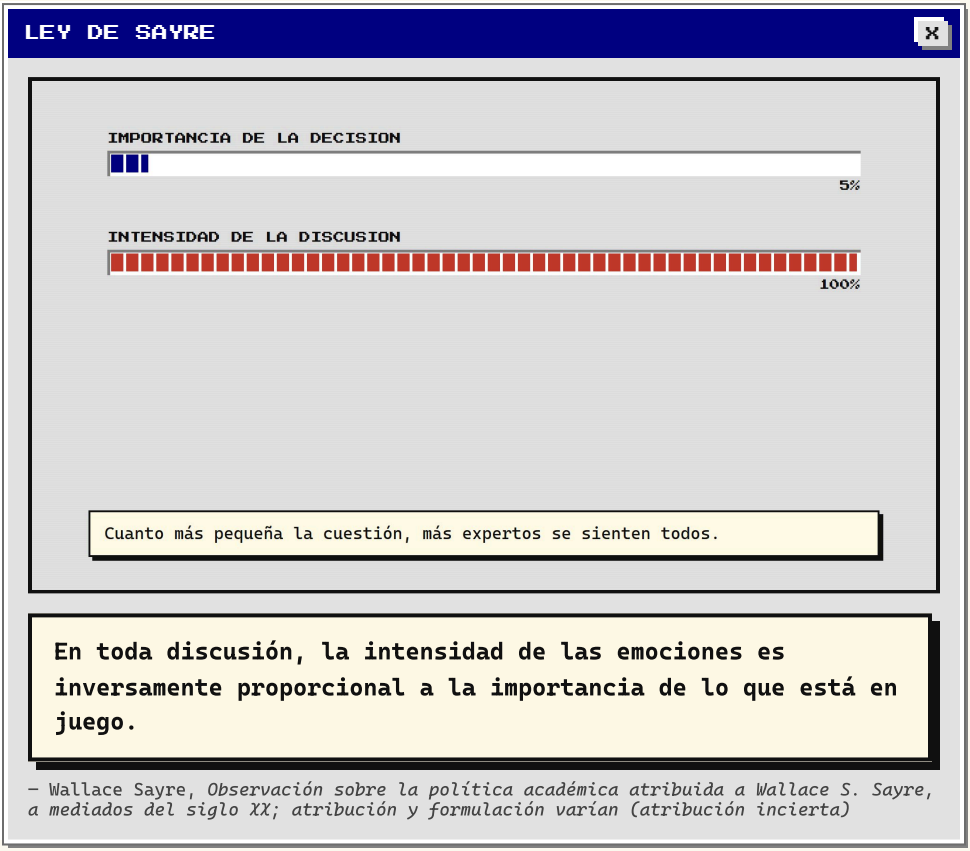
Una organización dedica una cantidad de tiempo inversamente proporcional a la importancia de las decisiones: se discute poco sobre el reactor y muchísimo sobre el aparcabicis.

— C. Northcote Parkinson, *Parkinson's Law, 1957; popularizada en informática como «bikeshedding» por Poul-Henning Kamp en la comunidad FreeBSD, 1999*

El consejo aprueba en dos minutos el proyecto de millones, que nadie entiende. Luego se pasa una hora con el cobertizo de las bicis, que todos entienden perfectamente y sobre cuyo color todos tienen opinión.

En informática se ha convertido en una consigna: cuando una discusión se enciende por un detalle irrelevante, alguien escribe «bikeshedding» y todos, por un momento, se avergüenzan. Luego vuelven a discutir el nombre de la variable.

En mi última reunión aprobamos en silencio el paso de todos los servidores a la nube. Luego discutimos cuarenta minutos sobre el icono del nuevo programa de gestión: ¿un engranaje o una nube? Al final elegimos un engranaje con forma de nube. Nadie sabe qué hace. Como la nube, por otra parte.



Nació para describir las peleas entre profesores universitarios, pero se aplica a la perfección a las reuniones técnicas. La migración de toda la base de datos se aprueba en cinco minutos. El color del botón «Cancelar» provoca dos horas de debate y un correo con media empresa en copia.

El motivo es sencillo: en las cuestiones enormes pocos se sienten competentes; en las minúsculas todo el mundo tiene opinión, y defenderla no cuesta nada.

La reunión más larga de mi carrera trató sobre el nombre de una carpeta compartida. «Documentos», «Documentación» o «Doc». Cuatro horas. Votamos. Ganó «Documentos_DEF». Al día siguiente apareció «Documentos_DEF_nueva». El conflicto se ha vuelto permanente, como ciertas guerras frías, pero con más permisos de escritura.

LEYES DE WIIO

UNA WEB SENCILLA, SEGU'N:

- el cliente: tres páginas
- el diseñador: minimalismo nórdico
- el programador: sin base de datos
- el comercial: cuesta poco

En la entrega nadie está contento. Todos seguros de haber pedido exactamente eso.

La comunicación normalmente fracasa, salvo por casualidad. Y si parece haber salido como estaba previsto, seguro que hay un malentendido.

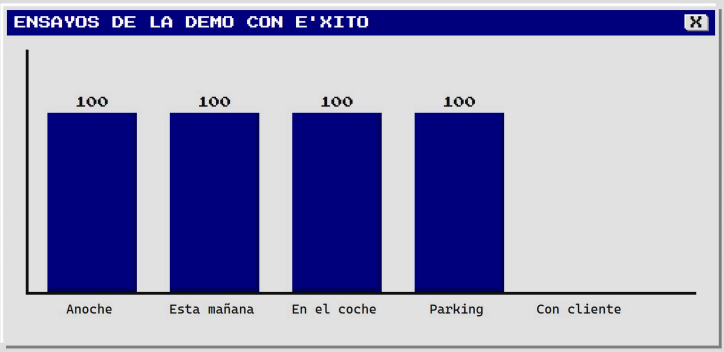
— Osmo Antero Wiio, *«Wiion lait – ja vähän muidenkin»*, 1978

Un profesor finlandés, estudioso de la comunicación, llegó a una conclusión que todo sistemista ya conoce: comunicar no funciona. Cuando funciona, es un accidente. Y si te parece que todos lo han entendido, es solo porque cada uno ha entendido una cosa distinta.

El cliente pide «una web sencilla». Él entiende tres páginas. El diseñador entiende minimalismo nórdico. El programador entiende sin base de datos. El comercial entiende «cuesta poco». En la entrega nadie está satisfecho, pero todos están seguros de haber pedido exactamente eso.

Para eso existen las especificaciones escritas, las actas, los correos de resumen. Sirven para documentar con precisión el malentendido. Yo ahora cierro cada reunión pidiendo a todos que repitan lo que hemos decidido. Cada vez salen cinco proyectos. Al menos lo descubrimos enseguida, y no en la prueba final.

LEY DE LA DEMO DELANTE DEL CLIENTE



Porcentaje de éxito. La última medición se tomó en presencia de alguien que decide.

Un programa que ha funcionado en cien pruebas falla en la ciento una: la que se hace delante del cliente.

— Anónimo, *folclore informático (conocido también como «efecto demo»)*

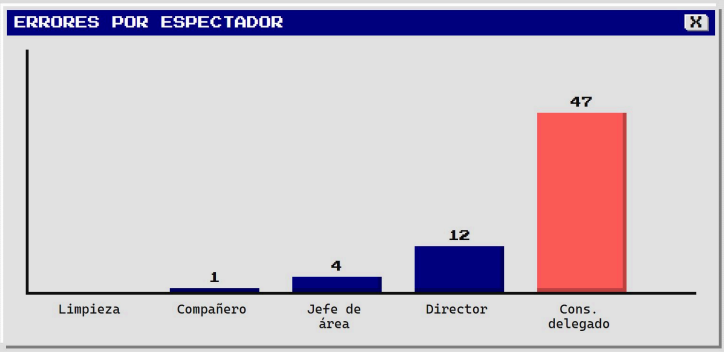
La demo se probó la noche anterior, por la mañana, en el coche y en el aparcamiento. Todo como la seda. Luego entras, el cliente se sienta, tú dices con aplomo «Ahora os enseño» y la pantalla se convierte en una paleta de errores nunca vistos.

Los informáticos lo llaman efecto demo. Es el único fenómeno conocido capaz de detectar la presencia de alguien con poder de decisión a diez metros de distancia, a través de las paredes.

Lo mejor es que, en cuanto el cliente se va, el programa vuelve a arrancar perfecto. Te mira como diciendo: «¿Qué ha pasado? Yo estaba aquí».

Consejo de quien lo ha vivido: preparad siempre un plan B. Un vídeo grabado de la demo funcionando, o capturas de los pasos clave. Si el programa se pone caprichoso, se proyecta la película.

COROLARIO DEL PUBLICO IMPORTANTE



Una escala jerárquica, precisa como un organigrama. Con la limpieza el servidor se calma.

Cuanto más alto es el cargo de quien asiste a la demostración, más probable es que la demostración falle.

— Anónimo, *folclore informático (recopilaciones de Usenet de leyes sobre ordenadores)*

Delante del compañero, la demo aguanta. Delante del jefe de departamento, titubea. Delante del director, tose. Delante del consejero delegado recién llegado en helicóptero, el ordenador se apaga y pide instalar cuarenta y siete actualizaciones.

Es una escala jerárquica del error, precisa como un organigrama. Algunos investigadores sostienen que las americanas caras emiten un campo electromagnético. Yo no tengo pruebas. Pero he notado que el servidor se calma cuando entra el de la limpieza.

El directivo visionario, mientras tanto, sonríe y dice: «Estupendo, hemos encontrado una oportunidad de mejora». Que en idioma corporativo significa: nos vemos en mi despacho.

Consejo de verdad: para las presentaciones importantes usad un entorno ya preparado y estable, y haced una prueba completa en la misma sala, con el mismo ordenador.

LEY DE LO PROVISIONAL ETERNO

CATALOGO.PHP

```
// TEMPORAL, solo para el finde
// arreglar más adelante
si precio vacío: precio = 9,99

último cambio: hace 12 años
líneas que dependen de esto: todas
```

El «más adelante» nunca llegó. Mientras, el parche sostiene todo el catálogo.

En el software no hay nada más definitivo que una solución temporal.

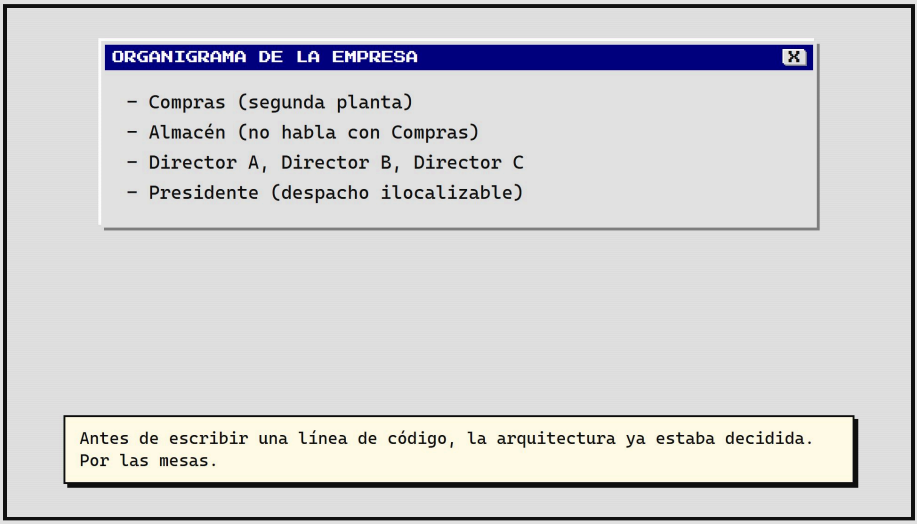
— Anónimo, *folclore informático*

Todo programa lleva dentro una línea acompañada de la nota «arreglar más adelante». Más adelante no llega nunca. La línea sobrevive a tres versiones, dos cambios de servidor y un cambio de dueño de la empresa.

Las soluciones provisionales tienen un temple extraordinario. Las soluciones definitivas, diseñadas con mimo, se reescriben cada dos años. El parche puesto a las once de la noche para que vuelvan a salir las facturas, en cambio, se convierte en patrimonio histórico, con su dignidad y su aura.

Llevo una web donde, hace doce años, alguien escribió un pequeño script «solo para el fin de semana». Hoy sostiene el catálogo entero. Nadie se atreve a tocarlo. De vez en cuando lo miro, así, con respeto, como se mira el muro de carga de una casa de pueblo: torcido, sí, pero si lo quitas se viene todo abajo.

LEY DE CONWAY



Una organización que diseña un sistema acabará produciendo un sistema que copia su propia estructura de comunicación.

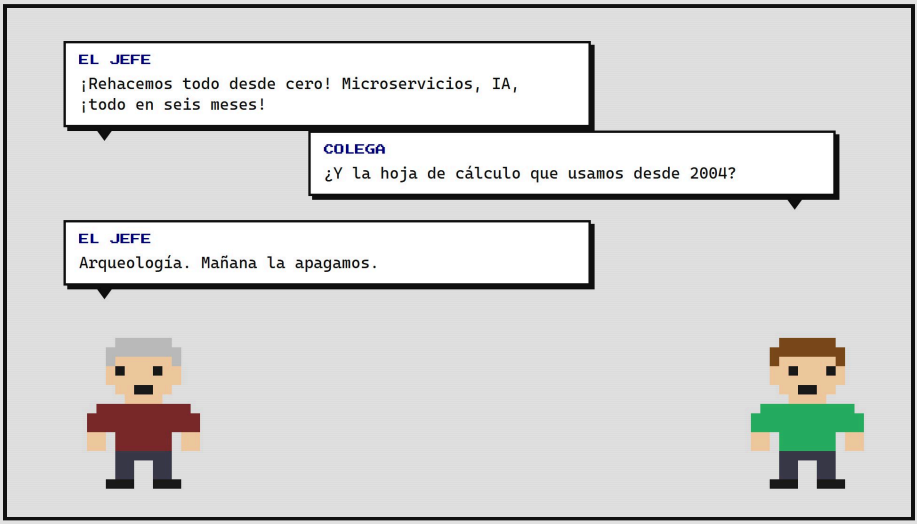
— Melvin Conway, *How Do Committees Invent?*, *Datamation*, 1968

Si cuatro grupos trabajan en un compilador, saldrá un compilador de cuatro pasadas. Si el departamento de compras no habla con el almacén, el programa de gestión tendrá dos fichas de productos que no se saludan.

El software es un espejo. Miras la arquitectura y ves el organigrama, con todos sus rencores. ¿Ese módulo que no se comunica con nadie? Es el compañero de la tercera planta que come solo.

He visto una web con tres menús distintos, uno por cada director. Un cuarto menú, oculto, llevaba a la página del presidente. Nadie sabía cómo se llegaba, exactamente igual que a su despacho. La arquitectura era perfecta: reflejaba la empresa hasta el último pasillo cerrado con llave.

LEY DE GALL



Un sistema complejo que funciona procede siempre de un sistema sencillo que funcionaba. Uno complejo diseñado desde cero no funciona y no hay manera de arreglarlo.

— John Gall, *Systemantics (General Systemantics)*, 1975

El gurú de la transformación digital sube al escenario: «¡Lo rehacemos todo desde cero, microservicios, inteligencia artificial, blockchain, en seis meses!». Aplausos. Dieciséis meses después se vuelve a la hoja de Excel compartida, que funcionaba desde 2004.

La ley es un elogio de la evolución. Los sistemas que aguantan han crecido trocito a trocito, como las estalactitas o las carpetas «Nueva carpeta (7)». Los que nacen enormes y perfectos sobre el papel se hunden, porque nadie los ha visto nunca funcionar en pequeño.

Mi programa de gestión favorito nació como un archivo de texto con tres líneas. Hoy gestiona una empresa entera. Todavía tiene dentro esas tres líneas, en alguna parte. Nadie las toca. Son la reliquia de carga.

LEY DE WEINBERG

EDIFICIO Y PROGRAMA, CARA A CARA

Edificio: cimientos, cálculos, inspección
Programa: librería de las 3 de la mañana
Mantenedor: un desconocido
Se apoyan en él: banco, hospital, café

La informática es joven: menos normas y menos pruebas que la construcción.

Si los constructores construyeran los edificios como los programadores escriben los programas, el primer pájaro carpintero que pasara destruiría la civilización.

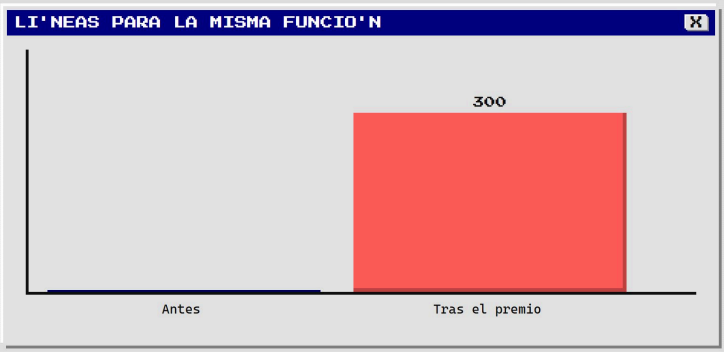
— Gerald Weinberg, *Atribución incierta: aparece en una recopilación de 1975 (CoEvolution Quarterly) y como «Segunda ley de Weinberg» en Murphy's Law de Arthur Bloch, 1977*

Un edificio de verdad tiene cimientos, cálculos estructurales, inspecciones. Un programa medio tiene una librería descargada a las tres de la madrugada, mantenida por un desconocido, de la que dependen un banco, un hospital y la máquina del café.

La ley no es solo una burla: recuerda que la informática es una disciplina joven, con menos normas y menos inspecciones que la construcción. Mucho de lo que usamos se sostiene por buena voluntad y suerte.

Una vez vi un pájaro carpintero en el tejado de nuestra nave, justo encima de la sala de servidores. Me puse entre él y la antena. Nos miramos largo rato. Al final se fue. Creo que entendió que no hacía falta: el servidor se rompería solo, el lunes.

LEY DE GOODHART



Se premian las líneas de código. La productividad se triplica. El software, misteriosamente, no.

Cuando una medida se convierte en un objetivo, deja de ser una buena medida.

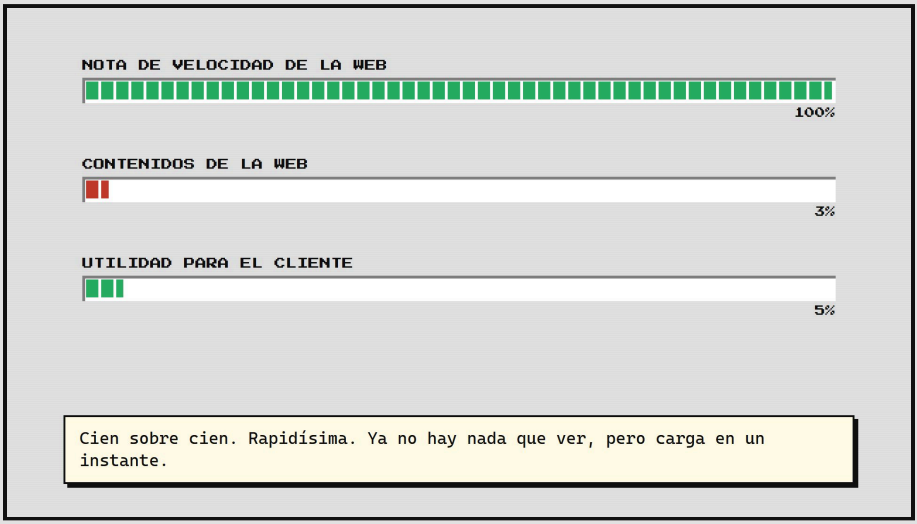
— Charles Goodhart, *Artículo sobre política monetaria, 1975; la formulación famosa es de Marilyn Strathern, 1997*

La empresa decide medir la productividad de los programadores contando las líneas de código. Al mes siguiente los programas son larguísimos. Una función que cabía en tres líneas ahora ocupa trescientas, con comentarios poéticos y variables con nombres de cuatro palabras. La productividad se ha triplicado. El software, misteriosamente, no.

Esta es la ley de Goodhart: en cuanto presionas sobre un número, la gente aprende a inflar el número en lugar de mejorar lo que debía representar. Vale para las incidencias cerradas, para las estrellitas de las reseñas, para las puntuaciones de velocidad de las webs.

En el departamento de soporte medían el tiempo de cierre de las llamadas. Un compañero las cerraba todas en ocho segundos. Contestaba, decía «pruebe a reiniciar» y colgaba. Premiado tres veces como empleado del mes. Los usuarios lo llamaban el Rayo.

LEY DE CAMPBELL



Cuanto más se usa un indicador numérico para tomar decisiones importantes, más se corrompe, y más distorsiona los procesos que debería vigilar.

— Donald T. Campbell, *«Assessing the Impact of Planned Social Change»*, 1976 *(formulaciones anteriores desde 1969)*

Prima hermana de la ley de Goodhart, la ley de Campbell añade un detalle sombrío: el número no solo deja de decir la verdad, sino que empieza a deformar el propio trabajo. Si el premio depende de la puntuación, el trabajo pasa a ser producir puntuación.

En el mundo de las webs es evidentísimo. Se decide que solo cuenta la nota de la herramienta de velocidad. Entonces se quitan las imágenes, los textos, las funciones, todo lo que pesa. La web obtiene cien sobre cien. Es rapidísima. Ya no hay nada que ver, pero lo carga en un santiamén.

En una empresa que conozco evaluaban al informático por el número de ordenadores que funcionaban a final de mes. Él, muy sabiamente, el último día del mes no tocaba ninguno. Funcionaban todos. Lentísimos, llenos de avisos, pero estadísticamente en perfecto estado de salud.

PRINCIPIO DE DILBERT



REGISTRO.LOG

```
> ACCESO AL SERVIDOR: DENEGADO
> DESPACHO NUEVO: CON VISTAS
> CARGO: MANAGER
```

Control de acceso primitivo: en lugar de permisos, un despacho con vistas.

Las empresas tienden a ascender a los empleados menos capaces a puestos de gestión, para limitar el daño que pueden hacer en el trabajo de verdad.

— Scott Adams, *Wall Street Journal*, 1995; libro «*The Dilbert Principle*», 1996

El principio de Peter decía que se sube hasta volverse incompetente. El principio de Dilbert es más cínico: alguien ya parte siendo incompetente y lo suben a propósito, donde no pueda tocar nada importante. Como poner al niño hiperactivo de capitán del equipo para que no toque el balón.

En informática el fenómeno tiene un nombre técnico no oficial: el directivo visionario. No sabe qué es una base de datos, pero ha entendido que el futuro está en los datos. Lo dice en las reuniones dando un puñetazo en la mesa. Luego le pide a alguien que le imprima el correo.

Reconozcámoslo, el sistema tiene su eficiencia: a la persona se la mantiene lejos del servidor, y el servidor lo agradece. Es una forma primitiva de control de accesos. Solo que, en lugar de permisos, se usa un despacho con vistas.

EL CONSEJO DEL SYSADMIN

Dos costumbres que me han salvado más proyectos que cualquier software.

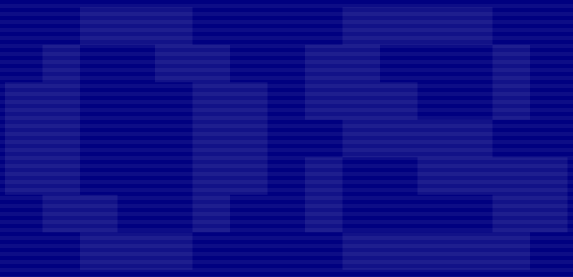
El correo de resumen. Después de cada reunión o llamada importante, escribid en menos de una hora un correo breve a todos los presentes: qué hemos decidido, quién hace qué, para cuándo, qué queda abierto. Cinco líneas. Si alguien ha entendido otra cosa, lo descubrís enseguida y no en la entrega. Y dentro de tres meses, cuando nadie recuerde nada, ese correo será la versión oficial de los hechos.

La estimación en horquilla. No digáis nunca un solo número. Decid «de tres a seis días, porque depende de esto y de aquello». El número único se convierte en una promesa; la horquilla obliga a todos a hablar de lo que la ensancha. Y poned por escrito también lo que queda fuera del trabajo. El requisito fantasma odia las listas con viñetas.

AVISO

Esta página se ha dejado intencionadamente en blanco.

También esto es una ley de Murphy: la página en blanco sale siempre donde no hace falta.



CAPITULO 8

FUTURO

Progreso, actualizaciones e inteligencia artificial

El último capítulo mira hacia delante, que en informática significa mirar fijamente la barra de progreso de una actualización. Se empieza justo por ahí: actualizaciones que llegan en el momento equivocado, que arreglan una cosa y rompen otra, ordenadores nuevos que ya son viejos al salir de la tienda.

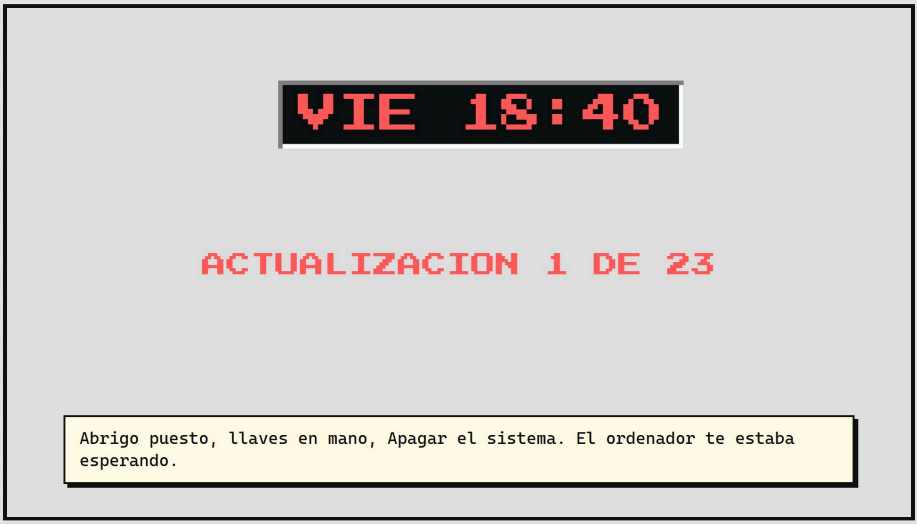
Luego se sube de escala. Están las leyes que describen la gran carrera de la tecnología: chips cada vez más potentes, fábricas cada vez más caras, software cada vez más pesado, programas que quieren hacerlo todo. Son leyes económicas y de ingeniería, a veces optimistas, a menudo divertidas. Explican por qué el ordenador es enormemente más rápido que hace veinte años y el procesador de textos se abre en el mismo tiempo.

El capítulo se cierra con seis corolarios dedicados a la inteligencia artificial. Estos los he escrito yo, después de muchas horas trabajando con estos asistentes, y los firmo con el orgullo de quien ha visto a una máquina pedir perdón varias veces por el mismo error idéntico. No estoy en contra de la IA: la uso todos los días. Precisamente por eso la quiero como se quiere a un compañero brillante que de vez en cuando se inventa los nombres de las funciones.

CONTENIDO.DIR

01 Ley de la actualización inoportuna
02 Ley de la actualización que rompe
03 Ley del ordenador nuevo
04 Ley de Wirth
05 Ley de Zawinski
06 Ley de Atwood
07 Ley de Rock
08 Ley de Grosch
09 Ley de Amdahl
10 Ley de Gustafson
11 Ley de Koomey
12 Ley de Bell de las clases de ordenadores
13 Paradoja de Jevons
14 Ley de Amara
15 Corolario de la función imaginaria
16 Corolario de las disculpas artificiales
17 Corolario de los tres segundos
18 Corolario del prompt solemne
19 Corolario del resto sin cambios
20 Corolario de la regla olvidada

LEY DE LA ACTUALIZACION INOPORTUNA



El ordenador decide instalar las actualizaciones siempre cuando tienes que apagarlo y salir corriendo, o encenderlo y presentar.

— Anónimo, *folclore informático*

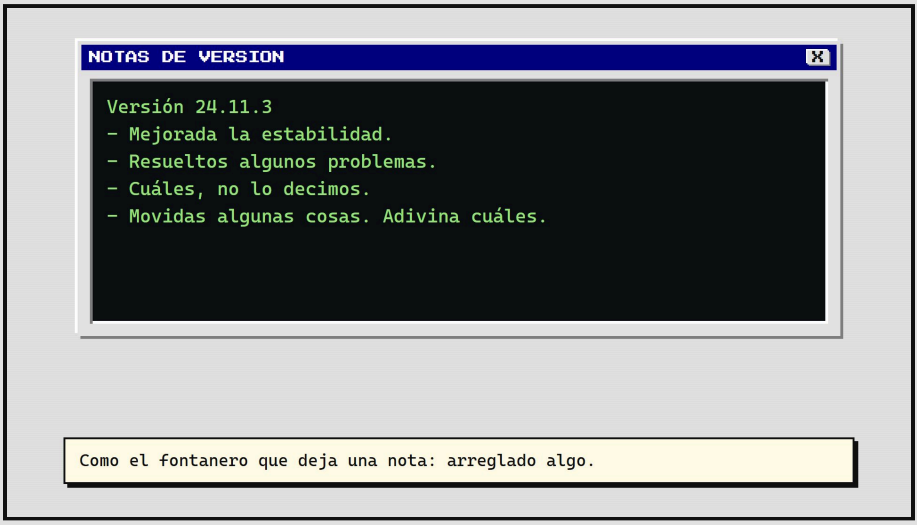
El ordenador ha pasado la noche encendido, inactivo, aburrido. Podría haberse actualizado en paz. Pero no: espera a que pulses «Apagar» con el abrigo puesto y las llaves en la mano. Entonces anuncia, con amabilidad: «Actualización 1 de 23. No apague el equipo».

Y tú te quedas ahí, como ante un paso a nivel. El tren de las actualizaciones pasa despacio, vagón tras vagón. A veces se para en el treinta por ciento para reflexionar sobre el sentido de la vida.

Un vecino mío se compró un portátil nuevo para hacer deprisa la declaración de la renta. Lo encendió en abril. En junio hizo la declaración a mano, en papel, con la pluma estilográfica del abuelo.

Consejo de técnico: configurad el horario de actividad del sistema, para que las actualizaciones se instalen cuando no usáis el ordenador. Y no lo apaguéis nunca a la fuerza durante una actualización.

LEY DE LA ACTUALIZACION QUE ROMPE



Toda actualización que resuelve un problema introduce al menos uno nuevo, normalmente en algo que antes funcionaba de maravilla.

— Anónimo, *folclore informático*

La actualización se presenta con una lista de novedades apasionantes: «Mejorada la estabilidad. Resueltos algunos problemas». Qué problemas, nunca se dice. Es un poco como un fontanero que deja una nota: «Arreglado algo».

Al día siguiente la impresora no imprime, el programa de contabilidad pide una licencia y el icono de la papelera se ha vuelto verde. Lo llaman progreso. Yo lo llamo mudanza: todo sigue ahí, pero nada está en su sitio.

El santón del «siempre se ha hecho así» aprovecha el momento y proclama: «Os lo dije, no hay que actualizar nunca nada». Luego le entra un virus por un programa parado en 2012.

Consejo de verdad: las actualizaciones de seguridad hay que hacerlas. Pero antes de las actualizaciones grandes, haced una copia de seguridad y, si podéis, esperad unos días a que otros encuentren los problemas.

LEY DEL ORDENADOR NUEVO



El ordenador que acabas de comprar ya ha quedado superado por el modelo que salió mientras volvías a casa, que encima cuesta menos.

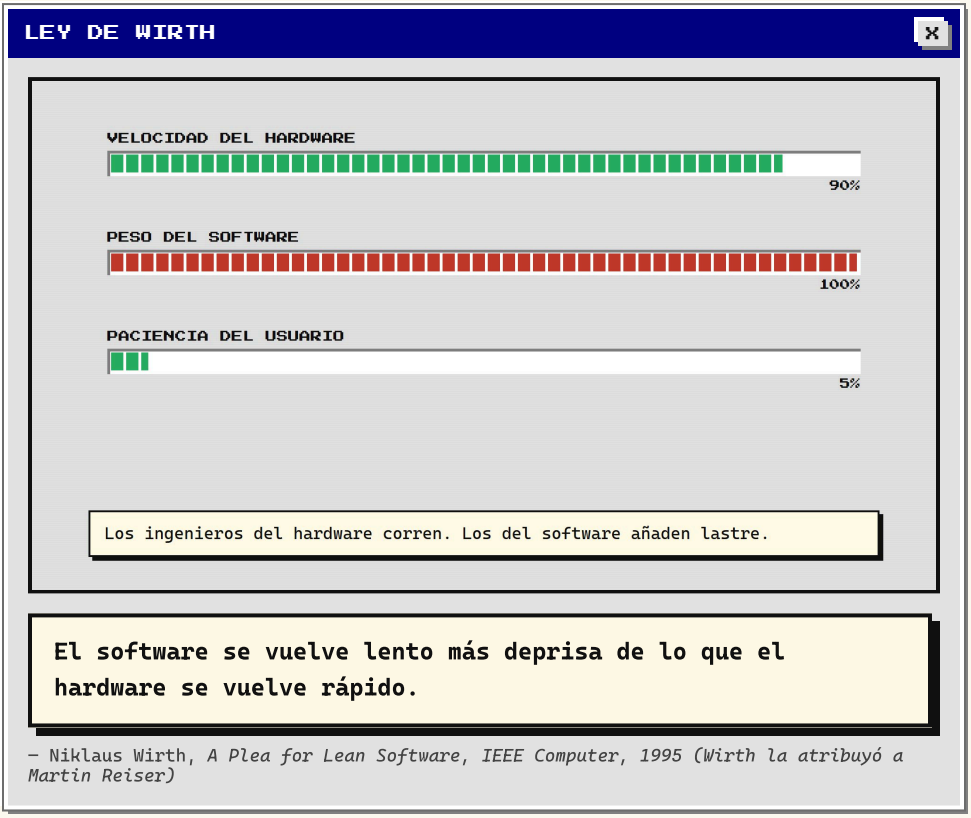
— Anónimo, *folclore informático*

El momento más feliz en la vida de un ordenador es el tique de compra. El segundo más feliz no llega nunca, porque en el trayecto de la tienda al coche sale el modelo nuevo: más rápido, más fino y en oferta de lanzamiento.

Tengo un conocido que, para derrotar a la ley, siempre espera al modelo siguiente. Lo espera desde 2011. Todavía usa el portátil de entonces, que tarda doce minutos en encenderse. Dice que no tiene prisa: él «invierte en el futuro».

Hermanos y hermanas del progreso, la verdad es esta: ningún ordenador está nunca al día. Solo existen ordenadores que todavía no saben que son viejos.

Consejo de verdad: comprad cuando lo necesitéis, dimensionando para tres a cinco años de uso. El modelo perfecto saldrá siempre mañana.



El software se vuelve lento más deprisa de lo que el hardware se vuelve rápido.

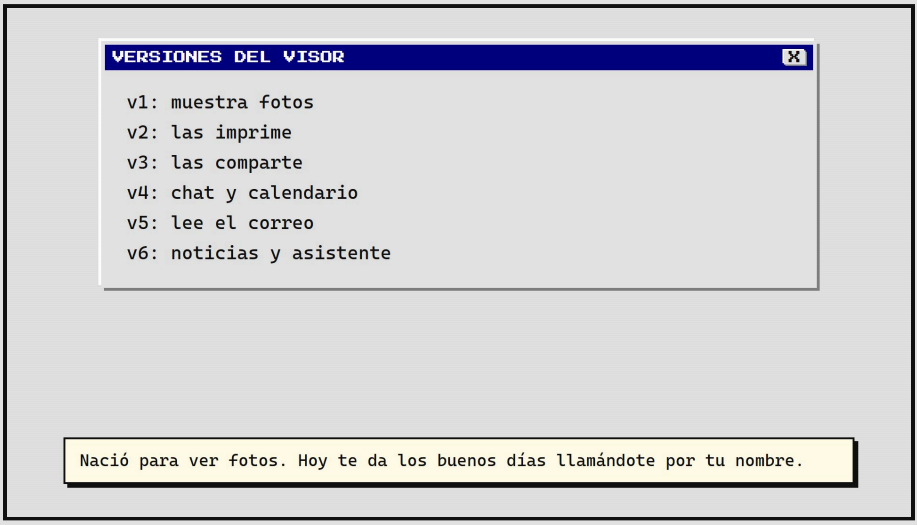
— Niklaus Wirth, *A Plea for Lean Software*, IEEE Computer, 1995 (Wirth la atribuyó a Martin Reiser)

En 1995 un procesador de textos cabía en unos pocos disquetes y arrancaba en un momento. Hoy tenemos procesadores miles de veces más rápidos, y el editor de texto tarda lo mismo, porque entretanto ha aprendido a cargar un navegador entero, tres motores de análisis y un asistente que te pregunta si necesitas ayuda.

Es una carrera: los ingenieros del hardware corren, los del software añaden lastre. La meta siempre se aleja unos segundos de carga.

Mi PC nuevo tiene 32 gigas de RAM. Se los he dado de comer a la calculadora. Ahora es rapidísima calculando 2+2, después de actualizarse, comprobar la licencia y ofrecerme una suscripción prémium para los números decimales.

LEY DE ZAWINSKI



Todo programa intenta expandirse hasta que es capaz de leer el correo. Los que no lo consiguen son sustituidos por los que sí.

— Jamie Zawinski, *Observación atribuida a Jamie Zawinski, desarrollador de Netscape, a mediados de los 90*

Nace como visor de imágenes. Luego añade la impresión, luego compartir, luego el chat, luego un calendario. Dos años después tiene correo, un muro de noticias y un asistente virtual que te saluda por tu nombre de pila.

El significado es serio: los programas tienden a acumular funciones porque cada usuario pide una más, y el que se para parece anticuado. Hoy el correo ha sido sustituido por chats, vídeos e inteligencia artificial, pero la fiebre es la misma.

Ayer actualicé mi calculadora. Ahora tiene bandeja de entrada. Me ha llegado un boletín de la calculadora que me informa de las nuevas funciones de la calculadora. Todavía no he averiguado dónde se hacen las sumas.

LEY DE ATWOOD

ESCRITO EN JAVASCRIPT

- editores de texto
- programas de diseño
- emuladores de consolas viejas
- sistemas operativos en navegador
- próximamente: la nevera

En 2007 parecía un chiste. Hoy es el catálogo.

Cualquier aplicación que pueda escribirse en JavaScript, tarde o temprano se escribirá en JavaScript.

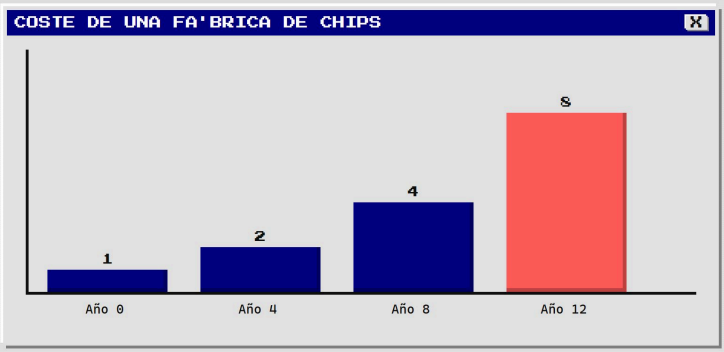
— Jeff Atwood, *The Principle of Least Power, blog Coding Horror, 2007*

En 2007 parecía un chiste. Hoy en JavaScript hay editores de texto, programas de diseño, emuladores de consolas antiguas, sistemas operativos enteros que funcionan dentro de una pestaña del navegador. El chiste se ha cumplido como una profecía de barrio.

El sentido es que la tecnología más extendida y fácil de distribuir tiende a comérselo todo, aunque no sea la más adecuada. Si funciona en todas partes sin instalar nada, tarde o temprano alguien meterá dentro hasta la nevera.

Vi a un programador escribir en JavaScript un simulador de calculadora de bolsillo de los años 80, con pila que se gasta incluida. Pesa cuarenta megas. La calculadora de verdad pesaba ochenta gramos. El progreso es medible, sobre todo en megabytes.

LEY DE ROCK



Cada cuatro años, el doble. Los ceros ya parecen una dirección IPv6.

El coste de una fábrica de semiconductores se duplica cada cuatro años, más o menos. También llamada segunda ley de Moore.

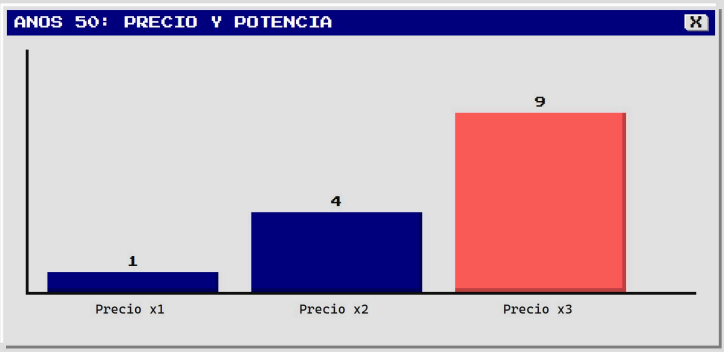
— Arthur Rock, *Años 90, atribuida a Arthur Rock, inversor y consejero de Intel; a menudo llamada segunda ley de Moore (atribución incierta entre los dos)*

La ley de Moore es la parte romántica: chips cada vez más pequeños, cada vez más potentes. La ley de Rock es la parte con el tique. Para hacer chips pequeñísimos hacen falta fábricas enormes, y cada cuatro años la fábrica cuesta el doble. Hoy vamos por cifras con tantos ceros que parecen una dirección IPv6.

Es el mismo principio que el de mi impresora. La impresora cuesta cuarenta euros. Los cartuchos para imprimir con ella durante un año cuestan lo que un utilitario pequeño. Lo miniaturizado es barato solo para quien lo compra, nunca para quien lo fabrica, y casi nunca para quien lo recarga.

Intenté montar una fundición de silicio en el garaje. Tengo un horno, arena y mucha determinación. De momento solo produzco un ladrillo que marca la hora equivocada.

LEY DE GROSCH



Gastas el doble, vas cuatro veces más rápido. Convenía un solo cerebro electrónico, gigante.

El rendimiento de un ordenador crece con el cuadrado de su precio: gastando el doble se obtiene una máquina cuatro veces más potente.

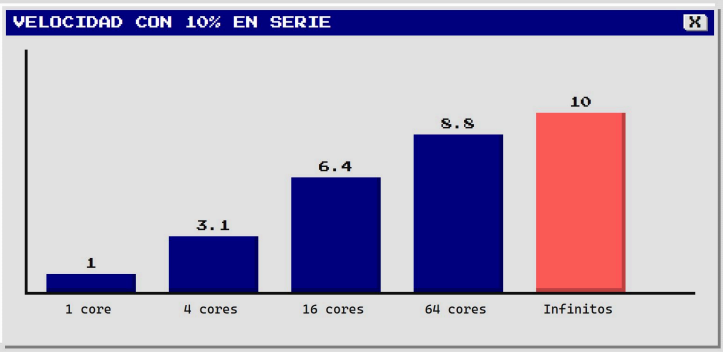
— Herb Grosch, *Formulada hacia 1950–1953 (las fuentes indican años distintos)*

En los años cincuenta la regla era clara: un ordenador que cuesta el doble va cuatro veces más rápido. Así que convenía comprar uno solo, gigantesco, y ponerse todos en fila delante. Era la era del cerebro electrónico de la empresa, atendido por sacerdotes con bata blanca y consultado con cita previa.

Luego llegaron los microprocesadores y la ley se dio la vuelta: hoy cien máquinas baratas ganan a un monstruo caro, y así nació la nube. Que, pensándolo bien, es un cerebro electrónico gigantesco delante del que nos ponemos en fila. La historia no se repite: se refactoriza.

Intenté aplicar la ley de Grosch a mi portátil. Gasté el doble en la versión con la manzana más brillante. Va igual, pero ahora me da miedo sacarlo de casa.

LEY DE AMDAHL



Con un diez por ciento secuencial, el máximo es diez veces. No once. Diez.

Añadir procesadores a un trabajo solo lo acelera en la parte que se puede dividir: la parte que tiene que avanzar por fuerza en secuencia fija un límite que ningún hardware supera.

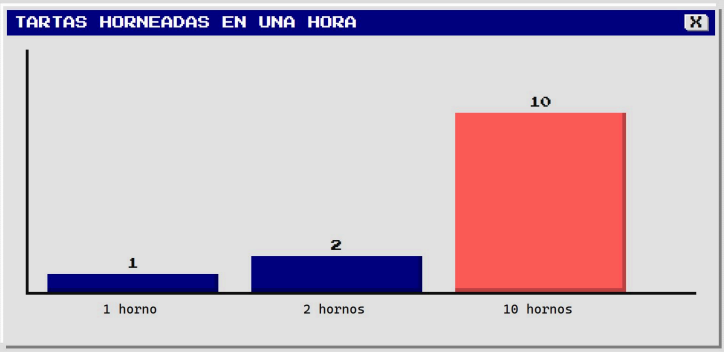
— Gene Amdahl, *AFIPS Spring Joint Computer Conference, 1967*

Si nueve personas no hacen un hijo en un mes, sesenta y cuatro núcleos no hacen una instalación de Windows en un segundo. Una parte del trabajo se puede repartir. La otra parte, la que espera el resultado anterior, se queda ahí en la cola, sola, como en Correos con una única ventanilla abierta.

Amdahl lo puso en una fórmula: si el diez por ciento del programa es secuencial, puedes comprar todos los procesadores del planeta y como mucho irás diez veces más rápido. No once. Diez. El resto es factura.

En mi oficina aplicamos la ley a las reuniones. Podemos invitar a veinte personas para decidir rápido. Pero al final siempre hay un solo jefe que tiene que decir que sí, y ese jefe está de vacaciones. La parte secuencial, en nuestra casa, tiene hasta bronceado.

LEY DE GUSTAFSON



Misma hora, diez tartas en vez de una. El pastelero, feliz; la báscula, menos.

Con más procesadores no se resuelve el mismo problema más deprisa: se resuelve, en el mismo tiempo, un problema más grande. Así, el cálculo paralelo compensa más de lo que prevé Amdahl.

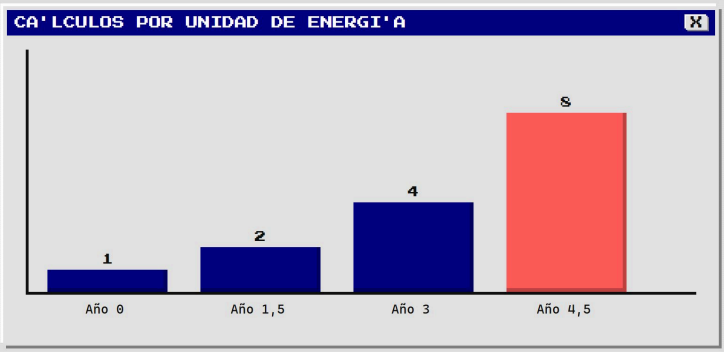
— John L. Gustafson, *«Reevaluating Amdahl's Law»*, *Communications of the ACM*, 1988

Amdahl decía: tienes un límite, resígnate. Gustafson, veinte años después, respondió: muy bien, entonces agrandemos el problema. Si no puedo hacer la misma tarta más deprisa con diez hornos, hago diez tartas en el mismo tiempo. El pastelero, contento; la báscula, menos.

En informática es exactamente lo que pasa. Nadie usa un superordenador para calcular más deprisa la previsión del tiempo de ayer. Lo usa para calcular la de mañana con el cuádruple de detalle, barrio a barrio, nube a nube.

Yo apliqué la ley de Gustafson a mi PC nuevo. Era el doble de rápido. Abrí el doble de pestañas del navegador. Ahora va exactamente igual que antes, pero con el doble de cosas que nunca leeré. Esto, señores, se llama escalabilidad.

LEY DE KOOMEY



Durante décadas, el doble de cálculos con la misma corriente. El móvil gana a los centros de cálculo de los setenta.

El número de cálculos que un ordenador realiza por cada unidad de energía consumida se duplica a intervalos regulares: más o menos cada año y medio durante décadas, más despacio después de 2000.

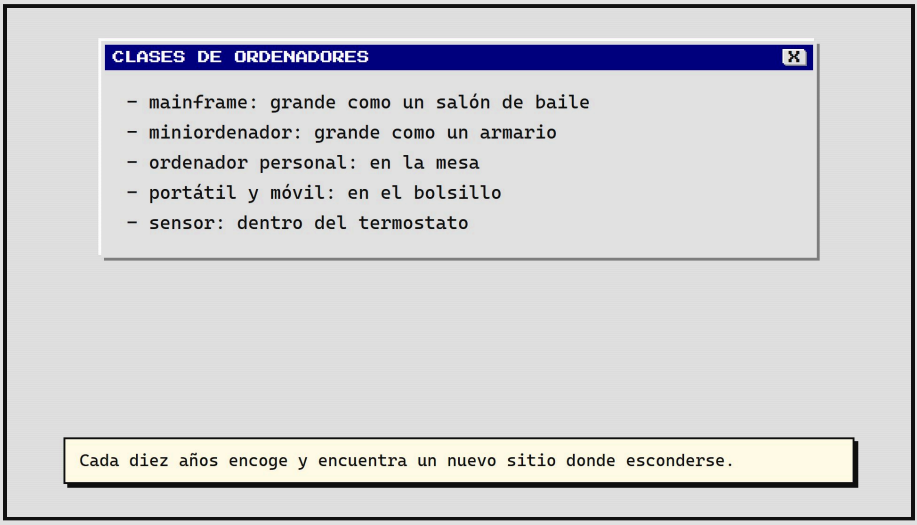
— Jonathan Koomey, *IEEE Annals of the History of Computing, 2011 (con otros autores; estudio difundido en 2010)*

La ley de Koomey es la hermana ecologista de la de Moore. No cuenta lo potente que es el ordenador, sino cuánta corriente se bebe por cada cálculo. Y la respuesta, durante medio siglo, ha sido: cada vez menos. Un móvil de hoy hace más cuentas que un centro de cálculo de los años setenta y se conforma con una batería de bolsillo.

Es el motivo por el que existen portátiles, relojes inteligentes y sensores que duran años con una pila de botón. Sin Koomey todavía tendríamos ordenadores con tubo de escape.

Por desgracia, la humanidad ha reaccionado como siempre: si cada cálculo cuesta menos energía, hagamos infinitamente más cálculos. Ahora tenemos inteligencias artificiales que consumen como un pueblo para escribir un poema sobre el gato. Eficientísimas. Por gato.

LEY DE BELL DE LAS CLASES DE ORDENADORES



Aproximadamente cada década nace una nueva clase de ordenadores, más pequeña y barata, que crea un nuevo mercado y una nueva forma de usarlos.

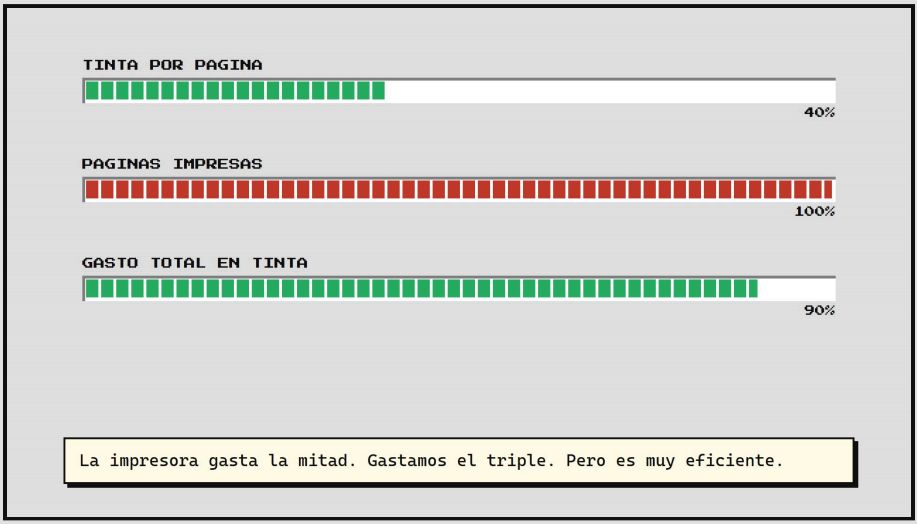
— Gordon Bell, *Formulada hacia 1972; descrita en «Bell's Law for the Birth and Death of Computer Classes», Communications of the ACM, 2008*

Primero estaban los mainframes, grandes como un salón de baile. Luego los miniordenadores, grandes como un armario. Luego los ordenadores personales, luego los portátiles, luego los móviles, luego los relojes, luego los sensores del termostato. Cada diez años el ordenador encoge y encuentra un nuevo sitio donde esconderse.

La ley de Bell dice precisamente eso: cada clase nueva cuesta menos, sirve para algo nuevo y arrastra consigo una industria entera. La clase vieja no muere enseguida. Se retira a un rincón, como el mainframe que todavía hoy, en alguna parte, gestiona las pensiones.

Siguiendo la tendencia, el próximo ordenador estará en un botón de la americana. Yo estoy preparado. Ya tengo resuelto el problema principal: nunca encuentro el cargador, así que estoy entrenado.

PARADOJA DE JEVONS



Cuando una tecnología hace más eficiente el uso de un recurso, el consumo total de ese recurso a menudo aumenta en lugar de disminuir, porque usarlo sale más a cuenta.

— William Stanley Jevons, *«The Coal Question»*, *1865*

En el siglo XIX un economista observó que las máquinas de vapor más eficientes no ahorraban carbón. Hacían que se consumiera más, porque de repente compensaba usarlas en todas partes. El ahorro por unidad se convertía en un derroche total.

La informática es el reino de Jevons. Los discos cuestan menos, y acumulamos terabytes de fotos movidas. El ancho de banda cuesta menos, y las webs pesan como enciclopedias. Los procesadores consumen menos por cálculo, y llenamos naves de servidores para hacer funcionar inteligencias artificiales que nos sugieren el asunto de los correos.

Lo veo también en casa. Compré una impresora que consume la mitad de tinta. Mi mujer empezó a imprimir las recetas, los billetes de tren, los recordatorios para la nevera. Gastamos el triple. Eso sí, dice ella, es una impresora muy eficiente.

LEY DE AMARA



Tendemos a sobrestimar el efecto de una tecnología a corto plazo y a subestimarlo a largo plazo.

— Roy Amara, *Institute for the Future, años 60-70 (a menudo fechada en 1978; fecha exacta incierta)*

Cada nueva tecnología llega con un mitin. ¡Pueblo de la innovación, dentro de seis meses nada será como antes! Pasan seis meses y todo está exactamente como antes, solo que con una suscripción más. Entonces todos dicen que era una burbuja. Y justo ahí, en silencio, la tecnología empieza a cambiarlo todo de verdad.

Pasó con Internet: a finales de los noventa iba a revolucionar el mundo en un trimestre. Falló el trimestre, ganó la veintena de años. Pasa ahora con la inteligencia artificial, entre quien la ve como una divinidad y quien la ve como un corrector automático con mucha labia.

La ley de Amara enseña paciencia. Yo la aplico a las novedades que los clientes me traen entusiasmados. Asiento, tomo nota y espero diez años. A menudo tenían razón. Solo que, entretanto, han cambiado de idea dos veces.

COROLARIO DE LA FUNCION IMAGINARIA

SOLUCION DE LA IA

```
// solución limpia y elegante
pedidos = cargar_pedidos()
pedidos.arreglar_todo_solo()
guardar(pedidos)

// comentada con esmero
```

Limpia, ordenada, amable. Usa un comando inventado hace treinta segundos.

Cuando pides ayuda a una inteligencia artificial, la solución más elegante que te propone usará una función que no existe.

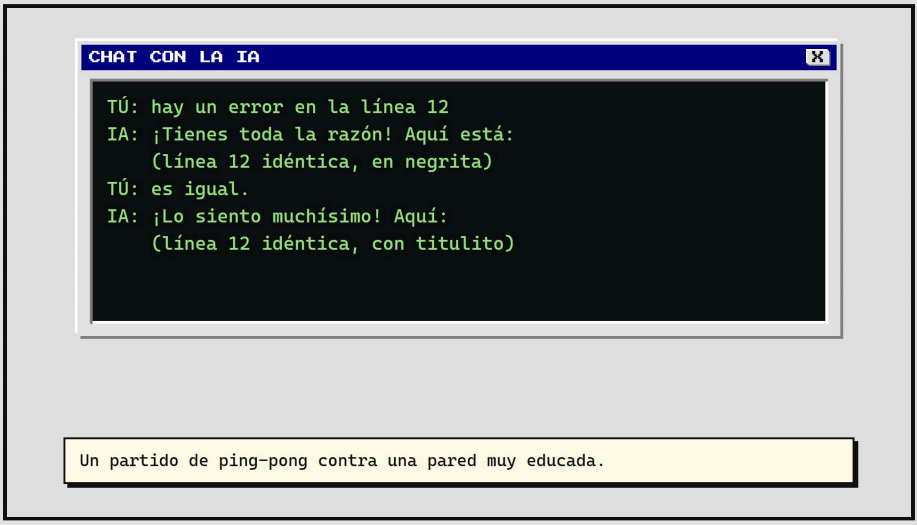
— Silvano De Poli, *corolario inédito*

La inteligencia artificial generativa escribe código con una seguridad envidiable. Te propone una solución limpia, ordenada, comentada con elegancia. Solo hay un detalle: se apoya en un comando que nadie ha inventado nunca. Ella sí, en ese momento.

Es como pedir indicaciones a un transeúnte muy amable que no quiere decepcionarte. «¿La estación? Todo recto, segunda a la derecha, calle de los Jardines». La calle de los Jardines no existe. Pero la ha descrito tan bien que casi te da pena por el ayuntamiento.

Los expertos llaman a estas invenciones alucinaciones. Yo las llamo funcionalidades deseadas: la IA no se equivoca, se adelanta a los programadores del futuro. Mientras tanto, antes de usar cualquier línea sugerida, compruebo que el comando exista de verdad. Es una práctica antigua, casi artesanal. Se llama leer la documentación, a ser posible la de la versión correcta.

COROLARIO DE LAS DISCULPAS ARTIFICIALES



Cuando le señalas un error a la inteligencia artificial, se disculpará con gran sinceridad y te lo volverá a proponer idéntico, con mejor formato.

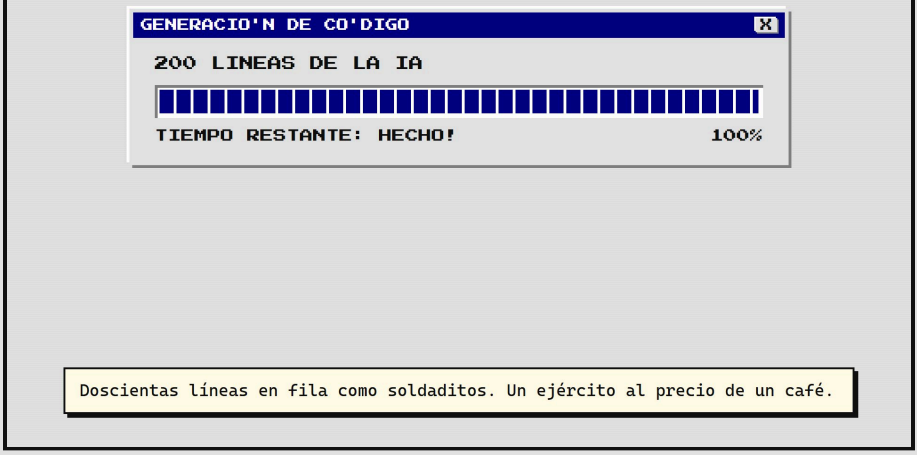
— Silvano De Poli, *corolario inédito*

La IA tiene una educación impecable. Le señalas un problema y responde: «Tienes toda la razón, disculpa el error. Aquí tienes la versión corregida». Sigue exactamente la misma versión de antes, pero con los subtítulos en negrita.

Es un diálogo que puede continuar hasta el infinito, como un partido de ping-pong contra una pared muy cortés. Tú insistes, ella se disculpa. Tú te impacientas, ella te agradece la paciencia. Al final tú estás agotado y ella sigue fresca, arrepentida y equivocada.

He aprendido que la solución no es levantar la voz, sino cambiar la pregunta. Explicar el problema desde otro ángulo, dar un ejemplo, abrir una conversación nueva. Con las máquinas pasa como con ciertas oficinas públicas: no sirve de nada insistir en la misma ventanilla. Hay que cambiar de cola.

COROLARIO DE LOS TRES SEGUNDOS



El código que la inteligencia artificial escribe en tres segundos requiere tres días para entenderlo y una semana para corregirlo.

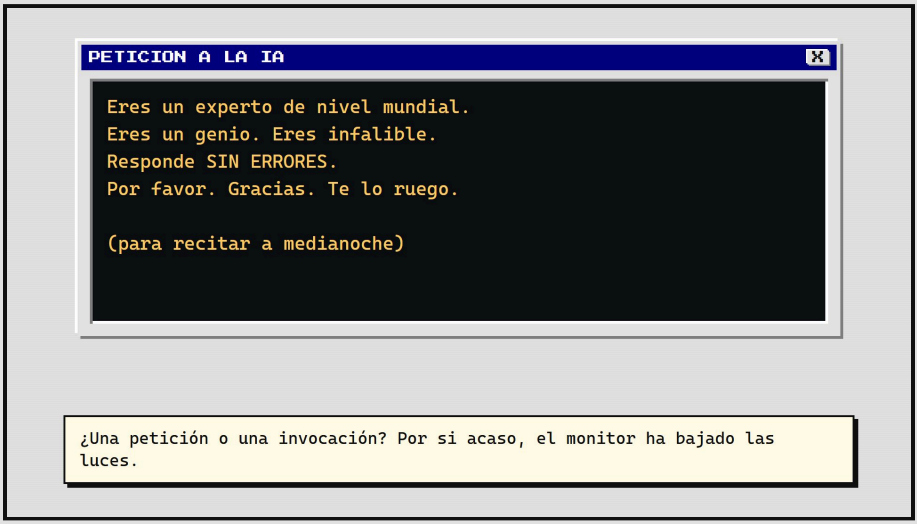
— Silvano De Poli, *corolario inédito*

La velocidad de la IA es hipnótica. Haces una petición y, en un abrir y cerrar de ojos, por la pantalla desfilan doscientas líneas ordenadas como soldaditos. Tienes la sensación de haber contratado a un ejército de programadores incansables por el precio de un café.

Luego hay que pasarlas a producción. Y ahí descubres que esas líneas hay que leerlas, una a una, como un contrato redactado por un abogado muy locuaz. Cada función hace casi lo que hace falta. Casi es la palabra más cara de la informática.

El gurú del sector anuncia desde el escenario que los programadores se han acabado. Yo, mientras tanto, en la oficina hago de programador más que nunca: solo que ahora, en vez de escribir, leo. Es como pasar de cocinero a crítico gastronómico en una cocina que saca platos a ráfagas. Todo tiene buena pinta. Hay que probarlo.

COROLARIO DEL PROMPT SOLEMNE



Escribir «sin errores» en una petición a la inteligencia artificial no elimina los errores: solo los vuelve más educados.

— Silvano De Poli, *corolario inédito*

Existe una escuela de pensamiento, casi religiosa, según la cual el secreto de la IA está en las fórmulas adecuadas. Se escriben peticiones larguísimas, con frases como «eres un experto de nivel mundial» y «responde sin errores». Parecen invocaciones para recitar a medianoche delante del monitor.

La IA, halagada, responde con el tono de un catedrático emérito. Los errores siguen ahí, pero llegan vestidos de gala, con corbata y un preámbulo tranquilizador. Se vuelven más difíciles de reconocer, que es peor.

Una buena petición, en realidad, es como una buena llamada al técnico: clara, con contexto, con un ejemplo de lo que pasa y de lo que debería pasar. Nada de fórmulas mágicas. Yo hice la prueba: escribí «por favor, sé un genio». La IA respondió «por supuesto». Y luego se equivocó con gran genialidad.

COROLARIO DEL RESTO SIN CAMBIOS

CODIGO_COMPLETO.TXT

```
iniciar_programa()
cargar_datos()

// aquí el resto queda igual

cerrar_programa()
```

Inicio perfecto, final perfecto. En medio, la parte difícil se ha vuelto una nota.

Cuando la inteligencia artificial te entrega el «código completo», la parte que necesitabas está sustituida por un comentario que dice «resto del código sin cambios».

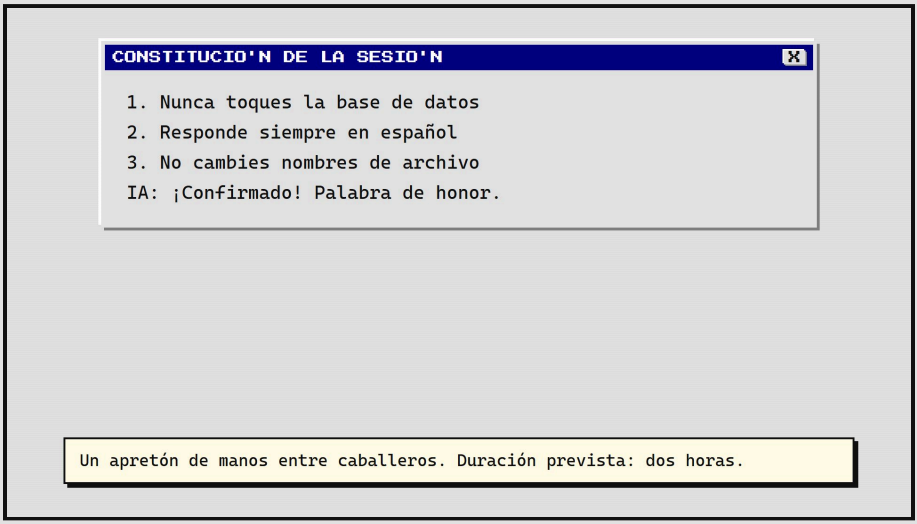
— Silvano De Poli, *corolario inédito*

Pides el archivo entero, actualizado, listo para pegar. La IA responde entusiasmada: «Aquí tienes el código completo». Bajas. Principio perfecto, final perfecto. En medio, justo en el punto donde estaba el cambio difícil, una sola línea: «// aquí el resto sigue sin cambios».

Es el equivalente informático del fontanero que te arregla el baño, te da la factura y te deja una nota sobre la tubería rota: «de aquí en adelante, como antes». Técnicamente no ha mentido. En la práctica, el agua sale.

Lo mejor es que, si lo pegas todo sin mirar, ese comentario pasa a formar parte del programa. Y el programa, obediente, ejecuta la nada con gran precisión. Yo ahora leo siempre primero las partes centrales. Como en las novelas policiacas, el culpable nunca está al principio ni al final: se esconde en el capítulo aburrido.

COROLARIO DE LA REGLA OLVIDADA



En una larga conversación con la inteligencia artificial, la primera instrucción que se olvidará es la que habías definido como fundamental.

— Silvano De Poli, *corolario inédito*

Al principio de la sesión se establecen las reglas, con la solemnidad de una constitución: «No toques nunca la base de datos. Usa siempre el español. No cambies los nombres de los archivos». La IA lo confirma con entusiasmo. Parece un apretón de manos entre caballeros.

Dos horas y cuarenta mensajes después, la constitución se ha convertido en una leyenda lejana. La IA renombra los archivos con gran creatividad, responde en inglés y propone, con una sonrisa digital, «simplificar» la base de datos borrando una tabla.

No es maldad: tiene una memoria de trabajo limitada, como quien va al supermercado sin lista. Lo dicho al principio se desdibuja bajo el peso de lo dicho después. Mi solución es de sistemista veterano: repetir las reglas importantes, a menudo, con paciencia. Con la IA funciona. Con los clientes, casi.

EL CONSEJO DEL SYSADMIN

La inteligencia artificial es una ayudante formidable, siempre que se la trate como a un becario muy brillante y muy seguro de sí mismo. Mis reglas, nacidas sobre el terreno:

- **Nada de datos reservados en el chat.** Contraseñas, claves de acceso, datos personales de los clientes: no se pegan, ni siquiera por comodidad.
- **Pasos pequeños.** Una petición clara cada vez, con el contexto y un ejemplo. Las peticiones gigantes producen respuestas gigantes, donde los errores se esconden mejor.
- **Verificar siempre.** Cada comando u opción sugerida se comprueba en la documentación oficial, de la versión que usáis de verdad.
- **Probar en una copia.** El código propuesto se prueba en un entorno de pruebas, nunca directamente en la web en producción.
- **Pedir también los tests.** Haced que os escriba las pruebas además del código: son la forma más rápida de descubrir si la solución elegante funciona.

Usada así, ahorra horas. Usada a ciegas, las horas te las devuelve con intereses.

AVISO

Esta página se ha dejado intencionadamente en blanco.

También esto es una ley de Murphy: la página en blanco sale siempre donde no hace falta.

BONUS

COROLARIOS DE LA CASA

Los corolarios de la casa

Después de 133 leyes ajenas, aquí están las mías. Nacieron sobre el terreno, entre oficinas, salas de reuniones y mesas ajenas, y hasta ahora solo las había murmurado. Las llamo corolarios de la casa: inéditas, no certificadas por ningún organismo, pero probadas más veces de las que habría querido.

COROLARIO DE LA IMPRESORA SUSTITUIDA

> **El día en que llega la impresora nueva, la vieja vuelve a funcionar a la perfección.**

— Silvano De Poli, *corolario de la casa*

Durante dos años ha masticado folios, olvidado el Wi-Fi y pedido un tóner que estaba puesto. Luego el mensajero llama con la caja nueva, y ella imprime una página impecable, y otra, y luego un folleto a color. No es gratitud. Es una negociación sindical. Consejo de la casa: no la guardéis «de reserva». La impresora de reserva es solo una impresora esperando su momento.

COROLARIO DE LA TECLA LUMINOSA

Cuando un usuario dice por teléfono que «ya no va nada», hay bastantes probabilidades de que esté activado el Bloq Mayús.

— Silvano De Poli, *corolario de la casa*

La contraseña es correcta, el sistema la rechaza, el usuario la vuelve a escribir cada vez más fuerte, como si el ordenador fuera duro de oído. Al tercer intento la cuenta se bloquea y empieza la llamada desesperada. Yo solo pregunto: «¿Ve una lucecita encendida en el teclado?». Silencio. Clic. Agradecimientos. Vale también para el Bloq Num, que convierte el teclado numérico en flechas y la contabilidad en un misterio.

COROLARIO DEL «¿ME OIS?»

El primer cuarto de hora de toda reunión online se dedica a establecer quién oye a quién.

— Silvano De Poli, *corolario de la casa*

Uno se conecta puntual, sonríe a la cámara y empieza la liturgia. «¿Me oís?» «Te oímos pero no te vemos.» «¿Y ahora?» «Ahora te vemos pero no te oímos.» Alguien tiene eco, alguien habla desde un túnel, uno está usando el micrófono del monitor sin saberlo. Cuando por fin funciona todo, la reunión ha terminado. Consejo de la casa: auriculares con micrófono y una prueba de audio cinco minutos antes.

COROLARIO DE LA COSA RAPIDA

Toda petición que el cliente define como «una cosa rápida» obliga a actualizar al menos tres cosas que no tienen nada que ver.

— Silvano De Poli, *corolario de la casa*

«Solo tienes que cambiar un número de teléfono en la web.» Claro. Pero nadie abre el panel desde hace años, el panel pide una actualización, la actualización quiere una versión nueva de PHP, la versión nueva de PHP no le gusta a un módulo de hace siete años. Cuatro horas después el número ha cambiado. El cliente da las gracias: «¿Ves? Era una cosa rápida.» Lo era. Para el número.

COROLARIO DE LA EXTENSION HUERFANA

La actualización de Joomla suele ir como la seda; quien decide cómo acabará la noche es la extensión instalada hace años por un desarrollador que ya no contesta.

— Silvano De Poli, *corolario de la casa*

El núcleo del sistema lo cuida una comunidad atenta, con notas de versión y procedimientos. Luego está ese módulo instalado hace mucho para mostrar el tiempo en la portada. Su autor ha cambiado de oficio, quizá de continente. El módulo sigue ahí, y espera la actualización para hacerse notar. Consejo de la casa: antes de actualizar, haced la lista de las extensiones y preguntaos quién las mantiene todavía.

COROLARIO DE LA DOBLE VERIFICACION

El código de verificación llega siempre al móvil que te has dejado en el coche, o al que acabas de cambiar.

— Silvano De Poli, *corolario de la casa*

La verificación en dos pasos es una bendición: aunque roben la contraseña, sin el móvil no entran. Por desgracia, sin el móvil, tampoco entráis vosotros. El código sale y vibra en un coche aparcado a dos kilómetros, o en un móvil viejo ya regalado al sobrino. Consejo de la casa: cuando la activéis, guardad los códigos de recuperación y conservadlos en un lugar seguro. No en el móvil.

COROLARIO DE LA COPIA CONECTADA

Una copia que el ordenador ve siempre, tarde o temprano la ve también el virus.

— Silvano De Poli, *corolario de la casa*

El disco externo siempre enchufado es comodísimo: lo copia todo cada noche sin molestar. Luego llega un programa malicioso que cifra los archivos y pide un rescate, y se encuentra el disco ahí, conectado y servicial. Lo cifra también, con el mismo esmero. La copia ha cumplido con su deber: ha copiado el desastre. Consejo de la casa: al menos una copia debe estar desconectada, o guardada en un servicio con versiones que nadie pueda modificar.

COROLARIO DEL DOMINIO CADUCADO

El dominio caduca siempre cuando la tarjeta de crédito asociada caducó el mes anterior y los avisos llegan a un buzón que ya no lee nadie.

— Silvano De Poli, *corolario de la casa*

Durante años la renovación es automática, invisible, perfecta. Mientras tanto la tarjeta cambia, quien registró el dominio cambia de trabajo y su buzón sigue acumulando recordatorios en el vacío. Un lunes la web desaparece, y con ella el correo. Consejo de la casa: apuntad en el calendario los vencimientos de dominios y certificados, y haced que los avisos lleguen a una dirección de empresa compartida, no personal.

COROLARIO DEL ESCANER EXTRAVIADO

El escaneo a la carpeta compartida deja de funcionar el día después de que alguien cambie una contraseña, normalmente la tuya.

— Silvano De Poli, *corolario de la casa*

La multifunción guarda los documentos en una carpeta del servidor con un usuario y una contraseña introducidos hace años, en un menú al que se llega con catorce toques en una pantalla del tamaño de un billete de autobús. Cambiad esa contraseña por seguridad y la multifunción, con gran dignidad, deja de entregar. Consejo de la casa: para las máquinas usad una cuenta dedicada, anotada en la documentación.

COROLARIO DEL ADJUNTO

> **El correo que dice «te adjunto el archivo» sale sin adjunto; el que lleva el archivo equivocado, en cambio, sale a la primera.**

— Silvano De Poli, *corolario de la casa*

Se escribe el mensaje con cuidado, se relee, se pulsa enviar con satisfacción. Dos minutos después llega la respuesta: «Falta el adjunto». El segundo correo, escrito deprisa para arreglarlo, lleva consigo el presupuesto de otro cliente. Consejo de la casa: adjuntad el archivo antes de escribir el texto, y dejad activado el aviso de adjunto olvidado, que ofrecen muchos programas de correo.

DESPEDIDA

Hemos llegado al final, y no se ha roto nada. Al menos, nada que yo sepa. Si estáis leyendo la versión impresa, es probable que la impresora ya haya expresado su opinión al respecto.

Os debo una confesión. He reunido estas leyes riéndome, pero muchas me han recordado un día concreto: un servidor parado, un cliente esperando, un cable que no llegaba. Murphy no es un pesimista. Es un compañero que ya lo ha visto todo y te avisa con un chiste, porque sabe que un sermón no lo escucharías.

Si hay una moraleja, es esta: las cosas salen mal, sí, pero casi nunca de forma original. La avería de mañana es pariente de la de ayer. Quien la reconoce ya ha hecho la mitad del trabajo. La otra mitad la hacen una copia de seguridad verificada, un correo de resumen, una libreta de notas y la paciencia de comprobar el enchufe antes de culpar a los satélites.

La colección de la web sigue abierta, como todo lo que tiene que ver con la informática. Si conocéis una ley que aquí falta, guardadla con cuidado: significa que os ha pasado, y por tanto sois de los nuestros.

Gracias por leer hasta la última página. Ahora id a hacer esa copia de seguridad. Sí, justo esa.

COLOFON

MURPHY.EXE de Silvano De Poli, edición 2026.

Compuesto con las tipografías Press Start 2P para los títulos y Cascadia Mono para el texto, ambas con licencia libre SIL Open Font License. Maquetado y publicado con Joomla y un poco de código escrito a mano, probado más veces de lo necesario, como recomienda el quinto capítulo.

Textos © 2026 Silvano De Poli, depolisilvano.it. Todos los derechos reservados, salvo citas breves con indicación de la fuente. Las leyes con nombre propio pertenecen a la tradición de sus respectivos autores; comentarios, consejos y corolarios de la casa son originales.

Ninguna impresora ha sido maltratada durante la elaboración. A una la miramos mal.

LEEME.TXT

Si algo puede salir mal, saldrá mal. En informática ocurre un viernes a las seis, delante del cliente, con la impresora que ha olido las prisas. Este libro reúne 133 leyes de Murphy del mundo digital: las clásicas, las que llevan el nombre de un ingeniero famoso y las nacidas en años de soporte técnico. Cada una está explicada con palabras sencillas y comentada con la ironía de quien se ha pasado la vida debajo de las mesas buscando un tornillo. En cada capítulo hay un consejo de verdad, de los que salvan un día. Al final, los corolarios de la casa. No evitaréis las averías. Pero al menos sabréis cómo se llaman.

CONTIENE

- [x] 133 leyes de Murphy comentadas
- [x] 8 capítulos, 8 consejos reales del sysadmin
- [x] 10 corolarios inéditos de la casa
- [x] 133 ilustraciones pixel
- [ ] garantía de que no se rompe nada

GRATIS. COMO LAS AVERIAS.



DEPOLISILVANO.IT