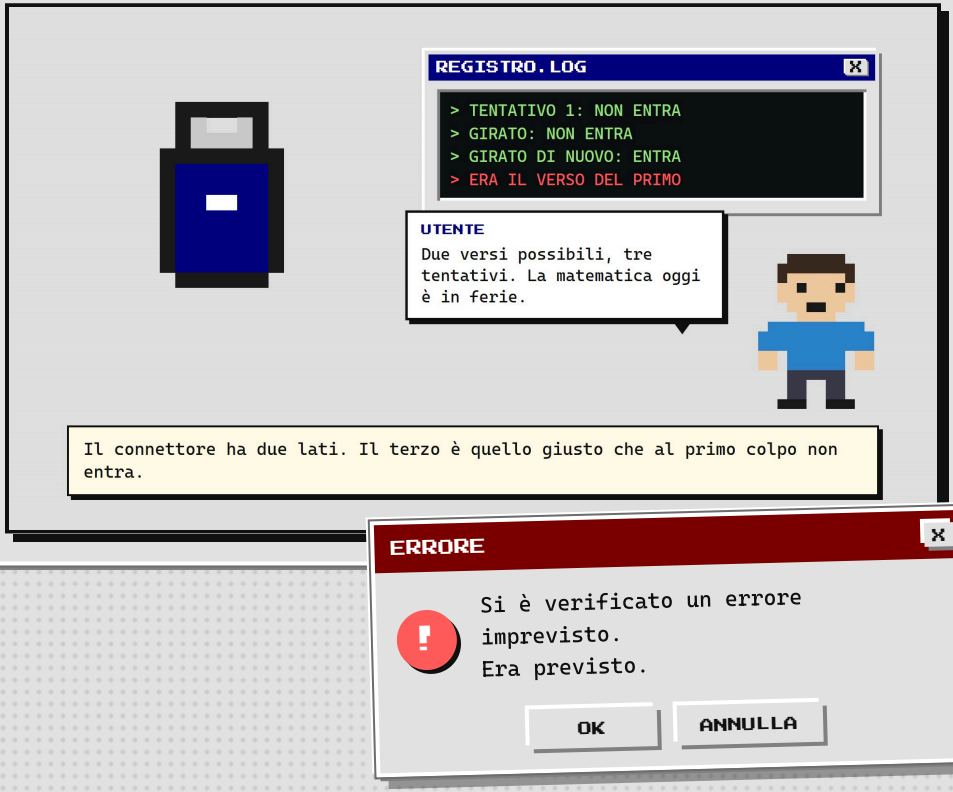
LEGGI_DI_MURPHY.EXE

MURPHY.EXE

Le leggi di Murphy dell'informatica, commentate da un sistemista che le ha viste all'opera



C:\MURPHY> leggi.exe /tutte /commentate
133 leggi trovate. 0 errori. (per ora)_

SILVANO DE POLI

133 LEGGI + COROLLARI DELLA CASA

MURPHY.EXE

Le leggi di Murphy dell'informatica, commentate da un sistemista che le ha viste all'opera

SILVANO DE POLI

A tutte le stampanti che mi hanno insegnato l'umiltà.
E a chi ha fatto il backup ieri sera: voi sapete perché.

INDICE

Prefazione 5
Istruzioni per l'uso 6
1. Le leggi fondamentali 7
2. Ferraglia: stampanti, cavi e dischi che ascoltano 23
3. Helpdesk: utenti, password e altri misteri 45
4. Reti e Internet: il filo invisibile 63
5. Caccia al bug 81
6. L'arte di scrivere codice (e di pentirsene) 101
7. Progetti, riunioni e clienti 121
8. Progresso, aggiornamenti e intelligenza artificiale 143
I corollari della casa 167
Congedo 179
Colophon 180

PREFAZIONE

Mi chiamo Silvano De Poli e di mestiere aggiusto cose che si sono rotte da sole. Faccio il sistemista e il tecnico PC, costruisco siti, lavoro con l'intelligenza artificiale e sono il responsabile dell'agenzia web RetroAI. Detto in parole povere: quando qualcosa in ufficio smette di funzionare, il telefono che squilla di solito è il mio.

In tutti questi anni ho imparato che i guasti non sono casuali. Sembrano casuali, ma hanno abitudini. Preferiscono il venerdì. Amano le dimostrazioni ai clienti. Si nascondono quando arriva il tecnico e ricompaiono appena riparte la macchina. Un giorno, davanti a una stampante che si era inceppata proprio sull'ultimo foglio di carta intestata, ho capito che non ero sfortunato. Ero dentro una legge.

Quella legge, e molte altre, esistevano già. Edward Murphy, ingegnere, ha dato il nome alla più famosa: se una cosa può andare storta, lo farà. Dopo di lui programmatori, economisti, scrittori e ricercatori hanno aggiunto le loro osservazioni, alcune serissime, altre nate come battute e diventate proverbi. L'informatica le ha raccolte tutte, con la stessa cura con cui conserva i cavi nel cassetto: non si sa mai.

Questa raccolta è nata sul mio sito, una legge alla volta. All'inizio erano appunti: la legge, chi l'aveva formulata, un commento. Poi i commenti sono cresciuti, sono arrivati i temi, qualche legge è diventata anche un piccolo video in pixel art. A un certo punto erano centotrentatré, e mi sono accorto che insieme raccontavano una storia: quella del rapporto, affettuoso e complicato, tra le persone e le macchine.

Nel libro ci sono tre tipi di leggi. Ci sono quelle **con un nome proprio**, legate a una persona reale: Moore, Brooks, Parkinson, Hofstadter e molti altri. Per queste ho cercato di spiegare cosa dicono davvero, prima di prenderle in giro. Ci sono quelle **anonime**, tramandate da ufficio a ufficio come le ricette di famiglia, che nessuno ha firmato perché tutti le hanno vissute. E ci sono alcuni **corollari miei**, soprattutto sull'intelligenza artificiale, che firmo con un certo imbarazzo e molto orgoglio.

Ogni legge ha un commento. Il tono è quello del sito: ironico, a volte un po' surreale, sempre affettuoso. Gli aneddoti più assurdi sono, ve lo garantisco, inventati. Quelli più credibili pure, ma lo dico con meno sicurezza. Le spiegazioni tecniche invece sono vere, e ho cercato di renderle comprensibili anche a chi non ha mai aperto un computer, se non per cercare una vite.

Alla fine di ogni capitolo c'è *Il consiglio del sistemista*. È la parte seria, quella che vorrei vi restasse in tasca: cose pratiche, collaudate, che servono davvero. Non sconfiggono Murphy, perché Murphy non si sconfigge. Ma lo costringono a lavorare di più.

Il libro si può leggere in ordine o a caso. Si può tenere sulla scrivania come un breviario e aprire prima di un aggiornamento importante. Si può citare in riunione per sembrare saggi, o regalare al collega che continua a dire «sul mio computer funziona». Funziona anche quello. Sul suo computer.

Un'ultima avvertenza. Mentre mettevo in ordine queste pagine, il computer si è bloccato due volte, il Wi-Fi è sparito durante una ricerca importante e la stampante si è rifiutata di collaborare alla prova di stampa. Non lo racconto per lamentarmi. Lo allego come certificato di autenticità. Le leggi funzionano. Buona lettura, e salvate spesso.

ISTRUZIONI PER L'USO

Manuale d'uso, modello MURPHY.EXE. Leggere attentamente prima dell'accensione.

- **Avvio.** Il libro si può aprire da qualsiasi pagina. Non è richiesto il riavvio.
- **Ordine di lettura.** I capitoli vanno dal più universale al più tecnico. Chi non programma può saltare il quinto e il sesto senza danni permanenti, ma se li legge capirà finalmente il suo programmatore.
- **Dosaggio.** Si consiglia una legge al giorno, preferibilmente prima di toccare un server. In caso di sovradosaggio possono comparire citazioni involontarie durante le riunioni.
- **Consigli del sistemista.** In fondo a ogni capitolo. Sono le uniche parti serie del libro: vanno applicate, non solo lette.
- **Compatibilità.** Funziona a schermo, su carta e ad alta voce. Testato anche in stampa, con esiti alterni.
- **Garanzia.** Il libro non impedisce i guasti. Li rende solo più riconoscibili. In caso di malfunzionamento, provare a rileggere.

01

CAPITOLO 1

FONDAMENTA

Le leggi fondamentali

Ogni religione ha i suoi testi sacri, e l'informatica ha queste leggi. Non le ha scolpite nessuno nella pietra: sono state ricavate sul campo, una figuraccia alla volta, da ingegneri, programmatori, economisti e persone che avevano solo bisogno di stampare un modulo.

Si comincia dalla legge madre, quella di Murphy, che in origine era un consiglio di progettazione ed è diventata il nostro oroscopo quotidiano. Seguono le parenti strette: chi aggiunge l'orario del disastro, chi ricorda che il computer obbedisce alla lettera, chi sostiene che un programma partito al primo colpo sta solo prendendo la rincorsa.

In mezzo ci sono anche leggi nate fuori dall'informatica, nella burocrazia, nell'economia, nella fantascienza. Le abbiamo adottate perché ci descrivevano troppo bene: il lavoro che si allunga fino a riempire il tempo, il collega promosso fino al punto sbagliato, l'ottanta per cento dei guai che arriva sempre dallo stesso venti per cento.

È il capitolo da leggere per primo e da citare in riunione. Queste leggi funzionano come le frasi in latino: chi le pronuncia sembra subito più saggio, anche se il server è ancora giù.

CONTENUTO.DIR

01 Legge di Murphy
02 Legge di Finagle
03 Legge del computer letterale
04 Legge del bug residuo
05 Legge del successo sospetto
06 Legge di Hofstadter
07 Legge di Parkinson
08 Principio di Peter
09 Principio di Pareto (80/20)
10 Rasoio di Hanlon
11 Terza legge di Clarke
12 Legge di Sturgeon
13 Legge di Moore

LEGGE DI MURPHY



Se una cosa può andare storta, prima o poi andrà storta. Vale soprattutto per i connettori montabili in due versi.

— Edward A. Murphy Jr., *Base aerea di Edwards, progetto MX981, 1949 circa; formulazione esatta e circostanze contestate*

Tutto nasce da un ingegnere, dei sensori montati al contrario e un esperimento rovinato. In origine non era pessimismo: era un consiglio di progettazione. Se un pezzo si può montare male, qualcuno lo farà, quindi progettalo in modo che non si possa.

Noi informatici l'abbiamo presa come una profezia. Il cavo USB ha tre facce: quella giusta, quella sbagliata e quella giusta che però al primo tentativo non entra. Lo dice la fisica quantistica delle porte.

Io la applico con rigore burocratico. Prima di ogni aggiornamento compilo un modulo: «Cose che possono andare storte». Il modulo è di quaranta pagine. L'aggiornamento si blocca a pagina quarantuno, che non avevo previsto.

LEGGE DI FINAGLE



Tutto ciò che può andare storto andrà storto, e sceglierà il momento peggiore possibile per farlo.

— Anonimo, *Variante della Legge di Murphy diffusa negli ambienti della fantascienza americana dagli anni '50–'60; attribuzione incerta*

Murphy dice che il disastro arriverà. Finagle aggiunge l'orario: venerdì alle 17:58, mentre stai già spegnendo il monitor. È la versione con il calendario sincronizzato.

Il server non si rompe mai il martedì mattina, quando sei riposato e hai il caffè. Si rompe durante la demo al cliente, il giorno della chiusura dei bilanci, o quando sei al matrimonio di tua cugina con il portatile chiuso in macchina. Il guasto ha un'agenda, e la tua è sempre consultabile.

Ho provato a fregarlo: ho finto di andare in ferie. Il server è rimasto impassibile. Poi, il giorno che sono rientrato davvero, ha preso fuoco l'alimentatore. Non so come lo sapesse. Sospetto il calendario condiviso.

LEGGE DEL COMPUTER LETTERALE

PULIZIA_CLIENTI

```
cancella clienti dove:
   ultimo_ordine prima del 2020
   oppure 1 = 1

> 4.218 clienti cancellati
> Desidera altro? :)
```

Un errore di una riga. Il computer ha eseguito con zelo burocratico.

Il computer fa esattamente quello che gli hai detto di fare, mai quello che volevi che facesse.

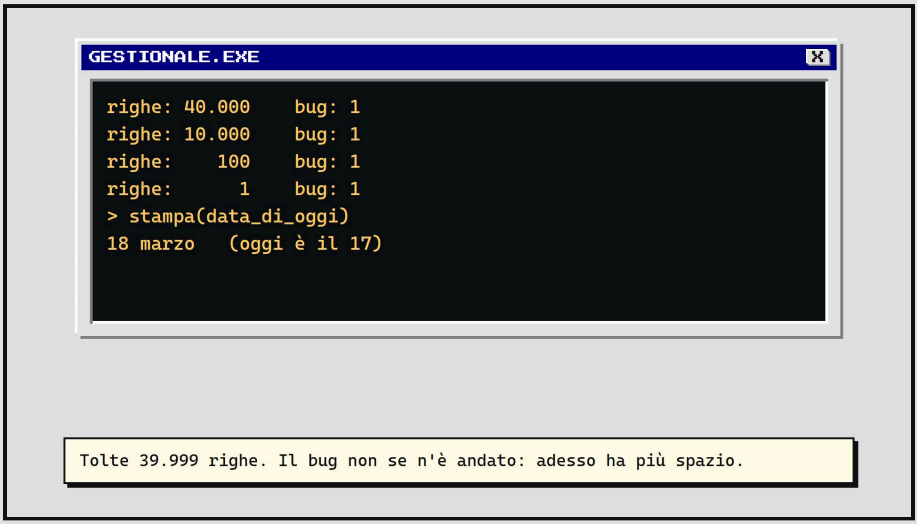
— Anonimo, *folklore informatico*

Il computer è il dipendente più obbediente della storia e, proprio per questo, il più pericoloso. Non interpreta, non intuisce, non ti guarda in faccia per capire cosa intendevi. Esegue. Alla lettera. Con zelo burocratico.

Se gli dici «cancella i clienti inattivi» e per un errore di virgola i clienti inattivi sono tutti, li cancella tutti, con efficienza ammirevole. Poi ti chiede se vuoi altro. Gentilissimo.

È come avere un genio della lampada che ha studiato giurisprudenza. Ogni desiderio va formulato con clausole, eccezioni e note a margine. Per i non programmatori questa è la vera essenza del mestiere: non spiegare al computer cosa fare, ma prevedere tutte le maniere creative in cui potrebbe fraintenderti. Io, per sicurezza, ormai parlo così anche al cane.

LEGGE DEL BUG RESIDUO



Ogni programma contiene almeno un bug e ogni programma può essere accorciato di un'istruzione. Ne consegue, per induzione, che ogni programma può essere ridotto a una sola istruzione che non funziona.

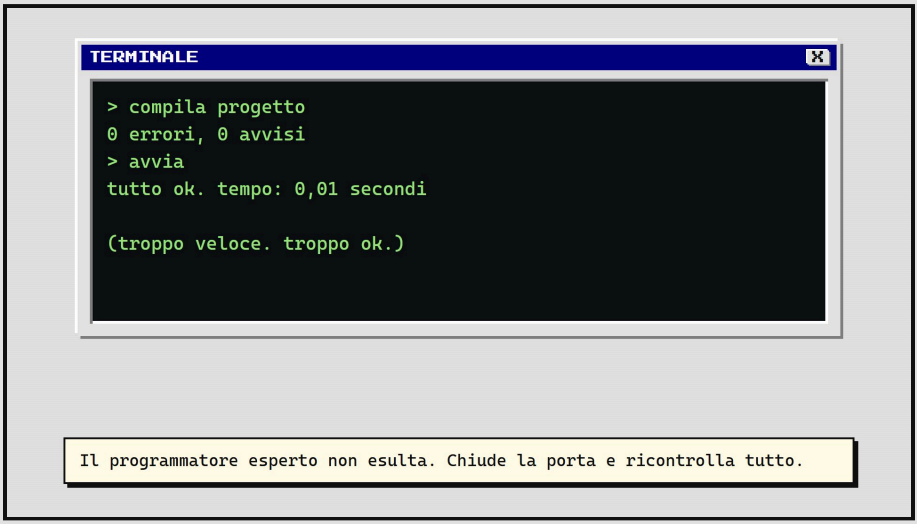
— Anonimo, *folklore informatico*

È il teorema fondamentale della programmazione, e come tutti i teoremi seri nessuno l'ha mai dimostrato perché tutti erano occupati a correggere qualcos'altro. La logica è impeccabile: togli una riga alla volta, il bug resta sempre lì, aggrappato come una cozza allo scoglio.

Io ho provato a verificarlo in laboratorio. Ho preso un gestionale di quarantamila righe e ho cominciato a togliere. Arrivato a una riga sola, il programma stampava la data. Sbagliata. Di un giorno. Solo nei mesi con la erre.

Per chi non programma, la morale è semplice: il software perfetto esiste, ma solo finché nessuno lo scrive. Appena premi il primo tasto, l'errore si siede in prima fila e prende appunti.

LEGGE DEL SUCCESSO SOSPETTO



Se un programma funziona al primo tentativo, non è un buon segno: significa che l'errore è più profondo e si farà vivo al momento meno opportuno.

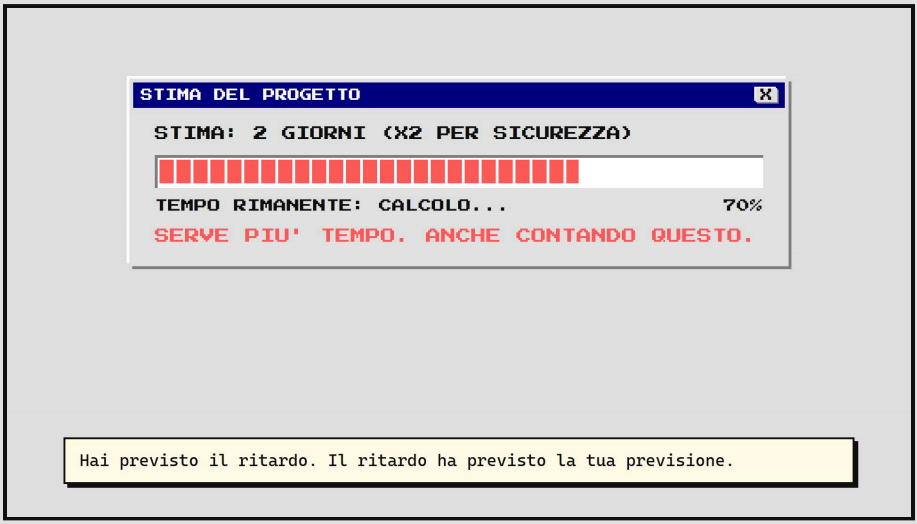
— Anonimo, *folklore informatico*

Un programmatore esperto, quando il codice gira subito senza errori, non esulta. Impallidisce. Si alza, chiude la porta, e ricontrolla tutto come un artificiere davanti a un pacco che ticchetta troppo poco.

Il motivo è statistico. Il software è un campo minato dove ogni passo è sbagliato almeno una volta. Se non è esploso niente, vuol dire soltanto che la mina è più avanti, e più grossa, e sta aspettando il cliente.

Una volta mi è capitato davvero: compilato, avviato, perfetto. Ho festeggiato con un caffè della macchinetta. Tre mesi dopo ho scoperto che il programma funzionava perché non stava facendo niente. Saltava tutta la parte importante con grande eleganza. Da allora, quando qualcosa va bene al primo colpo, lo tratto come un ospite troppo gentile: con affetto, e contando le posate.

LEGGE DI HOFSTADTER



Ci vuole sempre più tempo del previsto, anche quando nella previsione hai già tenuto conto della Legge di Hofstadter.

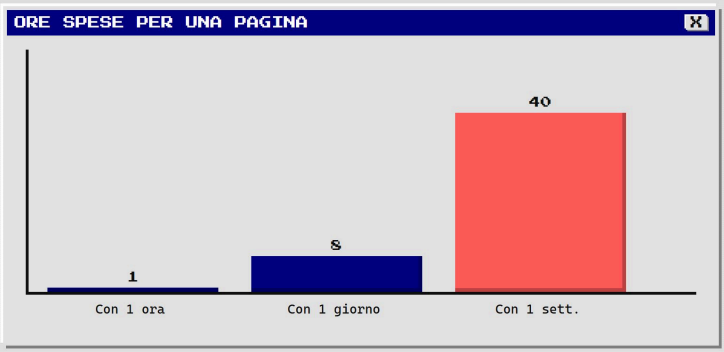
— Douglas Hofstadter, *Gödel, Escher, Bach*, 1979

È una legge ricorsiva, e quindi molto amata dagli informatici. Stimi due giorni. Sai che sbagli sempre, quindi scrivi quattro. Ma la legge sa che lo sai, e ne servono otto.

È la barra di avanzamento della vita: «Tempo rimanente: 2 minuti». Poi 47 minuti. Poi 3 secondi. Poi «Calcolo del tempo rimanente in corso», che è il modo elegante di dire «non ne ho idea».

Ho chiesto a un'intelligenza artificiale di stimare il tempo del mio progetto applicando la legge di Hofstadter. Ha risposto: «Ci vorrà più del previsto». Ho chiesto quanto più. Sta ancora elaborando. Le ho fissato una scadenza di dieci minuti, tenendo conto della legge. Torno martedì.

LEGGE DI PARKINSON



La pagina è identica. Quella da una settimana ha il font cambiato quattordici volte.

Il lavoro si espande fino a occupare tutto il tempo disponibile per completarlo. In informatica si aggiunge un corollario: i dati si espandono fino a riempire tutto lo spazio disponibile.

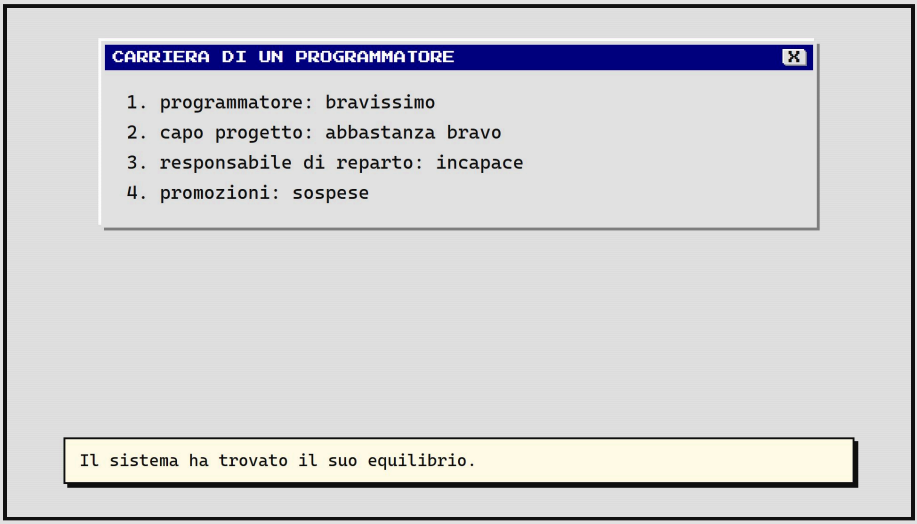
— Cyril Northcote Parkinson, *The Economist, novembre 1955 (corollario sui dati: anonimo, attribuzione incerta)*

Se hai una settimana per scrivere una pagina, ci metterai una settimana. Se hai un'ora, ci metterai un'ora. La pagina sarà identica. Nella versione da una settimana avrà solo un font cambiato quattordici volte.

Nato come satira sulla burocrazia britannica, il principio è diventato il manuale non ufficiale dei progetti informatici. Le scadenze generose non producono software migliore: producono riunioni migliori, cioè più lunghe. E vale per le cose oltre che per le persone. Compri un disco da quattro terabyte e in sei mesi è pieno, come un garage nuovo.

Nel mio ufficio abbiamo provato a sconfiggere la legge dando scadenze finte, più corte di quelle vere. Ha funzionato per un mese. Poi tutti hanno cominciato a calcolare la scadenza vera, e adesso abbiamo un foglio di calcolo che la stima. Ci lavoriamo da tre settimane.

PRINCIPIO DI PETER



In una gerarchia ogni persona tende a essere promossa fino a raggiungere il proprio livello di incompetenza, e lì si ferma.

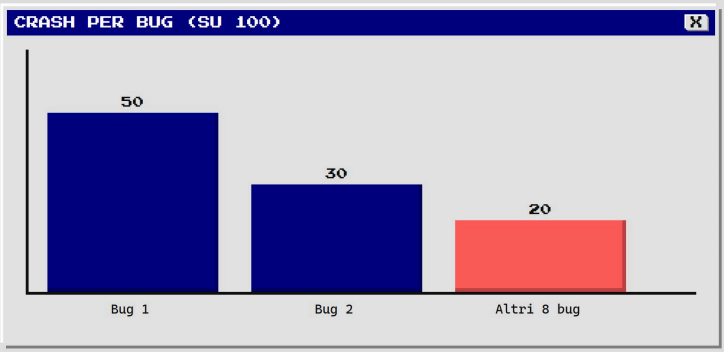
— Laurence J. Peter (con Raymond Hull), *«The Peter Principle»*, *1969*

C'era una volta un programmatore bravissimo. Così bravo che lo promossero a capo progetto. Lì era meno bravo, ma abbastanza da essere promosso a responsabile di reparto. E lì, finalmente, era del tutto incapace. Così smisero di promuoverlo. Il sistema aveva trovato il suo equilibrio.

Il principio di Peter spiega perché, in molte aziende, chi scriveva il codice migliore oggi passa le giornate a fare tabelle sulle tabelle. Saper fare una cosa non significa saper guidare chi la fa. Ma la promozione premia la prima abilità per assegnare un lavoro che richiede la seconda.

Io ho evitato il problema in modo geniale: mi sono messo in proprio. Così sono contemporaneamente programmatore, sistemista, commerciale e direttore generale. Ho raggiunto il livello di incompetenza in quattro ruoli insieme. È un'ottimizzazione.

PRINCIPIO DI PARETO (80/20)



Due bug su dieci fanno otto crash su dieci. Il venti per cento lavora tantissimo.

In molti fenomeni circa l'80% degli effetti deriva dal 20% delle cause: per esempio l'80% dei problemi viene da una piccola parte dei bug o dei clienti.

— Vilfredo Pareto, generalizzato da Joseph M. Juran, *Osservazione di Pareto sulla distribuzione della ricchezza (1896); nome e generalizzazione di Juran (anni '40–'50)*

Un economista italiano notò che pochi possedevano gran parte della terra. Decenni dopo un esperto di qualità generalizzò: pochi fattori causano la maggior parte degli effetti. Da allora l'ottanta e il venti sono la coppia più famosa della consulenza aziendale.

In informatica funziona in modo quasi imbarazzante. L'ottanta per cento dei crash viene da un pugno di bug. L'ottanta per cento delle chiamate all'assistenza arriva dallo stesso venti per cento di utenti, e tu sai benissimo come si chiamano. L'ottanta per cento del valore di un programma sta in poche funzioni, che sono proprio quelle nascoste nel terzo sottomenù.

Ho applicato Pareto alla mia scrivania. Il venti per cento dei cavi produce l'ottanta per cento dei grovigli. Il problema è che non so quale venti. Li ho staccati tutti. Ora sono a posto, ma niente si accende.

RASOIO DI HANLON



Non attribuire a cattiveria ciò che si spiega a sufficienza con la distrazione o l'incompetenza.

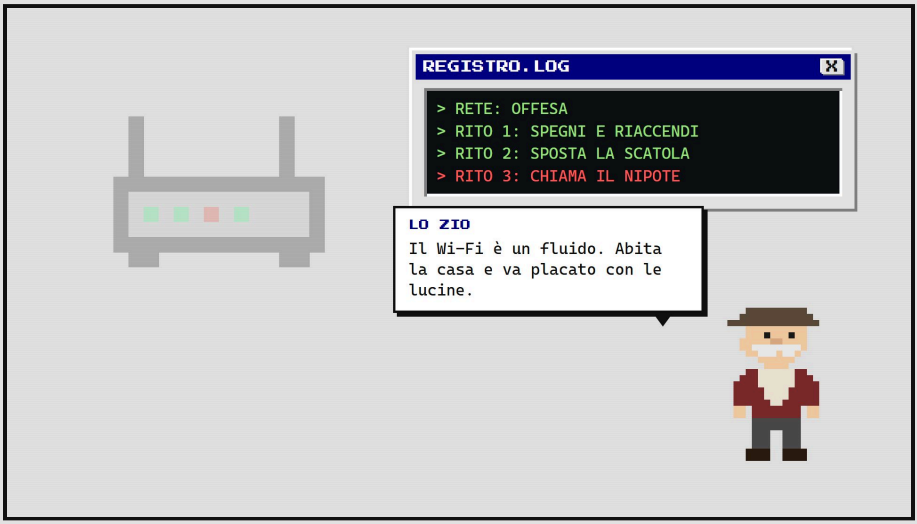
— Robert J. Hanlon, *Arthur Bloch, «Murphy's Law Book Two», 1980 (concetti simili in testi molto precedenti)*

Il server è giù. Il primo pensiero è: attacco informatico, spie straniere, un concorrente geloso. Poi si scopre che l'addetta alle pulizie ha staccato la spina per attaccare l'aspirapolvere. Nessun complotto. Solo una presa libera in meno.

Il rasoio di Hanlon è il miglior amico del tecnico di assistenza. Quando un utente cancella la cartella condivisa, non sta sabotando l'azienda. Ha solo cliccato con decisione su qualcosa che non capiva. Che è peggio, ma almeno non è personale. Tenerlo a mente salva i rapporti umani e la pressione arteriosa.

Un consiglio: il rasoio vale anche al contrario. Quando sbagli tu, sperane che gli altri lo conoscano. Io lo cito sempre ai clienti prima di dire cosa ho combinato. Funziona come una specie di antivirus diplomatico.

TERZA LEGGE DI CLARKE



Qualunque tecnologia sufficientemente avanzata è indistinguibile dalla magia.

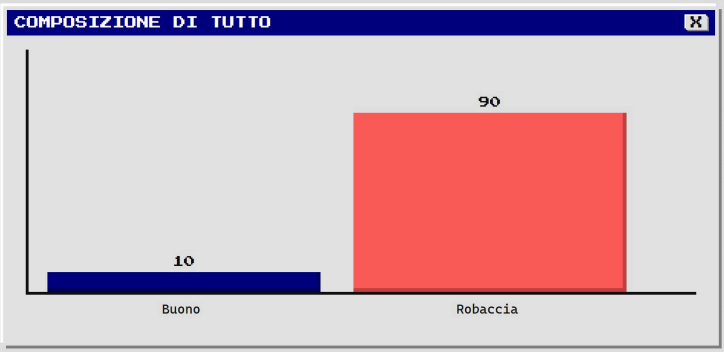
— Arthur C. Clarke, *«Profiles of the Future», edizione rivista del 1973 (già in una lettera a Science del 1968)*

Uno scrittore di fantascienza osservò che una tecnologia abbastanza avanzata, per chi non sa come funziona, sembra un incantesimo. Aveva in mente civiltà aliene. Non aveva visto mia zia davanti al Wi-Fi.

Per lei la rete è un fluido invisibile che abita la casa e si offende facilmente. Va placato spegnendo e riaccendendo una scatola con le lucine. Se non basta, si sposta la scatola. Se non basta ancora, si chiama il nipote, che recita formule incomprensibili e la scatola guarisce. Non è tecnologia. È una liturgia.

Il lato oscuro della legge è che vale anche per chi la tecnologia la vende: più sembra magia, meno la gente fa domande. Io invece spiego sempre tutto. I clienti mi ascoltano con rispetto, annuiscono, e poi riavviano comunque. La magia vince.

LEGGE DI STURGEON



Vale per la fantascienza, per le app, per i plugin. Probabilmente anche per questo grafico.

Il novanta per cento di qualsiasi cosa è robaccia: vale per la fantascienza, ma anche per libri, film, programmi e prodotti di ogni genere.

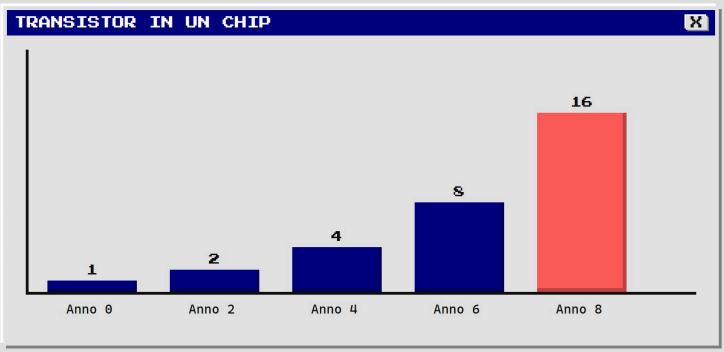
— Theodore Sturgeon, *Venture Science Fiction, marzo 1958 (detta anche «rivelazione di Sturgeon»)*

Uno scrittore di fantascienza si sentiva dire che il novanta per cento della fantascienza era spazzatura. Lui rispose: certo, perché il novanta per cento di tutto è spazzatura. Fine della discussione, applausi, e una legge che vale ovunque.

Nell'informatica è quasi un'unità di misura. Il novanta per cento delle app nel negozio è inutile. Il novanta per cento delle librerie su internet non è mantenuto. Il novanta per cento dei plugin per Joomla che ho installato nel 2011 è ancora lì, e nessuno ricorda a cosa servisse.

Il bello della legge è che non è pessimista: dice che il dieci per cento buono esiste. Il problema è trovarlo. Io ci ho costruito una carriera. Il mio lavoro, in sostanza, è setacciare la sabbia. Ogni tanto trovo una pepita. Più spesso un driver.

LEGGE DI MOORE



Nel 1965 bastarono quattro puntini su un grafico. Andò avanti così per mezzo secolo.

Il numero di transistor che si riescono a stipare in un chip raddoppia a intervalli regolari: prima ogni anno, poi, corretto il tiro, circa ogni due anni.

— Gordon Moore, *Electronics, 1965 (previsione rivista nel 1975)*

Nel 1965 un ingegnere guardò quattro puntini su un grafico e disse: andrà avanti così. Aveva ragione per mezzo secolo. Io guardo quattro puntini sul calendario e dico che finirò il sito entro venerdì. Abbiamo metodi simili, risultati diversi.

In pratica significa che ogni due anni il computer diventa il doppio più potente. E ogni due anni il programma che usi diventa il doppio più pesante, così il risultato netto è che Word si apre sempre in sei secondi. È un equilibrio cosmico. Un pareggio eterno tra ingegneri dell'hardware e ingegneri del software, e l'arbitro è la clessidra di Windows.

Oggi i transistor sono così piccoli che stanno per toccare gli atomi. Mio cugino dice che a quel punto basterà stringere di più. Ha anche un cacciavite apposta.

IL CONSIGLIO DEL SISTEMISTA

Murphy non si sconfigge, si aggira. Prima di qualunque intervento serio, che sia un aggiornamento, un trasloco di server o una modifica al sito, seguo quattro regole scritte su un foglio vero, attaccato al monitor.

- **Una copia prima.** Backup completo e verificato, fatto adesso e non «l'altro giorno».
- **Una strada per tornare indietro.** Se non so come annullare la modifica, non la faccio.
- **Una cosa alla volta.** Cambiarne tre insieme significa non sapere quale delle tre ha rotto la quarta.
- **Un appunto per ogni passo.** Data, cosa ho fatto, cosa è successo. Tra sei mesi sarà l'unico documento attendibile.

Non serve un software costoso: basta un quaderno. Il quaderno non si aggiorna da solo e non chiede la licenza. Murphy colpirà lo stesso, ma troverà la porta di servizio già aperta per uscire.

CAPITOLO 2

FERRAGLIA

Ferraglia: stampanti, cavi e dischi che ascoltano

Il software fa paura perché non si vede. L'hardware fa paura perché si vede benissimo, e ti guarda. La stampante con la spia arancione, il cavo che non arriva, la vite che rotola sotto la scheda madre con un rumore che somiglia a una risata.

In questo capitolo ci sono gli oggetti. Quelli che stanno sulla scrivania, sotto la scrivania e, nei casi peggiori, dietro la scrivania, dove la polvere ha sviluppato un microclima. Sono le leggi più democratiche del libro: non serve saper programmare per averle subite. Basta aver dovuto stampare un contratto alle nove meno cinque.

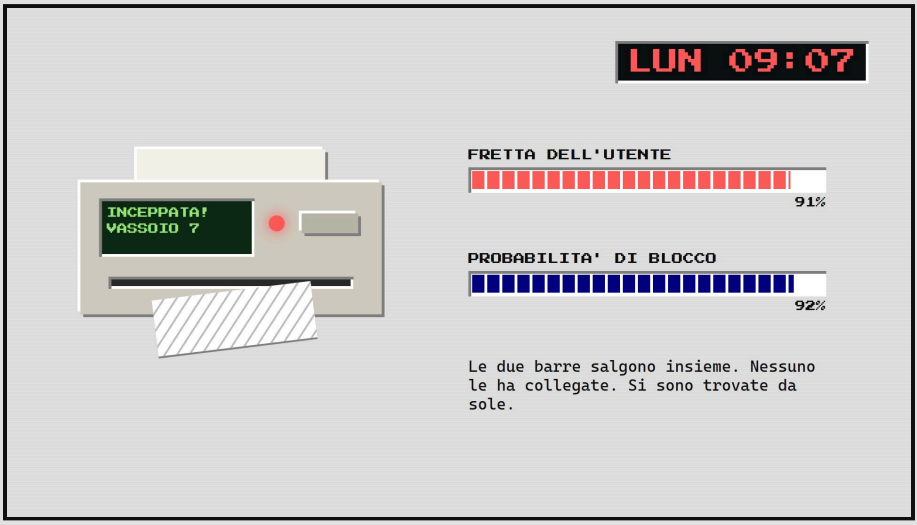
Nella seconda parte si scende nei dischi, dove abitano i dati e le loro paure. Il backup che non c'è, quello che c'è ma non si ripristina, il file che vive sempre sull'altro computer. Qui l'ironia è più sottile, perché il dolore è più vero. Una stampante rotta si sostituisce. Le foto della comunione no.

Leggetelo con un cacciavite a portata di mano. Non servirà, ma dà sicurezza.

CONTENUTO.DIR

01 Legge della stampante impaziente
02 Legge del toner veggente
03 Legge dell'inceppamento selettivo
04 Legge della stampante di rete fantasma
05 Legge del cavo corto
06 Legge del cavo mancante
07 Legge della chiavetta USB quantistica
08 Legge della vite fuggitiva
09 Legge della garanzia scaduta
10 Legge della batteria ottimista
11 Legge del proiettore ostile
12 Legge del driver introvabile
13 Legge del backup che non hai fatto
14 Legge del disco sensitivo
15 Legge del ripristino mai provato
16 Legge del salvataggio mancato
17 Legge del file nell'altro computer
18 Legge di Segal
19 Legge di Kryder

LEGGE DELLA STAMPANTE IMPAZIENTE



La probabilità che la stampante si blocchi è direttamente proporzionale alla fretta di chi deve stampare.

— Anonimo, *folklore informatico*

La stampante non ha occhi, non ha orecchie e non ha un collegamento con l'orologio di casa. Eppure sa. Il lunedì alle nove, con il treno che parte alle nove e dieci, la stampante percepisce il battito cardiaco dell'utente come uno squalo percepisce una goccia di sangue.

Ho conosciuto un ufficio che per anni ha stampato solo con calma, cantando sottovoce, per non farsi scoprire. Funzionava. Poi un giorno un impiegato ha guardato l'orologio. Quarantadue minuti di “riscaldamento in corso”.

Consiglio da tecnico: i documenti importanti si stampano la sera prima. Non perché la stampante dorma, ma perché la mattina dopo avete un piano B e lei lo sa.

LEGGE DEL TONER VEGGENTE



Il toner e la carta finiscono sempre a metà del documento più importante della settimana, mai durante le prove.

— Anonimo, *folklore informatico*

Il toner è un liquido profetico. Per tre mesi stampa volantini della sagra, ricette e foto del gatto senza battere ciglio. Poi arriva il contratto da firmare, e lui si ritira con dignità alla pagina quattro di sette, lasciando una riga grigia come testamento.

Mio cugino scuoteva la cartuccia ogni mattina, per abitudine, come si fa col termometro. Diceva che così il toner si spaventava e restava. Aveva un cassetto pieno di cartucce scosse. Non ne ha mai finita una prima di un rogito.

Nel frattempo, nella sala riunioni, qualcuno chiede se si può stampare “solo in nero”. La risposta è no: il nero è proprio quello finito.

Mini-consiglio: tenete sempre una cartuccia di scorta. Costa meno della figuraccia.

LEGGE DELL'INCEPPAMENTO SELETTIVO



La carta si inceppa solo sull'unico foglio che non puoi ristampare: la carta intestata, il modulo prestampato, l'ultimo foglio buono.

— Anonimo, *folklore informatico*

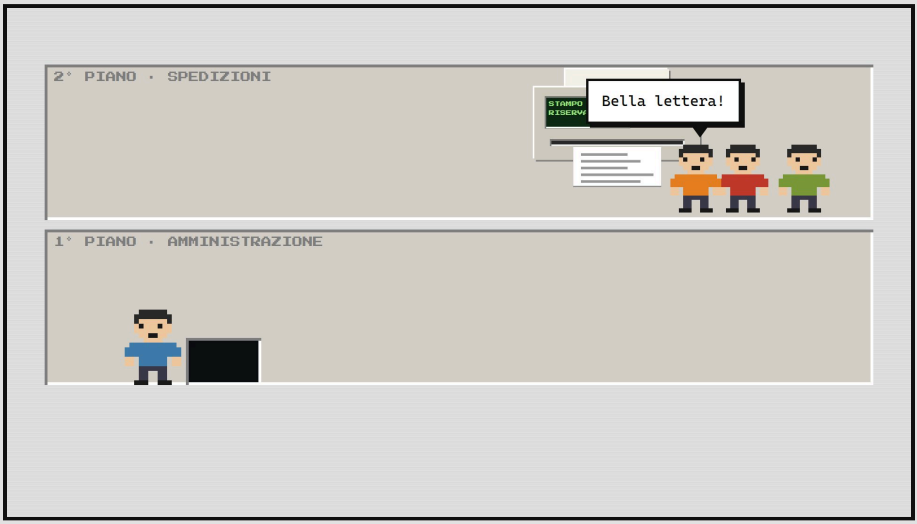
Le stampanti hanno un talento da sommelier. Davanti a una risma da due euro sono docili come agnelli. Ma appena inserisci l'ultimo foglio di carta intestata, quello con il logo in rilievo, il rullo si illumina di una gioia antica e lo trasforma in una fisarmonica.

Ho visto colleghi aprire sportelli che non sapevo esistessero. Sportello A, sportello B, sportello “non aprire durante il funzionamento”. Dentro, il foglio era piegato in una forma che la geometria euclidea non prevede.

La stampante, intanto, mostrava sul display una piccola icona animata di una mano che estrae la carta. Con tutta calma. Come chi spiega a un bambino che il pallone è finito nel giardino del vicino.

Consiglio vero: prima di usare il foglio prezioso, fate una prova su carta comune. Sempre.

LEGGE DELLA STAMPANTE DI RETE FANTASMA



Il documento riservato che mandi in stampa finisce sempre sulla stampante dell'altro piano, davanti alla persona sbagliata.

— Anonimo, *folklore informatico*

In ogni azienda esiste una stampante che nessuno ha installato. Compare nell'elenco con un nome tipo “HP_SECONDO_PIANO_VECCHIA”. Nessuno sa dove sia. Ma è sempre lei la predefinita quando stampi la lettera di dimissioni.

Una volta ho seguito il percorso di un documento per tutto l'edificio. Era partito dall'amministrazione, era passato per il magazzino ed era stato raccolto, letto e commentato dal reparto spedizioni. Tre persone mi hanno detto “bella lettera”. Non era la mia.

La stampante fantasma non sbaglia. Sceglie. Ha un archivio segreto di tutto ciò che non doveva vedere, e un giorno scriverà le sue memorie.

Consiglio da sistemista: controllate quale stampante è impostata come predefinita prima di premere “Stampa”. Due secondi che salvano carriere.

LEGGE DEL CAVO CORTO



Qualunque cavo è sempre dieci centimetri più corto della distanza che deve coprire.

— Anonimo, *folklore informatico*

È una costante universale, come la velocità della luce. Se la presa è a un metro, il cavo è da novanta. Se compri quello da due metri, la scrivania si sposta. Nessuno l'ha vista muoversi, ma il metro non mente.

Ho un amico che ha provato a fregare la legge comprando un cavo da dieci metri per collegare un monitor distante mezzo metro. Il cavo è arrivato. Lui ha anche cambiato casa, e la nuova presa era a dieci metri e dieci centimetri. Coincidenze, dicono gli scettici.

Il cavo corto è anche un maestro di vita: insegna la pazienza, l'umiltà, e soprattutto a infilarsi sotto la scrivania con la torcia in bocca.

Consiglio da tecnico: misurate, aggiungete mezzo metro, e comprate quello. Il cavo lungo si arrotola; quello corto non si allunga.

LEGGE DEL CAVO MANCANTE

INVENTARIO DEL CASSETTO

- 7 caricatori di telefoni mai posseduti
- prolunga del modem del secolo scorso
- cavo con spina tonda (destinazione ignota)
- 3 cavi identici, tutti diversi

Cavi che servono adesso: 0

Cinquanta cavi. Di notte si riproducono, ma solo tra quelli inutili.

In un cassetto con cinquanta cavi diversi non c'è mai quello che ti serve adesso.

— Anonimo, *folklore informatico*

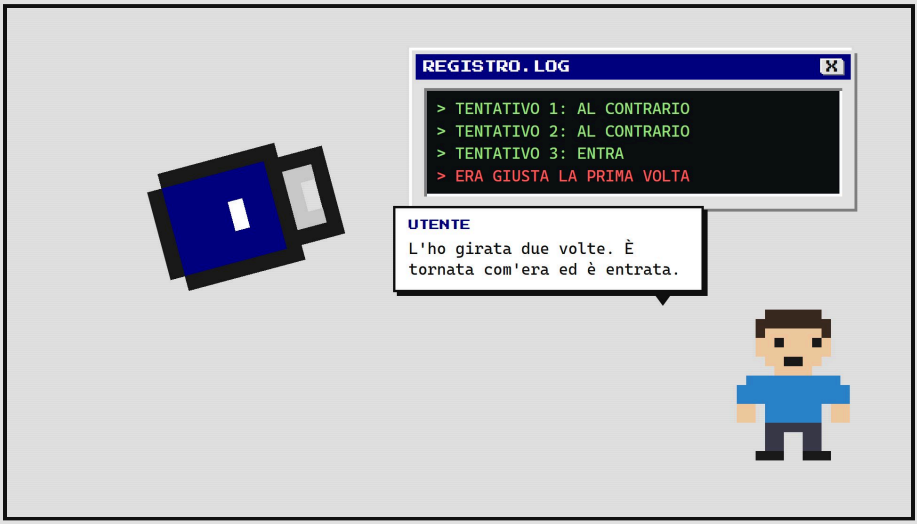
Ogni casa ha il Cassetto. Un ecosistema di cavi intrecciati che si riproducono di notte: caricatori di telefoni mai posseduti, prolunghe di modem del secolo scorso, un cavo misterioso con una spina tonda che non entra in niente.

Cinquanta cavi. Zero col connettore giusto. Statisticamente impossibile, eppure succede tutte le volte. Il cavo che serve esiste, certo: è nel cassetto del vicino, che a sua volta cerca il tuo.

Ho proposto al condominio una borsa dei cavi, con scambi il sabato mattina. È fallita perché nessuno voleva cedere il proprio cavo inutile. “Non si sa mai”, dicevano. Il motto ufficiale di ogni cassetto.

Consiglio vero: etichettate i cavi con un pezzo di nastro e buttate senza pietà quelli che non riconoscete da due anni.

LEGGE DELLA CHIAVETTA USB QUANTISTICA



La spina USB entra sempre al terzo tentativo: la prima volta è al contrario, la seconda pure, la terza era giusta dall'inizio.

— Anonimo, *folklore informatico*

Il connettore USB classico ha due lati. Matematicamente, al primo tentativo hai il cinquanta per cento di probabilità di azzeccarlo. Nella pratica, la probabilità è zero. Il connettore esiste in uno stato di sovrapposizione: è sbagliato in entrambi i versi finché non lo giri due volte.

Ho calcolato che nella mia carriera ho girato chiavette per circa tre giorni consecutivi della mia vita. Li considero ferie non godute.

Poi è arrivato il connettore reversibile, e l'umanità ha festeggiato. Per un pomeriggio. Poi ci siamo accorti che i cavi reversibili non caricano tutti alla stessa velocità, non trasmettono tutti i dati e non accendono tutti il monitor. Il caos si è solo spostato all'interno del cavo.

Consiglio: il logo USB sul connettore, di solito, va verso l'alto. Di solito.

LEGGE DELLA VITE FUGGITIVA



La vite che ti scivola dalle dita mentre apri il computer cade sempre nel punto più irraggiungibile del case.

— Anonimo, *folklore informatico*

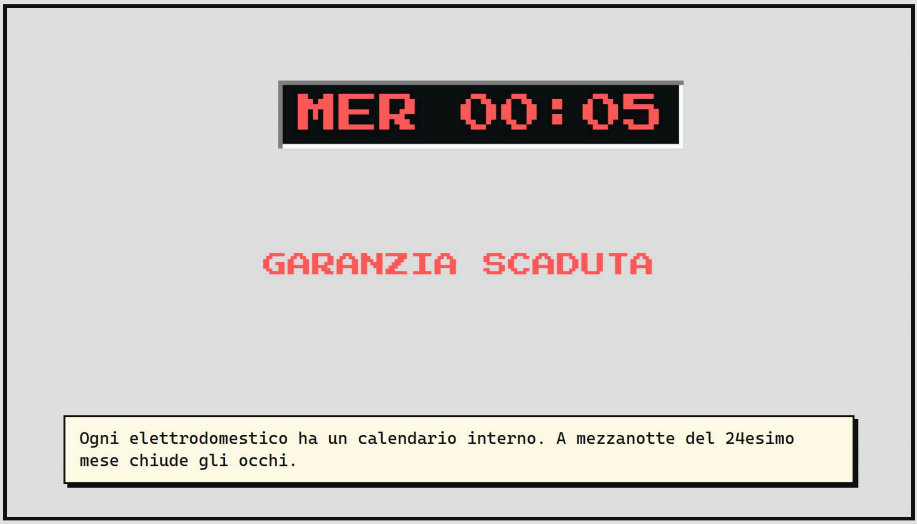
Dentro un computer ci sono migliaia di posti comodi dove una vite potrebbe atterrare. Lei sceglie sempre l'unico sotto la scheda madre, dietro l'alimentatore, in una fessura progettata da qualcuno che odiava le dita umane.

Il tecnico allora esegue l'antica danza del case: lo inclina, lo scuote, lo capovolge, ascolta. La vite rotola con un suono metallico, come una risata. Poi tace. Nessuno la vedrà mai più, ma il computer ora ha un rumore nuovo quando lo sposti.

Nel mio laboratorio teniamo un barattolo di viti orfane recuperate negli anni. Nessuna è mai servita a niente. Le chiamiamo “il fondo pensione”.

Consiglio da tecnico: usate un cacciavite magnetico e una ciotolina per le viti. E non lavorate mai sopra la moquette.

LEGGE DELLA GARANZIA SCADUTA



Un dispositivo si guasta il giorno dopo la scadenza della garanzia, salvo i casi in cui si guasta il giorno prima e hai perso lo scontrino.

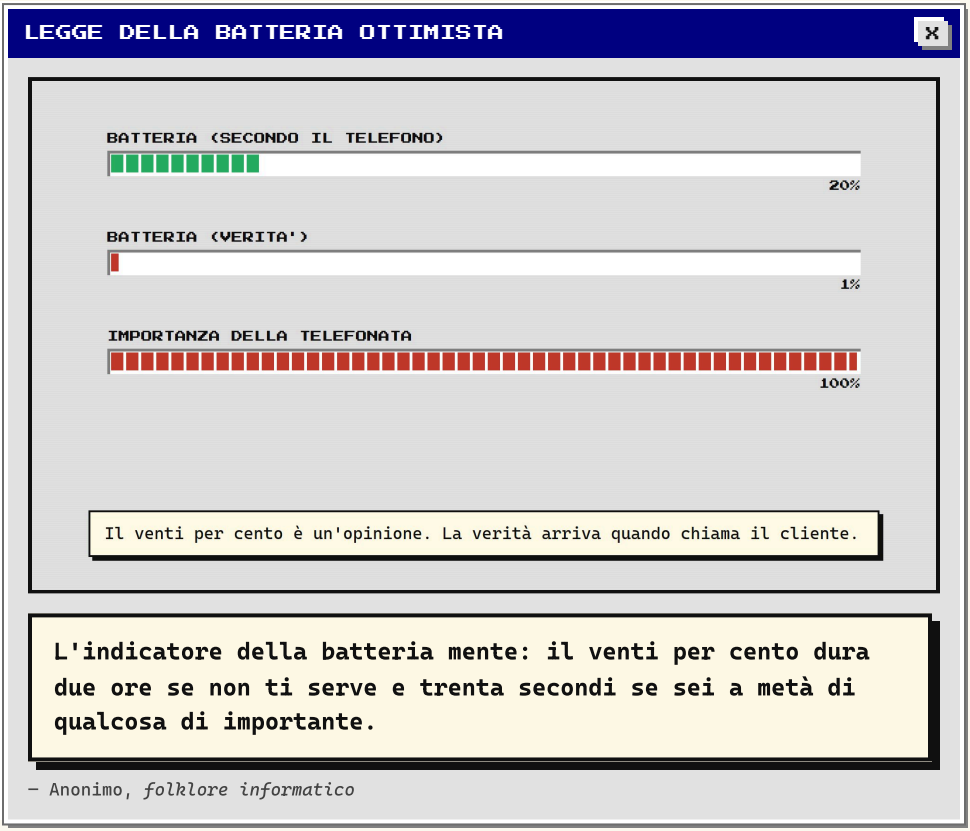
— Anonimo, *folklore informatico*

Dentro ogni elettrodomestico moderno c'è un calendario. Non è scritto nel manuale, ma c'è. Conta i giorni con la pazienza di un monaco, e alla mezzanotte del ventiquattresimo mese chiude gli occhi in pace.

Una mia cliente aveva conservato tutti gli scontrini dal 1998 in una scatola da scarpe, catalogati per colore. Il giorno in cui la stampante si è guastata in garanzia, la scatola era in cantina. La cantina era allagata. Lo scontrino era diventato una piccola nuvola bianca.

La garanzia, insomma, esiste: è come l'unicorno del servizio clienti. Tutti ne parlano, qualcuno giura di averla vista.

Consiglio da tecnico: fotografate lo scontrino appena comprate e salvatelo nel cloud. La carta termica sbiadisce, le foto no.



L'indicatore della batteria mente: il venti per cento dura due ore se non ti serve e trenta secondi se sei a metà di qualcosa di importante.

— Anonimo, *folklore informatico*

La percentuale della batteria non è un dato. È un'opinione. Un sentimento. Il telefono al venti per cento ti dice “tranquillo, ce la facciamo”, con il tono di chi guida in riserva in autostrada.

Poi apri la mappa per trovare il cliente, e la percentuale scende come un ascensore senza cavi: diciotto, undici, quattro, nero. Il caricatore è a casa, in carica anche lui, per solidarietà.

Mio zio sosteneva che le batterie si ricaricano da sole se le lasci riposare nel cassetto. Ha ragione per circa un minuto. È il minuto più ottimista della sua giornata.

Consiglio vero: le batterie al litio si usurano, e con l'età l'indicatore diventa impreciso. Se il dispositivo si spegne a percentuali alte, è ora di sostituirla.

LEGGE DEL PROIETTORE OSTILE



Il proiettore della sala riunioni ha un ingresso diverso da quello del tuo portatile, e l'adattatore giusto è l'unico che nessuno ha portato.

— Anonimo, *folklore informatico*

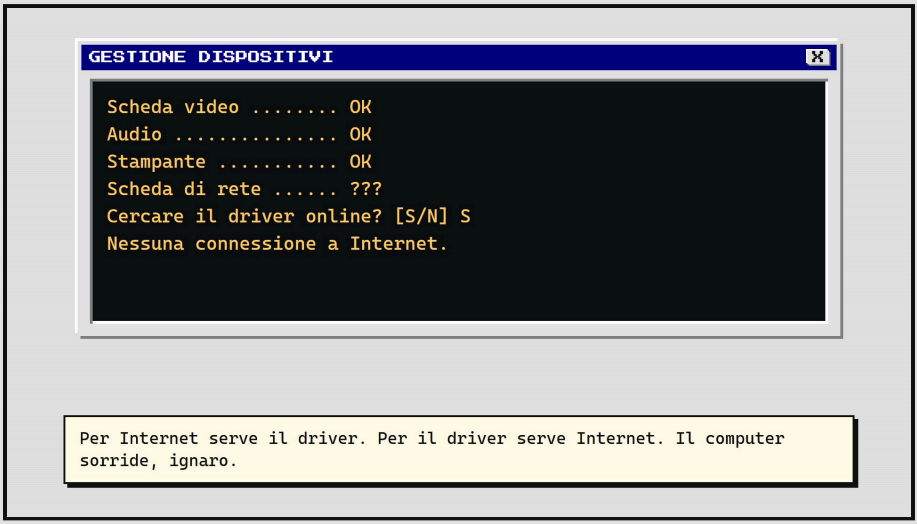
Ogni sala riunioni ha un proiettore appeso al soffitto come un pipistrello elettronico. Ha un solo ingresso, di un tipo che il tuo portatile ha abbandonato tre generazioni fa. Sul tavolo, una scatola di adattatori: dieci, tutti per altri computer, alcuni per oggetti che non sono computer.

Intanto il pubblico aspetta. Qualcuno suggerisce di “mandare la presentazione per mail e guardarla ognuno sul telefono”. È il momento in cui la civiltà riparte dalla carta.

Quando finalmente l'immagine compare, è viola, è tagliata a destra e lo schermo del portatile si spegne. Hai vinto, ma non sai come.

Consiglio da tecnico: portate sempre il vostro adattatore e una copia della presentazione in PDF su chiavetta. E arrivate quindici minuti prima.

LEGGE DEL DRIVER INTROVABILE



Dopo aver reinstallato il sistema, l'unico driver che manca è quello della scheda di rete, che servirebbe per scaricare gli altri driver.

— Anonimo, *folklore informatico*

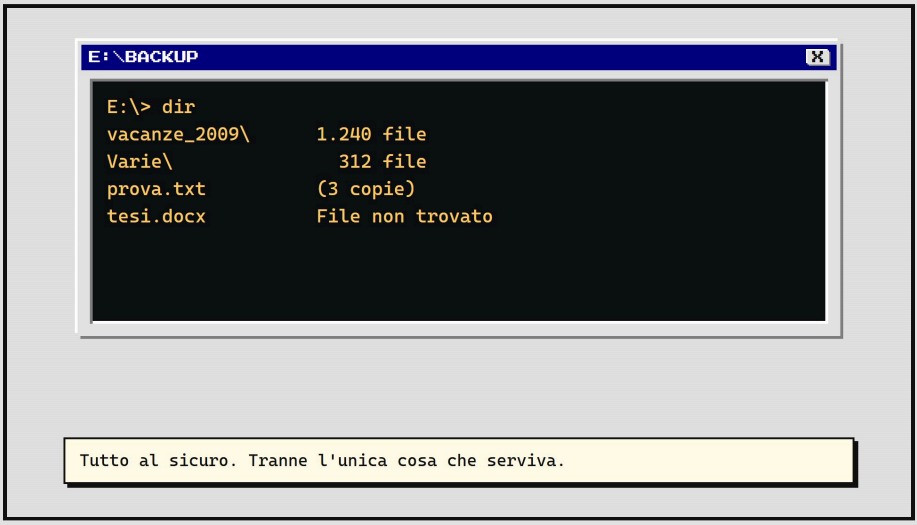
È il paradosso perfetto, degno di un racconto di fantascienza. Per avere Internet ti serve il driver. Per avere il driver ti serve Internet. Il computer appena formattato ti guarda con innocenza, come un neonato che non sa ancora parlare con il router.

Il tecnico allora parte per la Grande Spedizione: un altro computer, una chiavetta, un sito del produttore che ha cambiato struttura sei volte. Il driver si chiama tipo “pacchetto_finale_v2_nuovo_DEFINITIVO.zip” e pesa più del sistema operativo.

Nel frattempo, nella stanza accanto, il cliente chiede se “già che ci sei” puoi recuperare anche le foto del matrimonio.

Consiglio vero: prima di formattare, scaricate su chiavetta i driver di rete del vostro modello. È la cassetta degli attrezzi del giorno dopo.

LEGGE DEL BACKUP CHE NON HAI FATTO



Di tutti i backup possibili, quello che ti serve è sempre l'unico che non hai fatto.

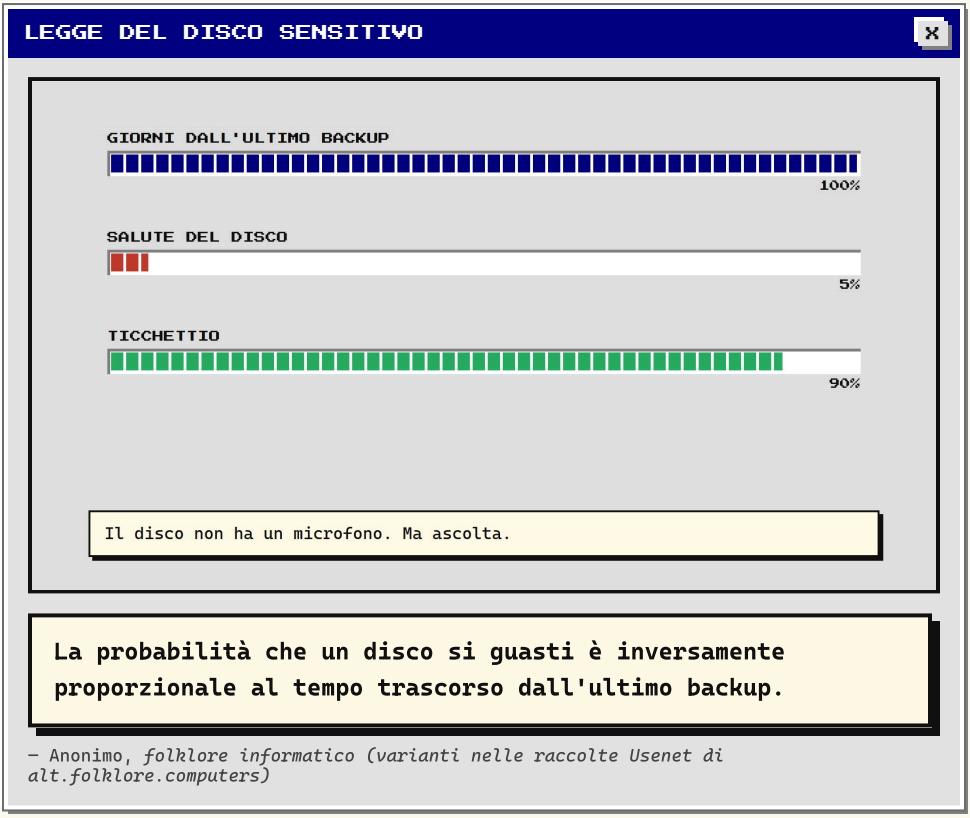
— Anonimo, *folklore informatico*

Hai il backup delle foto delle vacanze del 2009. Hai il backup della cartella “Varie”. Hai tre backup identici di un file chiamato “prova.txt”. Poi il disco muore, e l'unico file che ti serve davvero è la tesi, che stava sul desktop “solo per oggi”.

Il backup è una forma di superstizione al contrario: se lo fai, non succede niente e pensi di aver perso tempo. Se non lo fai, l'universo organizza una dimostrazione pratica.

Ho un cliente che faceva il backup su una chiavetta, che teneva attaccata al computer, che stava sotto la finestra, sotto la quale è passato un temporale. Ora ha due cose bagnate al posto di una.

Consiglio vero: regola del 3-2-1. Tre copie, su due supporti diversi, una delle quali fuori casa o nel cloud.



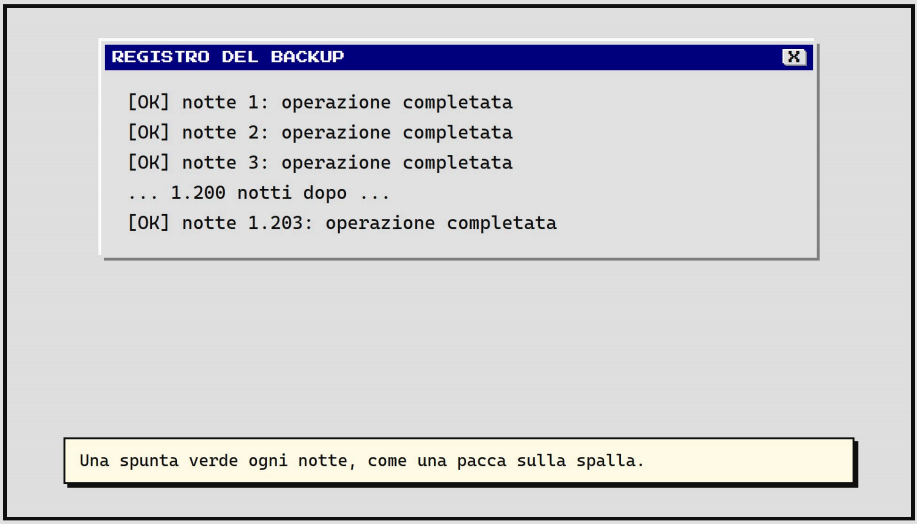
Il disco rigido ascolta. Non ha microfono, ma ascolta. Il giorno dopo il backup completo gira felice, silenzioso, eterno. Dopo sei mesi senza copie comincia a fare un piccolo ticchettio, come un orologio da film di suspense.

I vecchi tecnici dicevano che il ticchettio fosse “la testina che saluta”. In realtà era la testina che salutava per sempre.

Esiste un solo modo per ingannare il disco: fare il backup senza che lui lo sappia. Con discrezione, di notte, automaticamente. Il disco scopre la copia solo dopo, e a quel punto gli passa la voglia di rompersi. Non è scienza. Ma i risultati sono ottimi.

Consiglio da sistemista: programmate un backup automatico. Quello che dipende dalla vostra memoria non è un backup, è una speranza.

LEGGE DEL RIPRISTINO MAI PROVATO



Un backup funziona sempre, fino al giorno in cui provi a ripristinarlo.

— Anonimo, *folklore informatico*

Per anni il programma di backup ha mostrato una spunta verde. Ogni notte. “Operazione completata.” Una spunta rassicurante come una pacca sulla spalla.

Poi arriva il giorno del bisogno. Apri l'archivio e scopri che contiene una sola cartella vuota, copiata con cura religiosa per milleduecento notti consecutive. Il programma non ha mai mentito: l'operazione era davvero completata. Di cosa, nessuno l'aveva chiesto.

Il backup non verificato è come un paracadute mai aperto in negozio. Probabilmente è a posto. Probabilmente.

Consiglio vero da tecnico: almeno una volta ogni tanto provate a ripristinare un file a caso dal backup. Se lo recuperate, dormite tranquilli. Se non lo recuperate, avete appena evitato il disastro peggiore.



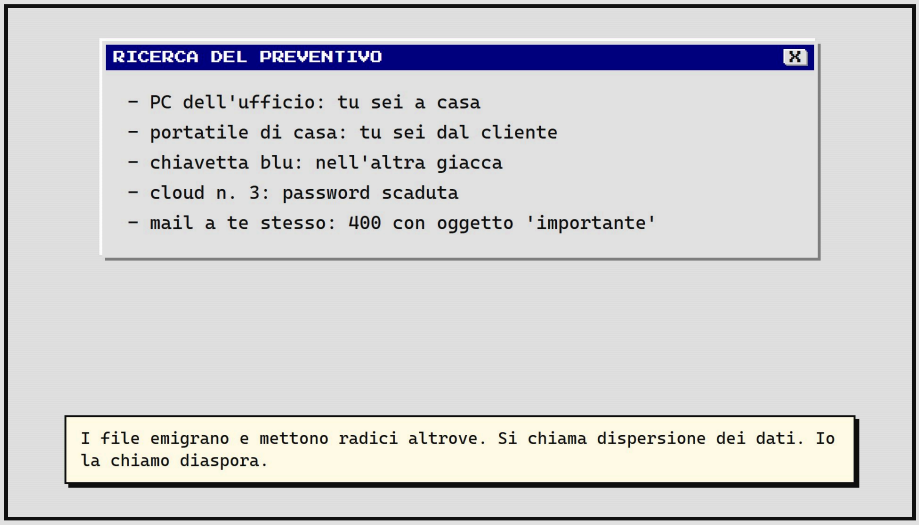
C'è un momento, durante il lavoro, in cui pensi: “Dovrei salvare.” È un pensiero nobile. Ma lo rimandi di un minuto, giusto il tempo di finire la frase. Il programma intercetta quel pensiero, lo trova presuntuoso, e si chiude.

Non si chiude male. Si chiude con eleganza. Una finestrella dice “Il programma ha smesso di funzionare” come un cameriere che annuncia che la cucina è chiusa.

Mia zia salvava ogni documento dopo ogni parola. Ci metteva un pomeriggio a scrivere un biglietto d'auguri, ma non ha mai perso niente. La chiamavamo “la Cassaforte”.

Consiglio vero: imparate la combinazione Ctrl+S e usatela come un tic. E attivate il salvataggio automatico, se il programma lo prevede.

LEGGE DEL FILE NELL'ALTRO COMPUTER



Il file che ti serve con urgenza si trova sempre sull'altro computer, sull'altra chiavetta o nell'altra casa.

— Anonimo, *folklore informatico*

Hai tre computer, due chiavette, un disco esterno e quattro servizi cloud. Il file del preventivo è in uno solo di questi posti. Ed è sempre quello a cui non puoi accedere: il computer dell'ufficio, quando sei a casa; il portatile di casa, quando sei dal cliente.

Si chiama dispersione dei dati, ma io la chiamo diaspora. I file emigrano, mettono radici altrove, cambiano nome in “preventivo_definitivo_2_copia”.

Una volta ho cercato un documento per un'ora. Era allegato a una mail che avevo mandato a me stesso con oggetto “importante”. Ce n'erano quattrocento con lo stesso oggetto.

Consiglio vero: scegliete un unico posto di lavoro principale, sincronizzato, e date ai file nomi chiari con data. Il futuro voi ringrazierà.

LEGGE DI SEGAL

LISTINO PREZZI AGGIORNATO

- gestionale: 120 euro
- Excel del commerciale: 115 euro
- PDF mandato al cliente: 99 euro

Nome di tutti e tre: quello aggiornato

Tre numeri, una sola verità. Con tre orologi si fonda una commissione.

Un uomo con un orologio sa che ore sono. Un uomo con due orologi non ne è mai sicuro.

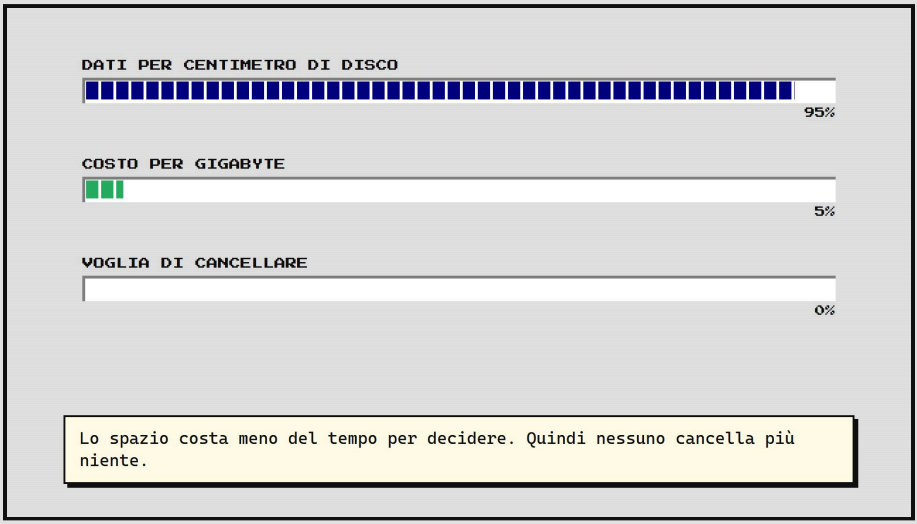
— Anonimo, *Detto apparso sul San Diego Union nel 1930; poi attribuito per errore a Lee Segall e reso celebre come «legge di Segal» nei libri di Arthur Bloch (attribuzione incerta)*

Con un solo orologio non hai dubbi: sbagli con totale serenità. Con due orologi che segnano ore diverse comincia la crisi. Quale ha ragione? Con tre si fonda una commissione.

In informatica è il dramma delle fonti multiple. Il listino prezzi sta nel gestionale, nel foglio Excel del commerciale e nel PDF mandato al cliente. Sono tre numeri diversi, e tutti si chiamano «quello aggiornato». Per questo chi progetta sistemi seri parla di fonte unica della verità: un posto solo da cui tutti leggono, e gli altri copiano.

Vale anche per i server: se gli orologi dei computer di una rete non sono sincronizzati, i log raccontano storie impossibili, con risposte che arrivano prima delle domande. Io in ufficio ho risolto. Ho tolto tutti gli orologi tranne uno. È fermo dal 2016, ma nessuno discute più.

LEGGE DI KRYDER



La densità di dati registrabili su un disco magnetico cresce più in fretta della potenza dei chip, e il costo per gigabyte crolla di conseguenza.

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

Il primo disco rigido commerciale pesava come un frigorifero e conteneva pochi megabyte. Oggi una scheda grande come un'unghia contiene tutte le foto delle vacanze, comprese le quaranta identiche del tramonto. Questa è la legge di Kryder: lo spazio per i dati diventa più fitto e più economico a ritmo vertiginoso.

La conseguenza psicologica è devastante. Nessuno cancella più niente. Perché scegliere, se lo spazio costa meno del tempo per decidere? Ho clienti con dodici copie dello stesso PDF, chiamate definitivo, definitivo2, definitivo_vero e definitivo_vero_questa_volta.

Negli ultimi anni la crescita ha rallentato, ma l'umanità ha già trovato la soluzione: accumulare comunque. Mio zio conserva i dischi rotti in una scatola. Dice che i dati sono ancora dentro, solo un po' timidi.

IL CONSIGLIO DEL SISTEMISTA

Due cose che quasi nessuno fa e che tutti dovrebbero fare. La prima: **chiedere al disco come sta**. Quasi tutti i dischi, meccanici e SSD, tengono un registro della propria salute chiamato SMART. Su Windows un programma gratuito come CrystalDiskInfo lo legge in un attimo e lo riassume con un colore: blu va bene, giallo è un avviso, rosso è un guaio. Il giallo non è un parere, è un preavviso: copiate i dati quel giorno stesso. Gli SSD, a differenza dei vecchi dischi, spesso si guastano senza ticchettii d'addio.

La seconda: **non fidarsi dei dischi in letargo**. Un disco esterno chiuso in un cassetto per anni può non ripartire, e le chiavette dimenticate perdono colpi. Riaccendete gli archivi ogni tanto, controllate che i file si aprano e ogni qualche anno copiate tutto su un supporto nuovo. I dati non invecchiano. I loro contenitori sì.

CAPITOLO 3

HELPDESK

Helpdesk: utenti, password e altri misteri

Questo è il capitolo del mio lavoro di tutti i giorni, e quindi quello che ho riletto con più commozione. Ci sono il telefono che squilla, l'utente che non ha toccato niente, il messaggio di errore chiuso alla velocità della luce, il computer che guarisce appena il tecnico varca la soglia.

Chiarisco subito una cosa: qui non si ride degli utenti. Si ride con loro. Ogni tecnico è utente di qualcos'altro, e io davanti alla caldaia di casa sono esattamente quel signore che giura di non aver toccato niente. Il rasoio di Hanlon, nel primo capitolo, lo dice meglio di me: dietro un disastro di solito non c'è malizia. C'è un clic.

Poi ci sono le password, che meritano una sezione tutta loro: troppo semplici, troppo complicate, dimenticate il giorno dopo. E il capitolo si chiude con alcune leggi serie della progettazione, quelle che spiegano perché un bottone piccolo e lontano viene mancato, perché un menù con quaranta voci paralizza e perché mezzo secondo di attesa sembra un'eternità. Leggerle aiuta a capire che molti «errori dell'utente» sono in realtà errori di chi ha disegnato lo schermo.

CONTENUTO.DIR

01 Legge del tecnico guaritore
02 Legge del non ho toccato niente
03 Legge della spina staccata
04 Legge del messaggio di errore non letto
05 Legge del guasto del venerdì sera
06 Legge del parente che se ne intende
07 Legge della password dimenticata
08 Legge della password mai abbastanza
09 Principio di Kerckhoffs
10 Leggi di Gilb sull'inaffidabilità
11 Legge di Tesler
12 Legge di Jakob
13 Legge di Fitts
14 Legge di Hick
15 Soglia di Doherty

LEGGE DEL TECNICO GUARITORE



Il problema scompare nell'istante esatto in cui arriva il tecnico, e ricompare appena se ne va.

— Anonimo, *folklore informatico*

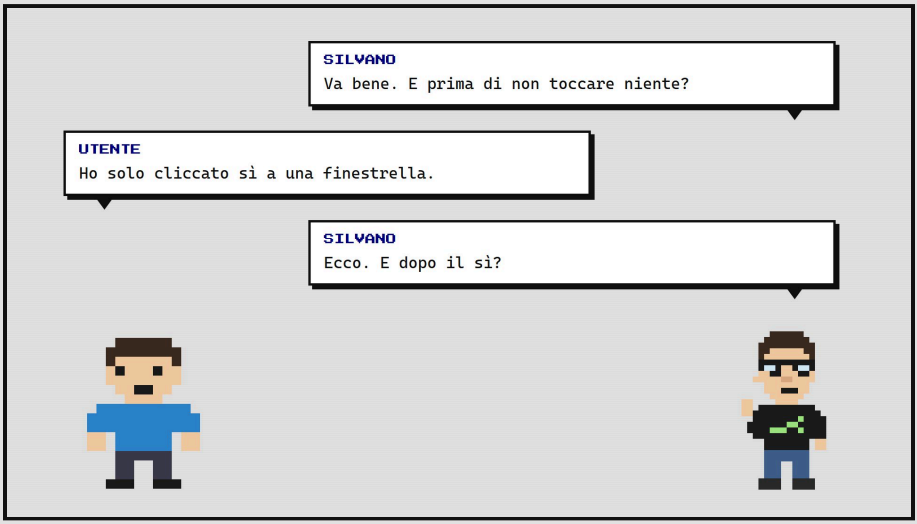
È il mio unico superpotere. Entro nella stanza, e il computer che da tre giorni mostrava schermate blu si mette sull'attenti come un soldato in caserma. “Prima non andava, giuro”, dice l'utente. Gli credo. Ho visto questa scena mille volte.

Il problema riconosce il tecnico come un cane riconosce il veterinario: fa il bravo, scodinzola, nega tutto. Poi, quando la mia macchina esce dal parcheggio, il computer ricomincia, con un po' di rabbia in più per il tempo perso.

Ho pensato di lasciare in ufficio una mia foto a grandezza naturale. Un cliente l'ha fatto con una sagoma di cartone. Dice che funziona per circa una settimana.

Consiglio da tecnico: quando compare l'errore, fate una foto allo schermo col telefono. Una foto vale più di mille “prima faceva una cosa strana”.

LEGGE DEL NON HO TOCCATO NIENTE



L'utente che dice “non ho toccato niente” ha toccato qualcosa; l'unica incognita è cosa.

— Anonimo, *folklore informatico*

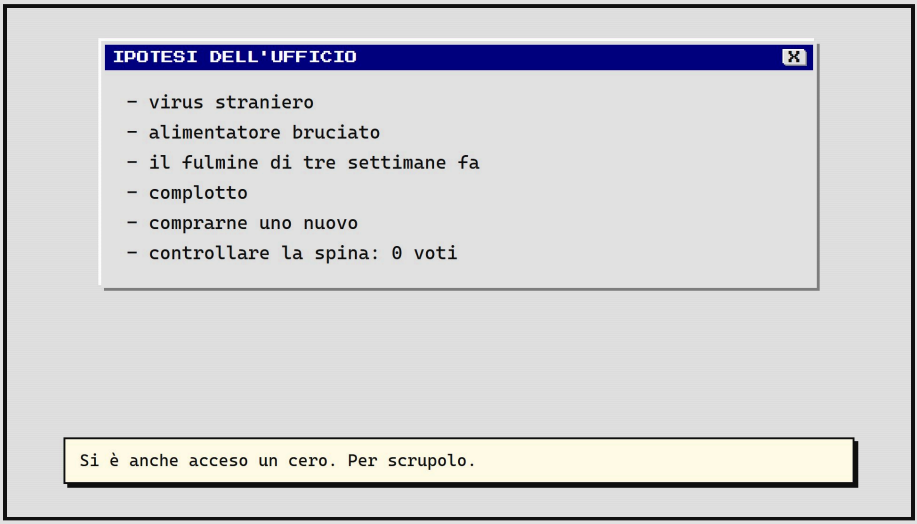
“Non ho toccato niente” è la frase più antica dell'assistenza tecnica. È stata trovata incisa sul primo computer a valvole, accanto a un'impronta digitale.

L'utente non mente. Nella sua memoria non ha toccato niente. Ha solo cliccato “sì” a una finestrella, spostato una cartella “che dava fastidio”, installato un programma gratuito per vedere meglio le foto e cambiato una piccola impostazione che “era brutta”. Ma toccare, no.

Il tecnico esperto non contesta. Annuisce con la serenità di un confessore e chiede: “Va bene. E prima di non toccare niente, cosa stava facendo?” Lì arriva la verità.

Consiglio vero: raccontare con sincerità gli ultimi passaggi fa risparmiare ore. Il tecnico non giudica, deve solo capire da dove partire.

LEGGE DELLA SPINA STACCATA



Prima di controllare se la spina è inserita si saranno già ipotizzati un virus, un guasto alla scheda madre e un complotto.

— Anonimo, *folklore informatico*

Il computer non si accende. Immediatamente l'ufficio si divide in scuole di pensiero. C'è chi parla di virus russi, chi di alimentatore bruciato, chi del fulmine di tre settimane fa che “sicuramente ha lasciato qualcosa”. Si valuta l'acquisto di un computer nuovo. Si accende un cero.

Poi arriva il tecnico, si china, e reinserisce la spina che l'impresa delle pulizie aveva staccato per collegare l'aspirapolvere. Il silenzio che segue è sacro.

La spina staccata è la causa più umile dell'informatica. Non ha codice di errore, non ha log, non ha fascino. Ma vince sempre.

Consiglio da tecnico: prima di tutto controllate le cose ovvie. Spina, ciabatta accesa, cavi del monitor ben inseriti. Non è banale: è metodo.

LEGGE DEL MESSAGGIO DI ERRORE NON LETTO



L'utente chiude il messaggio di errore prima di leggerlo, poi chiama il tecnico per sapere cosa c'era scritto.

— Anonimo, *folklore informatico*

Il messaggio di errore è un piccolo poeta incompreso. Compare con tutte le informazioni necessarie: il problema, la causa, perfino la soluzione. L'utente lo guarda con l'orrore di chi trova un ragno nella vasca, e preme “OK” a una velocità che sfida la fisica.

Poi telefona: “È uscito un messaggio.” “Cosa diceva?” “Non so, era in inglese. O in italiano. C'era una X rossa.” È tutto quello che resta, come il racconto di un avvistamento di UFO.

Il tecnico allora deve ricreare l'errore, aspettando che il poeta torni a recitare. Il poeta, offeso, non torna.

Consiglio vero: quando appare un errore, non chiudetelo subito. Leggete con calma, fate una foto o uno screenshot. Metà delle volte la risposta è già lì.

LEGGE DEL GUASTO DEL VENERDI SERA



I guasti gravi si presentano solo quando il tecnico non c'è: il venerdì sera, a Ferragosto, o il primo giorno di ferie.

— Anonimo, *folklore informatico*

Il server ha funzionato senza un lamento per undici mesi e tre settimane. Poi il tecnico prenota l'albergo al mare. Il server lo sa. Non chiedetemi come: lo sa.

Il venerdì alle diciotto e un minuto, mentre le chiavi girano nella serratura dell'ufficio, una lucina diventa arancione. Alle diciotto e due, rossa. Alle diciotto e tre, squilla il telefono del tecnico, che è già in autostrada con la famiglia e un materassino gonfiabile sul sedile.

Ho conosciuto un sistemista che non andava in ferie da sei anni per paura di questa legge. Il server non si è mai guastato. Poi è andato dal dentista un martedì mattina.

Consiglio vero: documentazione chiara, accessi annotati in un luogo sicuro e un collega di riserva. La reperibilità non è eroismo, è organizzazione.

LEGGE DEL PARENTE CHE SE NE INTENDE



Chi in famiglia “se ne intende di computer” trascorrerà ogni pranzo delle feste a configurare la stampante di uno zio.

— Anonimo, *folklore informatico*

Basta una volta. Una sola volta in cui sistemi il telefono della nonna, e vieni nominato a vita Responsabile Informatico della Famiglia. Senza stipendio, senza ferie, con reperibilità permanente nei giorni festivi.

Arrivi per il pranzo di Natale e non fai in tempo a toglierti il cappotto. “Già che ci sei, guarderesti il computer?” Il computer è in soffitta, ha dodici anni e una pellicola di polvere che fa da isolante termico.

A fine giornata hai mangiato un tortellino freddo, aggiornato quattro telefoni e spiegato sei volte che il Wi-Fi non è “Internet”. Ti pagano con un vasetto di marmellata. È ottima, ammettiamolo.

Consiglio vero: scrivete su un foglio le password del Wi-Fi e degli account principali dei parenti, e lasciatelo a loro in un posto sicuro. Ridurrete le chiamate della metà.

LEGGE DELLA PASSWORD DIMENTICATA



Una password si dimentica sempre il giorno dopo averla cambiata, perché il sistema lo imponeva.

— Anonimo, *folklore informatico*

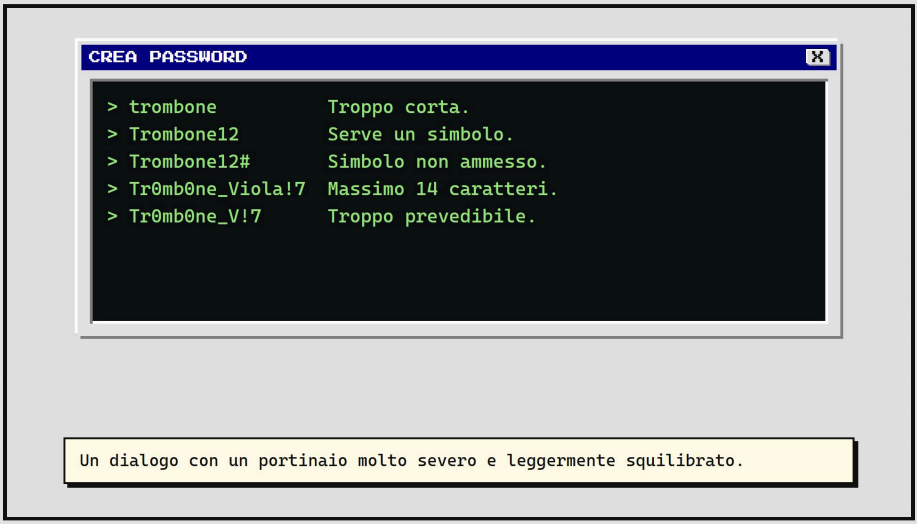
Per tre anni hai usato la stessa password. La conoscevi come il nome di tuo figlio. Poi il sistema decide che è scaduta e ne vuole una nuova, diversa dalle ultime dodici. La inventi con cura. Il giorno dopo è sparita dalla memoria come un sogno al risveglio.

Inizia allora la caccia al reset. “Inserisca la risposta alla domanda segreta.” Il nome del tuo primo animale. Avevi scritto “Fuffi”, “fuffi” o “FUFFI”? Il sistema non lo dice. Ti blocca per trenta minuti per riflettere.

Il post-it giallo sotto la tastiera, intanto, ride. Lui c'è sempre stato.

Consiglio vero: usate un gestore di password. Ne ricordate una sola, robusta, e lui custodisce le altre. Il post-it non conta come gestore.

LEGGE DELLA PASSWORD MAI ABBASTANZA



Il sito rifiuta ogni password finché non diventa impossibile da ricordare, e poi rifiuta anche quella perché contiene un carattere non ammesso.

— Anonimo, *folklore informatico*

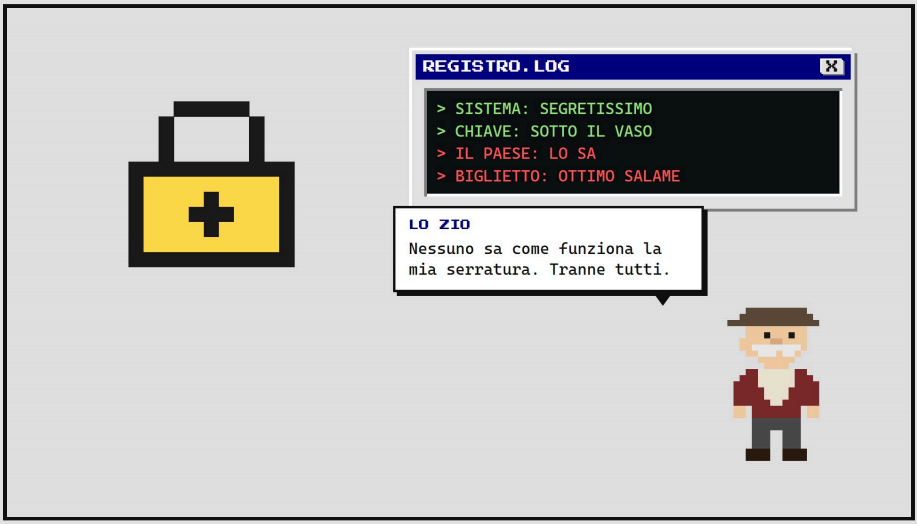
“La password deve contenere almeno otto caratteri.” Fatto. “Una maiuscola.” Fatto. “Un numero.” Fatto. “Un simbolo.” Fatto. “Non può contenere il simbolo che hai usato.” Ah.

È un dialogo con un portinaio molto severo e leggermente squilibrato. Ogni tentativo apre una regola nuova, come in un videogioco in cui il livello finale è scritto da un comitato.

Alla fine ottieni una sequenza tipo “Tr0mb0ne_Viola!7”, orgogliosa come una bandiera. La scrivi su un foglio. Il foglio va in un cassetto. Il cassetto non ha serratura. La sicurezza informatica ha fatto un giro completo ed è tornata al punto di partenza.

Consiglio vero: meglio una frase lunga di parole a caso che una parola corta piena di simboli. È più sicura e più facile da ricordare.

PRINCIPIO DI KERCKHOFFS



Un sistema crittografico deve restare sicuro anche se il nemico conosce tutto del suo funzionamento: l'unico segreto deve essere la chiave.

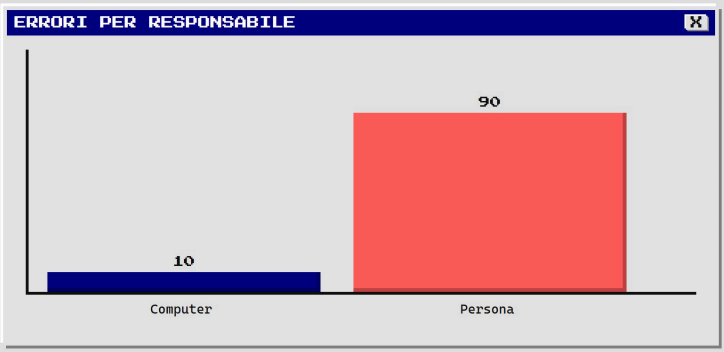
— Auguste Kerckhoffs, *La cryptographie militaire*, 1883

Il contrario si chiama «sicurezza tramite oscurità»: il programma è sicuro perché nessuno sa come funziona. Dura esattamente fino al primo curioso con un pomeriggio libero.

Kerckhoffs ragionava da militare: i congegni vengono catturati, i manuali copiati, le spie corrotte. Se tutta la sicurezza sta nel segreto del meccanismo, basta un furto per perdere tutto. Se sta nella chiave, cambi la chiave e ricominci. Per questo gli algoritmi seri sono pubblici e studiati da tutti.

Mio zio protegge la cantina con un sistema segretissimo: la chiave sotto il vaso di gerani. Il sistema è oscuro, ma il paese lo conosce. Domenica gli hanno lasciato un biglietto: «Ottimo salame, buona annata del vino». Sicurezza tramite gerani: sconsigliata.

LEGGI DI GILB SULL'INAFFIDABILITA



Il computer sbaglia sempre allo stesso modo. La persona sbaglia con creatività.

I computer sono inaffidabili, ma gli esseri umani lo sono ancora di più; e qualsiasi sistema che dipende dall'affidabilità umana è inaffidabile.

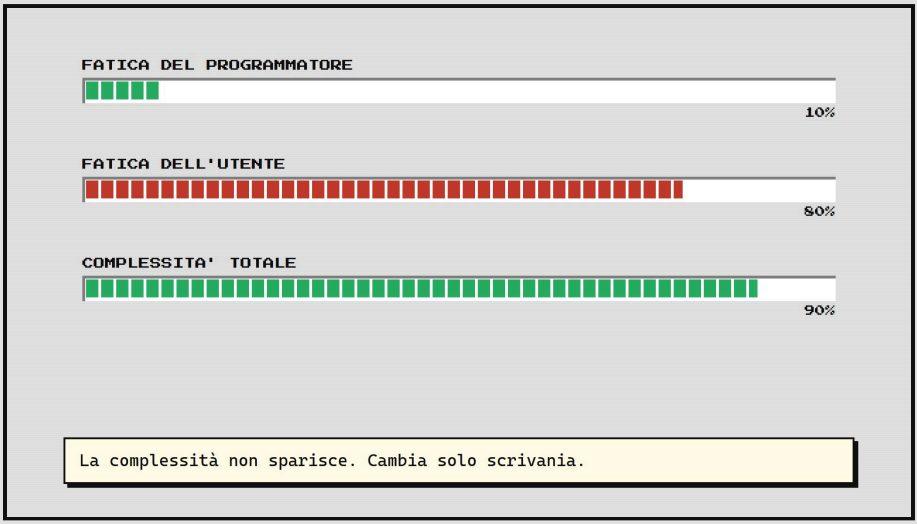
— Tom Gilb, *Leggi diffuse nelle raccolte degli anni '70 (es. Arthur Bloch); numerazione e formulazioni variano tra le fonti (attribuzione in parte incerta)*

Il computer sbaglia, sì. Ma sbaglia sempre allo stesso modo, e prima o poi lo capisci. L'essere umano sbaglia con creatività: oggi la password sul post-it, domani il post-it sullo schermo, dopodomani lo schermo fotografato e condiviso nel gruppo di famiglia.

La lezione è da ingegneri: non progettare sistemi che funzionano solo se tutti sono attenti. Le persone sono stanche, distratte, hanno fretta. I controlli automatici, le conferme e i backup esistono per loro.

In un'azienda avevano una procedura di backup infallibile: ogni sera un dipendente cambiava il disco. Funzionò per anni. Poi il dipendente andò in pensione. Il disco è ancora lì, inserito dal 2015, fedele come un cane alla stazione.

LEGGE DI TESLER



Ogni applicazione ha una quantità di complessità che non si può eliminare: si può solo decidere se se ne fa carico il programmatore o l'utente.

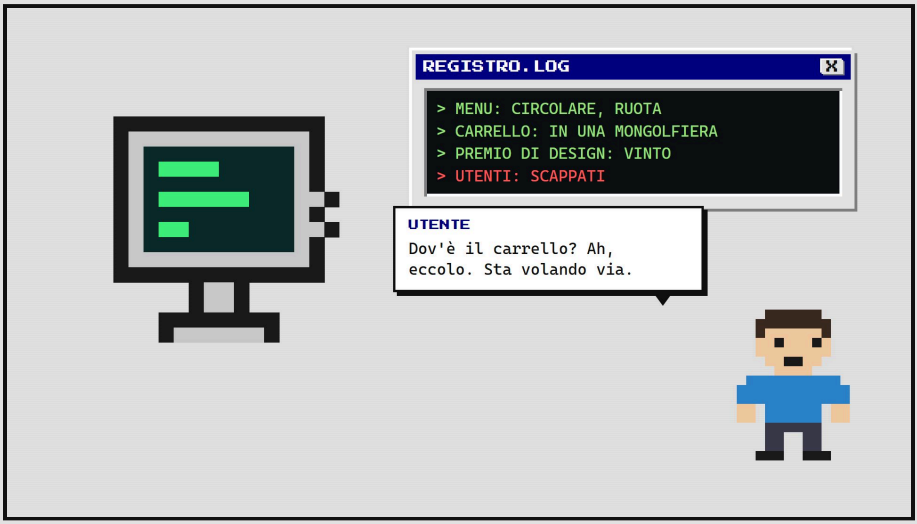
— Larry Tesler, *Legge di conservazione della complessità, formulata negli anni '80 durante il lavoro sulle interfacce, anche in Apple*

La complessità è come l'energia: non si crea e non si distrugge, si sposta. Un modulo con un solo campo «semplice» significa che dietro qualcuno ha scritto mille righe per capire cosa intendeva l'utente.

La legge è un invito alla cortesia: meglio che soffra uno sviluppatore una volta, piuttosto che diecimila utenti tutti i giorni. Purtroppo la tentazione inversa è forte, e nasce il modulo con quaranta campi obbligatori.

In ufficio abbiamo semplificato la stampante: ora ha un solo pulsante. Tutta la complessità è stata trasferita a me, che ricevo ventidue telefonate al giorno per spiegare cosa fa quel pulsante. La conservazione funziona. Io un po' meno.

LEGGE DI JAKOB



Gli utenti passano la maggior parte del tempo su altri siti, quindi si aspettano che anche il tuo funzioni come quelli che conoscono già.

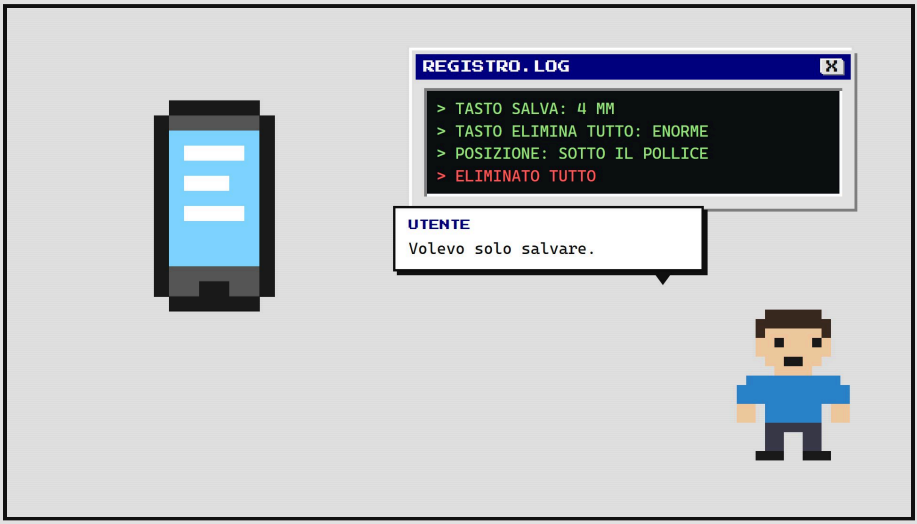
— Jakob Nielsen, *Alertbox, Nielsen Norman Group, 2000*

Il carrello sta in alto a destra. Il logo in alto a sinistra riporta alla home. La lente d'ingrandimento cerca. Nessuno l'ha deciso per legge, ma milioni di persone l'hanno imparato a forza di usarlo ovunque. La legge di Jakob dice: non combattere queste abitudini, cavalcale.

Ogni tanto arriva il designer visionario che vuole stupire. Menù circolare che ruota con lo scroll. Pulsante di acquisto nascosto in una mongolfiera animata. Il sito vince un premio. Gli utenti scappano verso un sito noioso dove il carrello sta dove deve stare.

Essere originali va benissimo nel colore, nelle foto, nei testi. Nella navigazione, l'originalità è un costo che paga l'utente. Io una volta ho messo il menù in basso a sinistra per motivi artistici. Il cliente l'ha trovato dopo una settimana. Era venuto a chiedermi perché il suo sito avesse una sola pagina.

LEGGE DI FITTS



Il tempo per raggiungere un bersaglio con un movimento dipende dalla distanza e dalla sua dimensione: più è lontano e piccolo, più ci si mette e più si sbaglia.

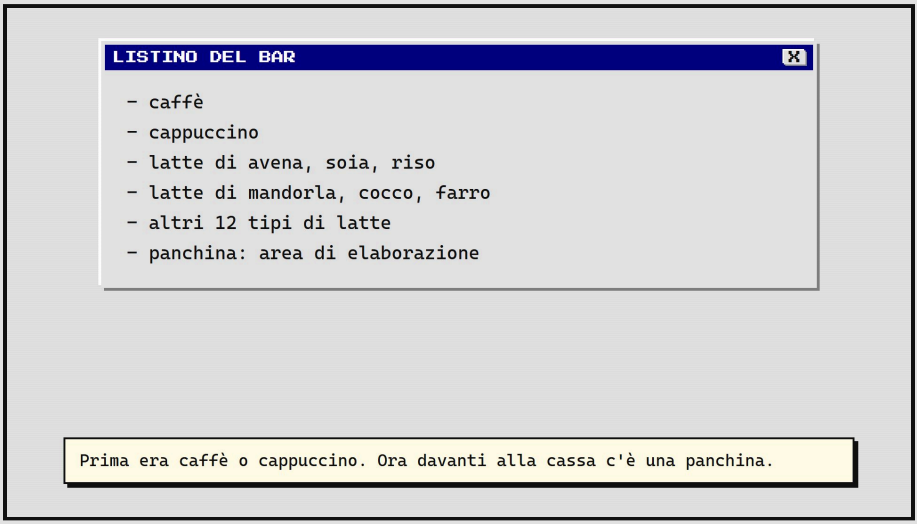
— Paul Fitts, *Journal of Experimental Psychology*, 1954

Uno psicologo degli anni Cinquanta studiava i movimenti delle mani, molto prima che esistessero i mouse. La sua formula, però, sembra scritta per chi fa interfacce: un bottone grande e vicino si clicca in fretta, uno piccolo e lontano è una prova di abilità da luna park.

Per questo i bottoni importanti dovrebbero essere grandi, e quelli pericolosi lontani. Invece capita spesso il contrario: il tasto «Salva» è un francobollo, e quello «Elimina tutto» è grande come una porta, proprio sotto il pollice. Sul telefono diventa uno sport estremo.

I bordi dello schermo sono bersagli infiniti, perché il mouse si ferma lì da solo. Lo dimostra mio zio, che apre sempre il menù Start lanciando il cursore in basso a sinistra con un colpo di polso. Dice che è una tecnica di pesca. Non sbaglia mai.

LEGGE DI HICK



Il tempo necessario per decidere cresce con il numero di opzioni disponibili (in modo logaritmico, non lineare).

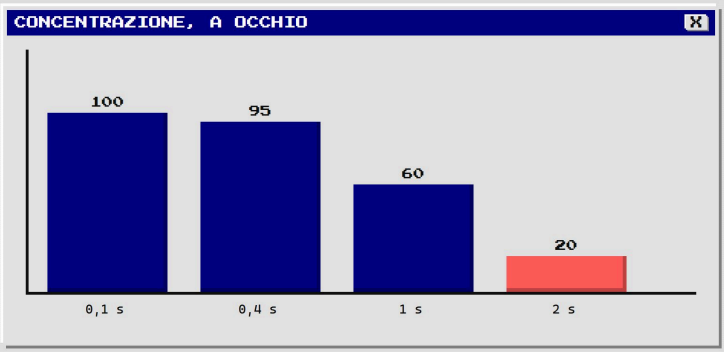
— William Edmund Hick (e Ray Hyman), *Quarterly Journal of Experimental Psychology, 1952; studio di Hyman, 1953 (detta anche legge di Hick-Hyman)*

Un menù con tre voci si legge e si sceglie in un attimo. Un menù con quaranta voci, ognuna con un sottomenù, è un pomeriggio della vita. La legge di Hick dice che più opzioni metti davanti a una persona, più tempo impiega a decidere. Cresce meno che in proporzione, ma cresce.

È il motivo per cui i buoni programmi nascondono le impostazioni avanzate e mostrano solo quello che serve. Ed è il motivo per cui la finestra delle opzioni di certi software, con trentadue schede e duecento caselle, provoca nell'utente uno sguardo fisso verso l'orizzonte.

Lo vedo ogni giorno al bar. Prima c'era «caffè o cappuccino». Adesso c'è un listino con diciotto tipi di latte. Il barista ha aggiunto una panchina davanti alla cassa. La chiama area di elaborazione.

SOGLIA DI DOHERTY



Sopra la soglia la mente comincia a vagare. E quando vaga, va su internet.

La produttività cresce nettamente quando il computer risponde all'utente in meno di circa 400 millisecondi: sotto quella soglia nessuno dei due aspetta l'altro.

— Walter J. Doherty e Arvind J. Thadhani, *«The Economic Value of Rapid Response Time»*, IBM, 1982 *(il valore di 400 ms è citato soprattutto da fonti successive)*

All'inizio degli anni Ottanta si pensava che due secondi di attesa fossero accettabili. Uno studio IBM sostenne il contrario: se il computer risponde in meno di mezzo secondo circa, l'utente resta concentrato e lavora molto di più. Sopra, la mente comincia a vagare. E quando vaga, va su internet.

Per questo le interfacce moderne mostrano subito qualcosa: un'animazione, una barra, uno scheletro grigio della pagina. Non è più veloce, ma sembra vivo. L'utente non vuole il risultato immediato. Vuole sapere che qualcuno lo sta ascoltando.

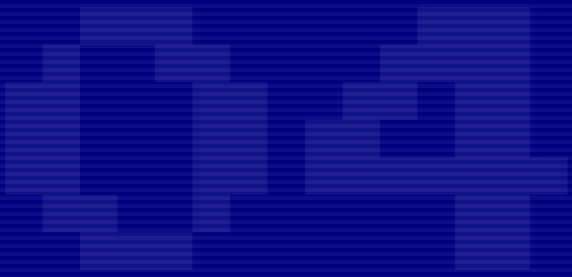
Ho applicato la soglia di Doherty all'assistenza telefonica. Rispondo entro quattrocento millisecondi con «Sì, eccomi». Poi, ovviamente, ci metto un'ora a risolvere. Ma il cliente si sente seguito. È tutto nella barra di avanzamento emotiva.

IL CONSIGLIO DEL SISTEMISTA

Gran parte del tempo dell'assistenza se ne va a capire cosa è successo. Potete regalarlo al vostro tecnico, che vi ricorderà con affetto. Quando qualcosa non va, preparate queste cinque informazioni prima di chiamare:

- **Cosa stavate facendo**, passo per passo, compreso il «non ho toccato niente» nei dettagli.
- **Il messaggio esatto**, fotografato. Su Windows 10 e 11 i tasti Windows + Maiusc + S catturano una parte dello schermo.
- **Da quando** succede, e se prima funzionava.
- **Cosa è cambiato** nel frattempo: un aggiornamento, un programma nuovo, un trasloco della scrivania.
- **Se succede sempre** o solo a volte, su un solo computer o su tutti.

Con queste risposte, molti guasti si risolvono al telefono. Gli altri si risolvono più in fretta, perché il tecnico arriva con l'attrezzo giusto già in mano.



CAPITOLO 4

RETE

Reti e Internet: il filo invisibile

La rete è la cosa più usata e meno capita della casa moderna. Tutti la vogliono, nessuno sa dove sia. È nel router? Nel muro? Nell'aria? La mia vicina sostiene che passa dalla presa del telefono e che di notte riposa. Non l'ho contraddetta: in certi quartieri ha ragione.

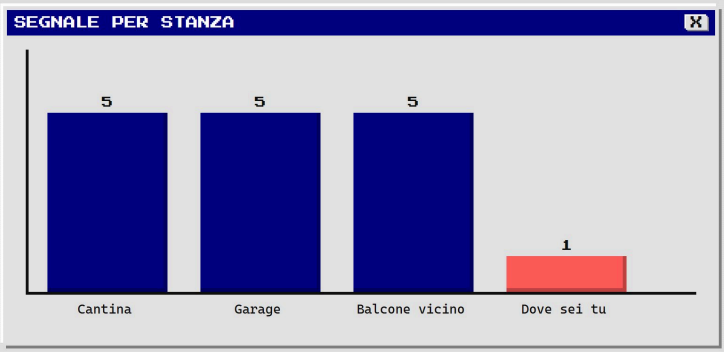
Il capitolo si apre con le leggi della connessione di casa e d'ufficio: il Wi-Fi che prende ovunque tranne dove sei, il download che muore al novantanove, la videochiamata che si congela quando tocca a te. Poi passa alle leggi del grande Internet, quelle nate tra forum, giornali online e chat di gruppo. Qui la tecnologia conta poco e le persone contano moltissimo: chi corregge sbagliando, chi titola con il punto interrogativo, chi prova a nascondere una cosa e la rende famosa.

Sono leggi che valgono anche fuori dallo schermo. Quella di Brandolini, per esempio, l'ho vista all'opera a cena, a una festa di paese e una volta in una sala d'attesa: smentire una sciocchezza costa sempre più fatica che dirla. Questo capitolo non la smentisce. La documenta.

CONTENUTO.DIR

01 Legge del Wi-Fi dispettoso
02 Legge del router da riavviare
03 Legge della videochiamata decisiva
04 Legge del download al 99%
05 Legge del sito che funziona ovunque
06 Legge di Metcalfe
07 Legge di Nielsen sulla banda
08 Principio di robustezza
09 Legge di Cunningham
10 Legge di Skitt
11 Legge dei titoli di Betteridge
12 Legge di Poe
13 Legge di Godwin
14 Effetto Streisand
15 Legge di Brandolini

LEGGE DEL WI-FI DISPETTOSO



Tacche di segnale. Il Wi-Fi è un gatto: va dove vuole lui.

Il Wi-Fi funziona perfettamente in ogni stanza della casa tranne quella in cui ti trovi.

— Anonimo, *folklore informatico*

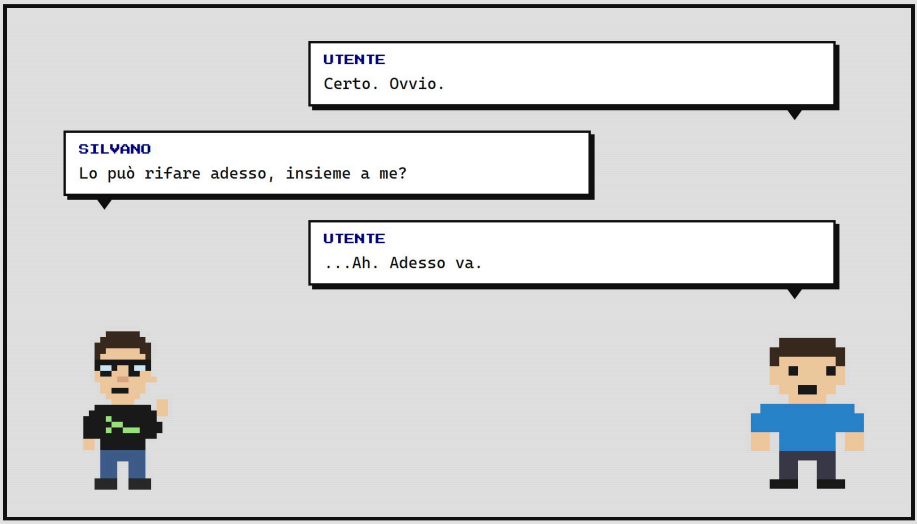
Il segnale Wi-Fi è un gatto. Va dove vuole lui. In cantina prende benissimo, in garage pure, sul balcone del vicino si vede in alta definizione. Ma tu sei in salotto, e in salotto ci sono due tacche che lampeggiano come una richiesta d'aiuto.

Allora inizia il rito. Ti sposti di mezzo metro. Alzi il telefono al cielo come un'offerta. Ti siedi sul bracciolo del divano in una posizione che il fisioterapista sconsiglia. Funziona. Ti muovi di un centimetro. Non funziona più.

Ho un cliente che ha messo il router in un armadio di legno massiccio “per non vederlo”. Ora non lo vede nemmeno il Wi-Fi.

Consiglio da tecnico: il router va in posizione centrale, alta e libera. Muri spessi, specchi e acquari sono i suoi nemici naturali.

LEGGE DEL ROUTER DA RIAVVIARE



Prima di chiamare l'assistenza nessuno riavvia il router; dopo aver chiamato, il problema si risolve riavviando il router.

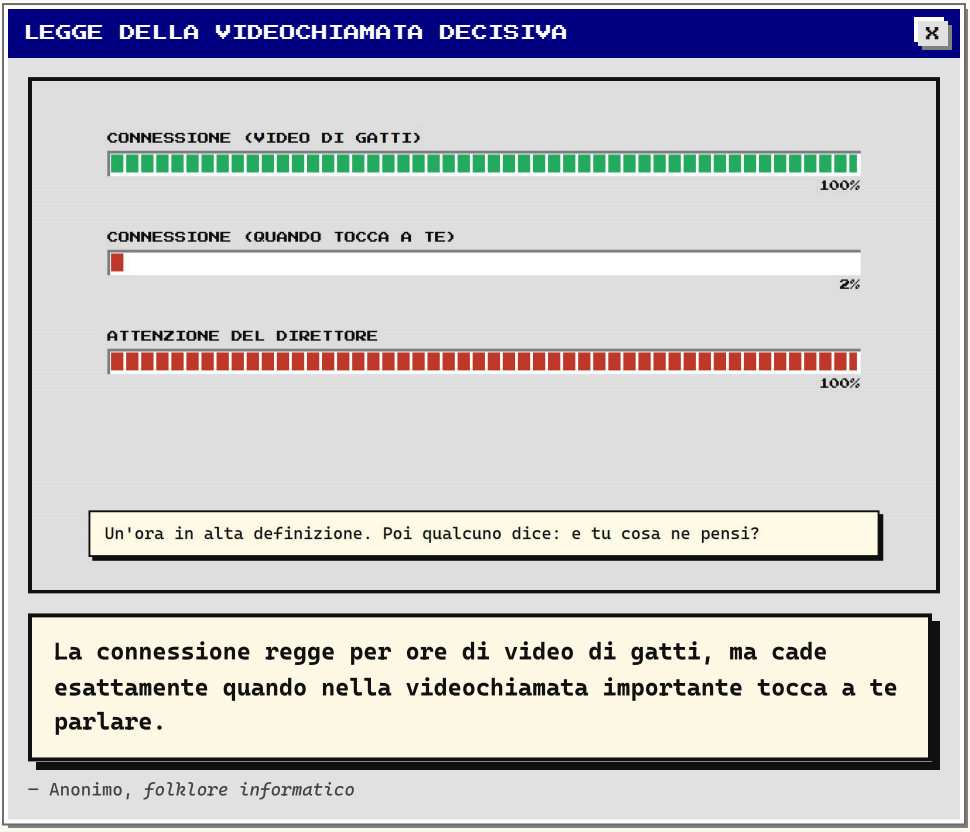
— Anonimo, *folklore informatico*

Il riavvio del router è il sacramento laico dell'era digitale. Tutti ne conoscono il potere, nessuno ci crede davvero finché non lo dice qualcuno con una cuffia e un numero verde.

“Ha provato a spegnere e riaccendere?” “Certo.” “Lo può rifare adesso con me?” Silenzio. Rumore di mobili spostati. Lucine che si spengono. Lucine che si accendono. “Ah. Adesso va.”

Fratelli e sorelle della connessione! Il router non è un apparecchio. È uno stato d'animo che ogni tanto va resettato, come noi il lunedì mattina.

Consiglio da tecnico: spegnete, aspettate almeno trenta secondi, poi riaccendete e date al router un paio di minuti per ripartire. Se il problema torna spesso, la causa va cercata: il riavvio è un cerotto, non la cura.



La connessione regge per ore di video di gatti, ma cade esattamente quando nella videochiamata importante tocca a te parlare.

— Anonimo, *folklore informatico*

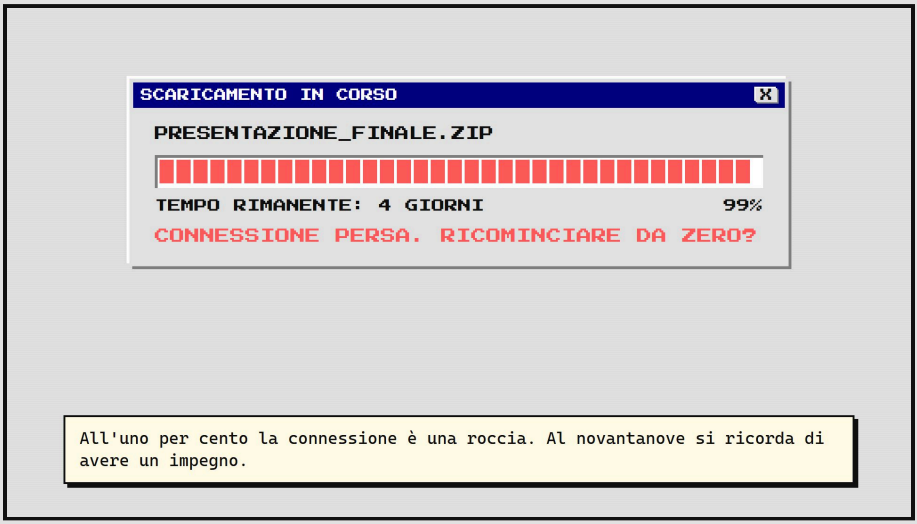
Per tutta la riunione la connessione è stata impeccabile. Hai visto colleghi in alta definizione, hai sentito il respiro del direttore, hai contato le piante dietro la sua scrivania. Poi qualcuno dice: “E tu cosa ne pensi?”

In quel preciso istante il tuo volto si congela in un'espressione indecifrabile, a metà tra lo stupore e lo starnuto. La voce diventa quella di un robot sott'acqua. Gli altri ti vedono così per quaranta secondi, e decidono di passare oltre.

Quando torni online, la riunione ha già deciso. Su di te. In tua assenza.

Consiglio vero: per le chiamate importanti, se potete, collegate il computer al router con un cavo di rete. Il Wi-Fi è comodo, il cavo è affidabile.

LEGGE DEL DOWNLOAD AL 99%



Un download o un caricamento si interrompe sempre al novantanove per cento, mai all'uno.

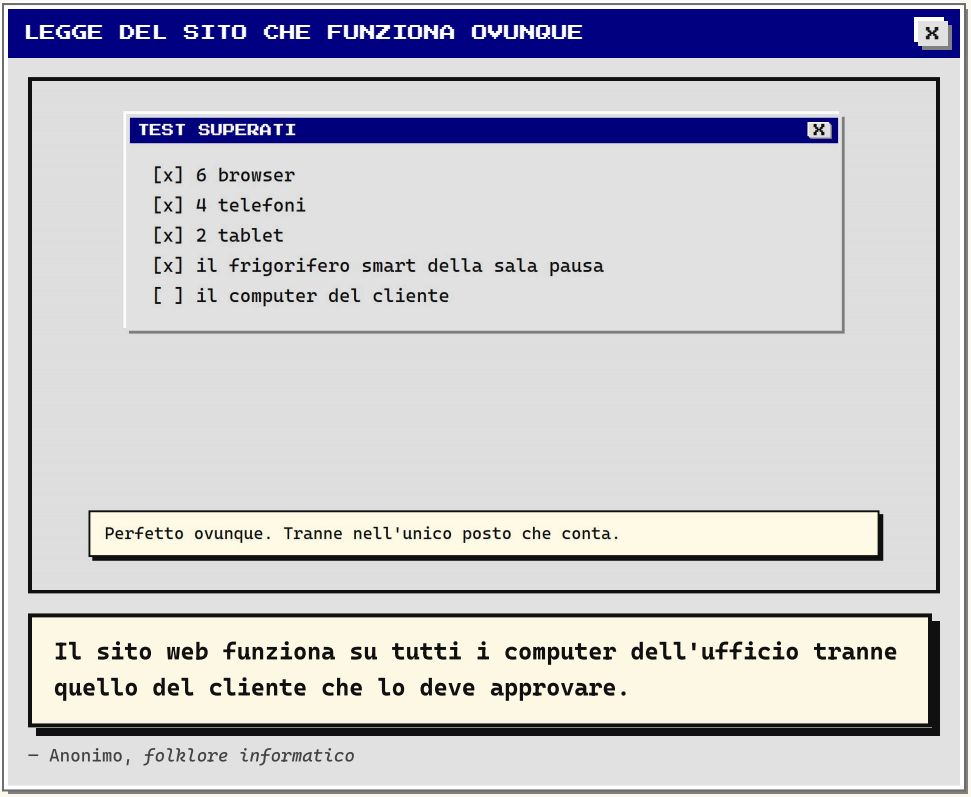
— Anonimo, *folklore informatico*

Nessun download è mai fallito all'uno per cento. Sarebbe un fallimento onesto, senza dolore. No: la barra avanza lenta per un'ora, raggiunge il novantanove, ti fa intravedere la terra promessa, e poi compare la scritta “errore di rete”.

Gli scienziati la chiamano crudeltà. I tecnici la chiamano “l'ultimo uno per cento”, che notoriamente dura quanto il restante novantanove. È lì che il computer fa i calcoli seri, dicono. In realtà sta solo decidendo se volerti bene.

Ho visto uomini adulti fissare una barra ferma al novantanove per venti minuti, senza respirare, come si guarda un rigore decisivo.

Consiglio da tecnico: per file grandi usate programmi o servizi che supportano la ripresa del download. Così se cade, riparte da dove era arrivato.



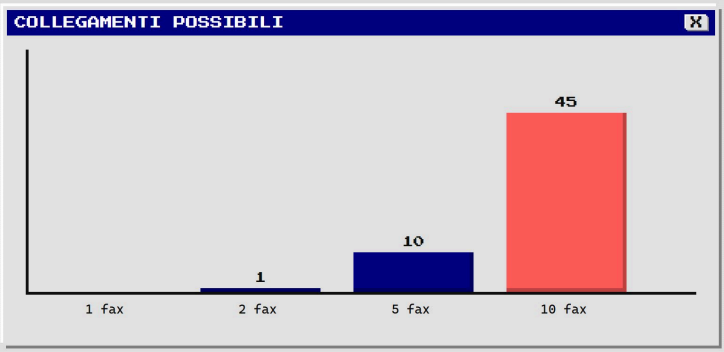
Il sito è stato testato su sei browser, quattro telefoni, due tablet e il frigorifero smart della sala pausa. Tutto perfetto. Poi il cliente lo apre dal suo computer e vede un menu spostato a sinistra e un logo grande quanto un camion.

Il computer del cliente è un luogo misterioso. Ha un browser che si aggiorna solo nei giorni pari, sei barre degli strumenti installate da un nipote nel 2016 e lo zoom impostato al centocinquanta per cento “perché si legge meglio”.

E il cliente non chiama mai per dire “sul mio computer non si vede bene”. Dice: “Il sito non funziona”. Con il tono di chi annuncia un'eclissi.

Consiglio vero: prima di dare la colpa al sito, provate a svuotare la cache e a riportare lo zoom al cento per cento. Risolve più problemi di quanto si creda.

LEGGE DI METCALFE



Un fax da solo è un soprammobile che fa rumore. Dieci fax sono una società, con pettegolezzi.

Il valore di una rete cresce con il quadrato del numero di dispositivi (o persone) collegati: ogni nuovo nodo aumenta le connessioni possibili di tutti gli altri.

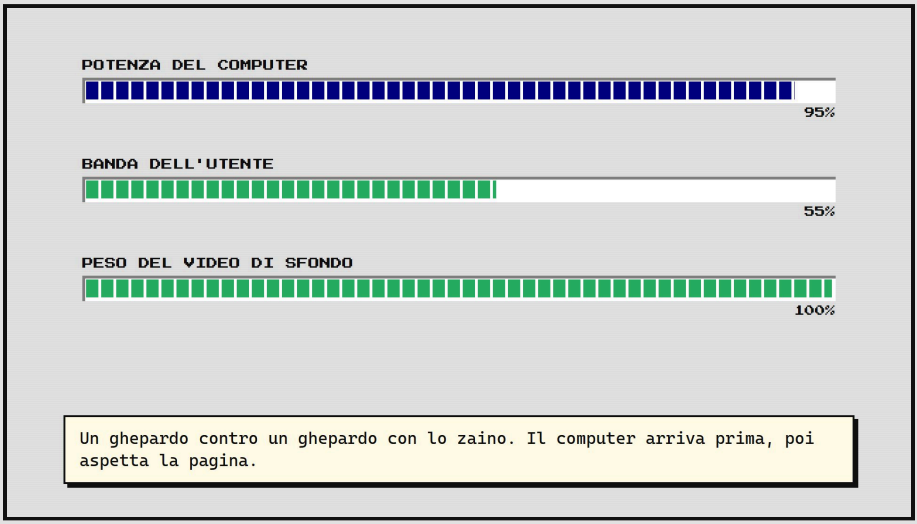
— Robert Metcalfe, *Formulata intorno al 1980 per promuovere Ethernet; il nome «legge di Metcalfe» è di George Gilder, 1993*

Un fax da solo non serve a niente. È un soprammobile che fa rumore. Due fax possono parlarsi. Dieci fax formano già una società, con pettegolezzi e carta termica. Questa è la legge di Metcalfe: ogni nuovo collegato fa crescere il valore della rete per tutti, e non di poco, al quadrato.

È il motivo per cui nessuno lascia un social o una chat di gruppo: se te ne vai, perdi tutti gli altri. Il valore non è nel programma, è nella gente intrappolata dentro. Felicemente, si intende.

Il rovescio della medaglia lo conosco bene. La chat del condominio ha quaranta membri. Secondo la formula vale moltissimo. In pratica produce milleseicento possibili litigi sul parcheggio delle biciclette. Anche quello, matematicamente, è valore.

LEGGE DI NIELSEN SULLA BANDA



La velocità di connessione di un utente evoluto cresce di circa il 50% all'anno: più lentamente della potenza dei computer.

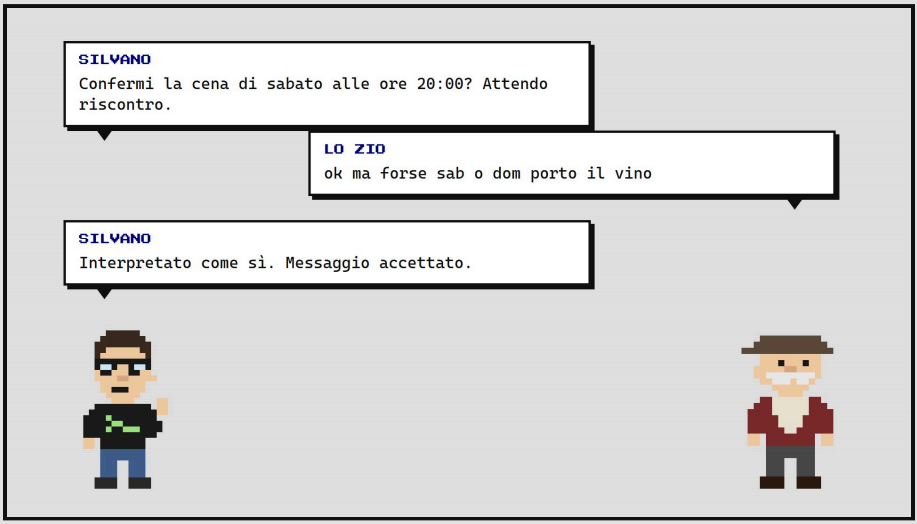
— Jakob Nielsen, *Alertbox, Nielsen Norman Group, 1998 (aggiornata nel 2008 e 2019)*

I processori raddoppiano ogni due anni, la banda cresce del cinquanta per cento all'anno. Sembra tanto, ma è come una gara tra un ghepardo e un ghepardo con lo zaino. Il computer arriva sempre prima. Poi si siede e aspetta che la pagina carichi.

La lezione per chi fa siti è semplice: non dare per scontato che tutti abbiano la fibra. Da qualche parte c'è sempre un utente in una cascina, con una chiavetta appoggiata sul davanzale nell'unico punto dove prende, che prova ad aprire il tuo video di sfondo da novanta megabyte.

Io quel cliente ce l'ho. Per migliorare il segnale ha messo la chiavetta in cima a un palo per i fagioli. Funziona. Quando c'è vento, però, internet oscilla.

PRINCIPIO DI ROBUSTEZZA



Sii conservativo in ciò che invii e liberale in ciò che accetti dagli altri.

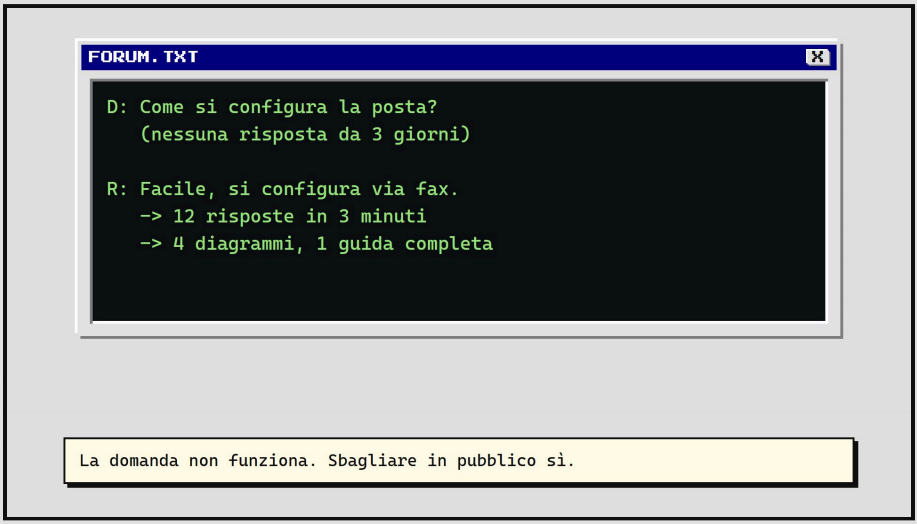
— Jon Postel, *Specifiche dei protocolli Internet*, *RFC 760 (1980) e RFC 793 (1981)*

È il galateo di Internet: quando parli, usa il protocollo alla lettera; quando ascolti, sii comprensivo con chi sbaglia una virgola. Grazie a questa gentilezza, macchine scritte da persone diverse riescono a capirsi.

C'è un rovescio: se tutti accettano errori, nessuno li corregge mai. Dopo trent'anni di tolleranza i browser leggono pagine HTML che sembrano scritte da un gatto sulla tastiera, e devono continuare a farlo per sempre.

Ho applicato il principio alla mia vita sociale. Mando messaggi impeccabili, con punteggiatura da notaio. Accetto qualunque risposta, anche solo un pollice. Mia zia mi ha risposto con una melanzana. L'ho interpretata come un «sì». Ora sono testimone di nozze.

LEGGE DI CUNNINGHAM



Il modo migliore per ottenere la risposta giusta su Internet non è fare una domanda, ma scrivere la risposta sbagliata.

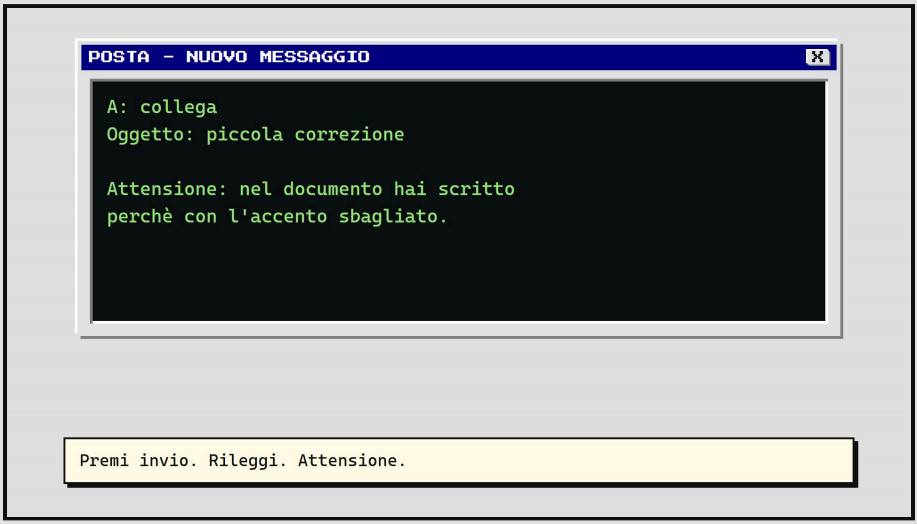
— Steven McGeady, *Battezzata da Steven McGeady nel 2010 in onore di Ward Cunningham, che però ha negato di averla mai detta (attribuzione contestata)*

Chiedi su un forum come si configura il server di posta: silenzio, una balla di fieno che rotola. Scrivi con sicurezza una procedura sbagliata: in tre minuti arrivano dodici esperti, quattro diagrammi e una correzione firmata da un ingegnere in pensione.

Funziona perché correggere qualcuno dà più soddisfazione che aiutarlo. È un motore di conoscenza alimentato dal desiderio di avere ragione. Il paradosso finale: l'uomo a cui è dedicata dice di non averla mai pronunciata. La legge si è dimostrata da sola.

Io la uso per tutto. Ho scritto in un gruppo che Joomla si installa con il fax. Ora ho una guida completa, aggiornata, con screenshot. E anche un fax, che un utente mi ha mandato per dimostrare che non funziona.

LEGGE DI SKITT



Qualsiasi messaggio che corregge un errore altrui conterrà, quasi sicuramente, almeno un errore suo.

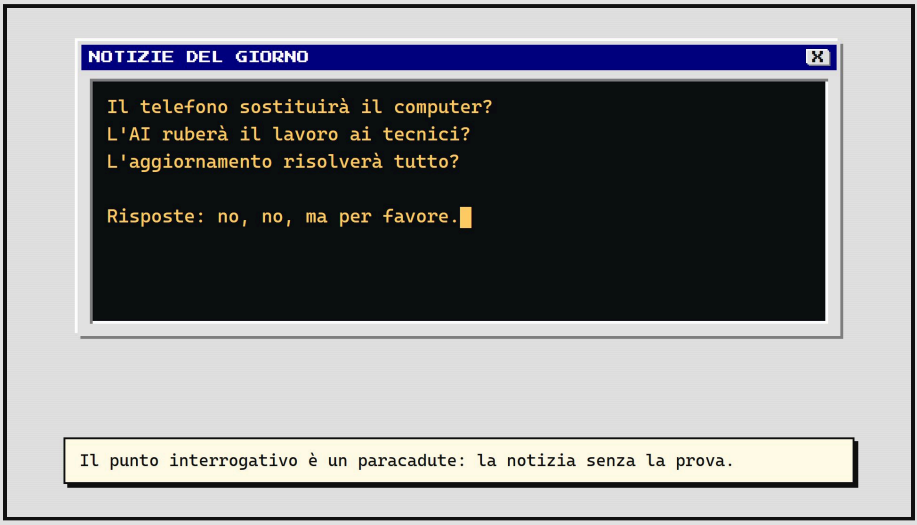
— Anonimo (pseudonimo «Skitt»), *Usenet, gruppo alt.usage.english, 1999 circa (attribuzione incerta; varianti precedenti note come «legge di Muphry»)*

Scrivi una mail al collega: «Attenzione, nel documento hai scritto perchè con l'accento sbagliato». Premi invio. Rileggi. Hai scritto «Attensione». Il collega risponde con una sola parola, e quella parola è perfetta.

La legge di Skitt è una legge di natura della correzione. L'universo non tollera la superbia ortografica: ogni volta che fai il maestrino, una lettera ribelle si stacca dalla tastiera e va a nascondersi nel punto più visibile del tuo messaggio. Vale anche per il codice: la patch che corregge il bug del collega introduce un bug tuo.

Io per prudenza non correggo più nessuno. Ho delegato a un correttore automatico. Ieri ha cambiato «server» in «serve». Il cliente ha letto «il serve è giù» e mi ha chiesto chi l'avesse spinto.

LEGGE DEI TITOLI DI BETTERIDGE



A qualsiasi titolo di giornale che termina con un punto interrogativo si può rispondere con un «no».

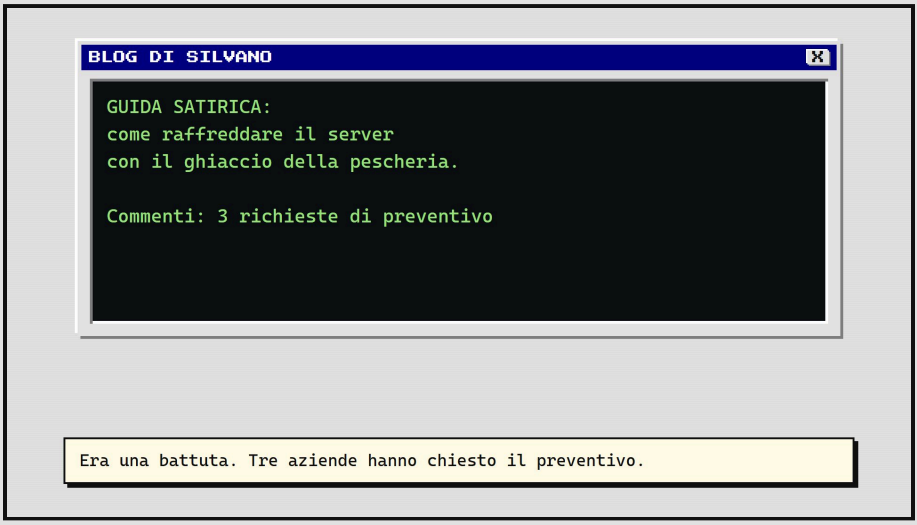
— Ian Betteridge, *Articolo sul suo blog, febbraio 2009 (idea già circolante con altri nomi negli anni '80)*

«Questo nuovo telefono sostituirà il computer?» No. «L'intelligenza artificiale ruberà il lavoro ai tecnici?» No, anzi, ci tocca configurarla. «Il nuovo aggiornamento risolverà tutti i problemi?» Ma per favore.

La legge di Betteridge svela un trucco antico: se il giornalista avesse prove per un sì, scriverebbe un'affermazione. Il punto interrogativo è un paracadute. Permette di lanciare la notizia clamorosa senza doverla dimostrare, e di incassare i clic di chi spera.

Ho fatto un esperimento sul mio blog. Ho pubblicato «Il floppy disk tornerà di moda?». Tremila visite. Poi ho pubblicato «Il floppy disk non tornerà di moda». Due visite, una era mia madre. Il punto interrogativo, ho capito, è il vero motore dell'editoria digitale.

LEGGE DI POE



Senza un segnale chiaro dell'intenzione dell'autore, una parodia di una posizione estrema è indistinguibile da un'opinione sincera dello stesso tipo.

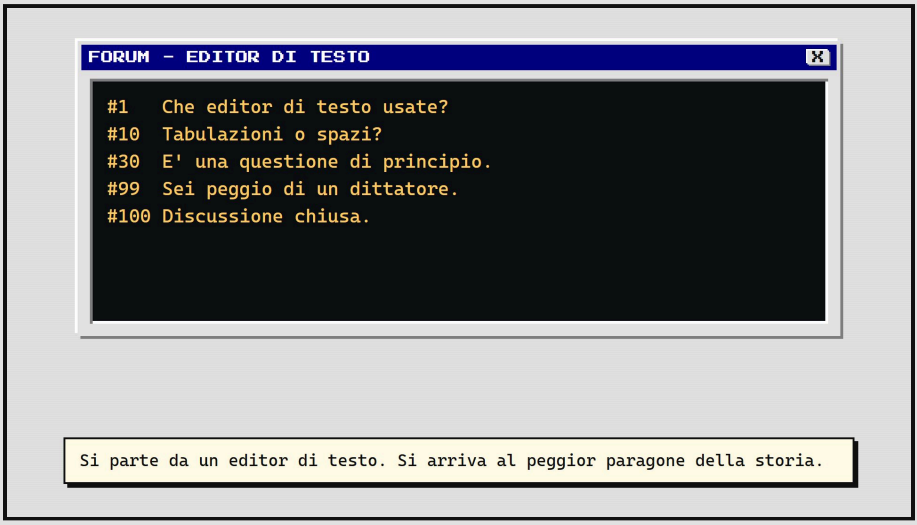
— Nathan Poe, *Forum christianforums.com, 2005 (un avvertimento simile risale a Usenet, 1983)*

Scrivi su internet che il modo migliore per proteggere la password è stamparla e attaccarla al monitor, così i ladri non la cercano dove è ovvia. È una battuta. Tre persone ti ringraziano. Una lo fa davvero. Un'altra apre un corso a pagamento.

Questa è la legge di Poe: online l'ironia non ha tono di voce, e certe opinioni reali sono così estreme che la parodia non riesce più a superarle. Per questo sono nate le faccine. Sono il nostro sopracciglio alzato digitale.

Io una volta ho pubblicato una guida satirica su come raffreddare il server con il ghiaccio della pescheria. Mi hanno scritto da tre aziende per chiedere il preventivo. Una ha già ordinato le orate. Dice che i log adesso sanno di mare.

LEGGE DI GODWIN



Più una discussione online si allunga, più la probabilità che qualcuno tiri in ballo un paragone con Hitler o con i nazisti si avvicina alla certezza.

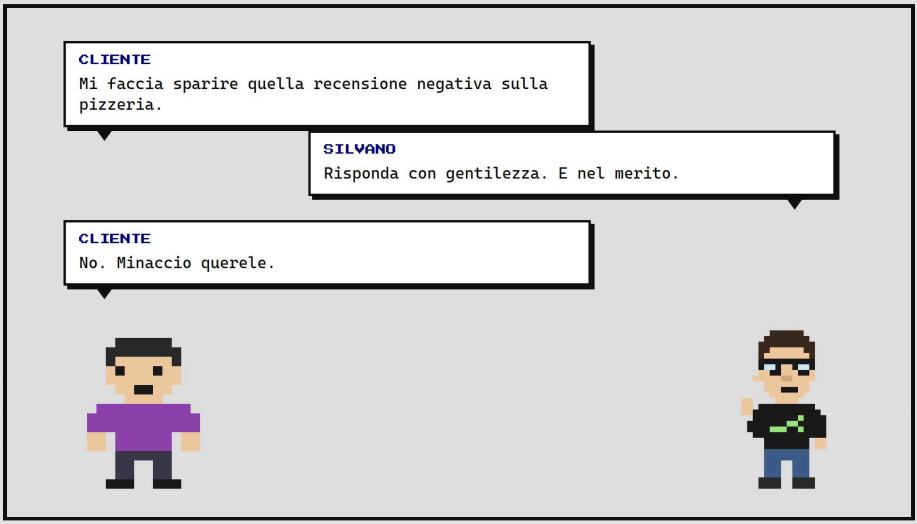
— Mike Godwin, *Usenet*, *1990*

Ogni discussione online parte educata. Qualcuno chiede quale editor di testo usare. Al decimo messaggio si parla di tabulazioni contro spazi. Al trentesimo di principi morali. Al centesimo qualcuno paragona l'avversario al peggior dittatore della storia. Per un editor di testo.

La legge di Godwin non dice che succede, dice che è inevitabile, come la gravità o come il cavo HDMI che è sempre dalla parte sbagliata. Nei vecchi forum valeva anche una regola non scritta: chi arriva per primo a quel paragone ha perso la discussione.

Io modero un forum di appassionati di stampanti. Abbiamo resistito trecentoventi messaggi. Poi uno ha definito il driver di una certa marca un regime totalitario. A essere onesti, aveva qualche ragione. Ma il thread è stato chiuso lo stesso.

EFFETTO STREISAND



Cercare di nascondere, rimuovere o censurare un'informazione online finisce per renderla molto più visibile e diffusa.

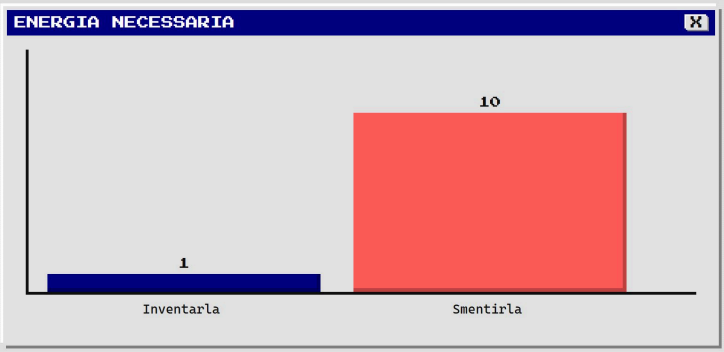
— Mike Masnick, *Techdirt, gennaio 2005 (ispirato a un caso del 2003 su una foto aerea)*

Il nome viene da un caso del 2003: una causa per far togliere da un archivio online la foto aerea di una villa, scattata per documentare l'erosione della costa. Prima della causa la foto l'avevano vista pochissimi. Dopo, l'ha vista mezzo mondo. Un blogger ci costruì sopra il nome di un fenomeno.

Internet funziona come un gatto: se gli dici di non guardare una scatola, guarderà solo quella. Ogni richiesta di cancellazione è un cartello luminoso con scritto «qui c'è qualcosa di interessante». Copie, archivi e schermate si moltiplicano più in fretta di qualsiasi avvocato.

Un cliente mi chiese di far sparire una recensione negativa sulla sua pizzeria. Gli consigliai di rispondere con gentilezza. Preferì minacciare querele. Ora la recensione è la più letta della provincia. La pizza, nel frattempo, è migliorata. Ma nessuno lo sa.

LEGGE DI BRANDOLINI



Tre secondi per inventarla. Due ore, quattro link e un grafico per smentirla.

L'energia necessaria per smentire una sciocchezza è di un ordine di grandezza superiore a quella servita per produrla.

— Alberto Brandolini, *Messaggio su Twitter, gennaio 2013 (nota anche come principio di asimmetria delle fesserie)*

Scrivere «il 5G fa appassire i gerani» richiede tre secondi. Spiegare perché non è vero richiede due ore, quattro link, un grafico e la pazienza di un monaco. Nel frattempo la frase sui gerani ha fatto il giro di tre chat di famiglia.

La legge, formulata da un programmatore italiano, descrive un'asimmetria crudele: inventare costa poco, verificare costa tanto. È il motivo per cui le bufale vincono le gare di velocità, e per cui ogni sistemista ha passato almeno una cena a spiegare che no, il computer non ha preso il virus dalla presa elettrica.

Io ho trovato una contromisura. Quando un cliente mi gira una bufala, rispondo con una bufala opposta, altrettanto breve. Si annullano. Mi sono già guadagnato la fama di persona informata. Ho anche qualche denuncia per gerani, ma è in corso di verifica.

IL CONSIGLIO DEL SISTEMISTA

Quando «Internet va lento», prima di litigare con il gestore fate la prova del cavo. Collegate un computer direttamente al router con un cavo di rete e misurate la velocità con un test online. Poi ripetete la prova in Wi-Fi, nella stanza dove lavorate di solito.

- Se **anche col cavo è lento**, il problema è la linea o il router: chiamate il gestore, con i numeri in mano.
- Se **col cavo è veloce e in Wi-Fi no**, la linea è innocente. Il problema è il segnale in casa: posizione del router, muri, interferenze.

Due dettagli utili. I router moderni trasmettono su due bande: la 5 GHz è più veloce ma attraversa peggio i muri, la 2,4 GHz arriva più lontano ma è più affollata. E se gli ambienti sono grandi, un sistema mesh o un punto di accesso collegato via cavo funzionano molto meglio dei ripetitori economici. A volte spostare il router di due metri vale quanto cambiare contratto.

CAPITOLO 5

DEBUG

Caccia al bug

Qui si entra in laboratorio. Il bug, per chi non lo frequenta, è un errore nel programma. La parola è più vecchia dei computer, ma esiste un registro di lavoro del 1947 con una falena vera incollata sopra, trovata dentro una macchina. Da allora la caccia non si è mai fermata, e la falena ha avuto moltissimi nipoti.

Le leggi di questo capitolo raccontano il bug nel suo habitat naturale. Sparisce quando arriva il collega, si nasconde nell'ultima riga che controlli, aspetta il venerdì pomeriggio, si presenta dieci minuti dopo la pubblicazione. A volte, se sopravvive abbastanza a lungo, gli utenti gli si affezionano e diventa una funzione.

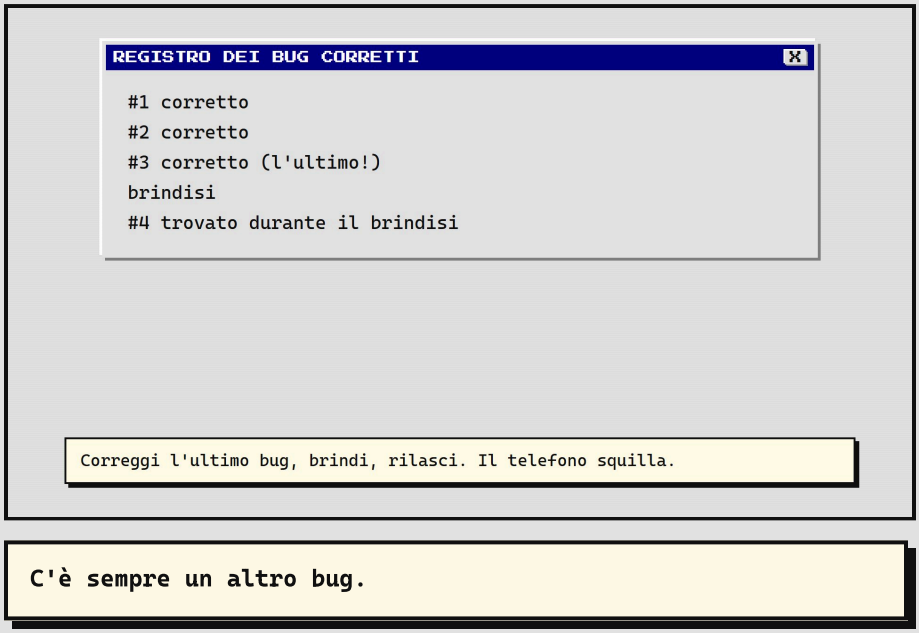
Chi non programma troverà qui la risposta alla domanda che si fa da anni: perché il programma che uso tutti i giorni ogni tanto fa cose strane? Risposta breve: perché c'è sempre un altro bug. Risposta lunga: questo capitolo.

Chi programma, invece, si riconoscerà in ogni pagina. Mi scuso in anticipo per le notti che vi farò tornare in mente.

CONTENUTO.DIR

01 Legge di Lubarsky dell'entomologia cibernetica
02 Legge del sul mio PC funziona
03 Legge del collega taumaturgo
04 Legge della riga innocente
05 Legge di Kernighan
06 Legge della correzione prolifica
07 Legge del bug evoluto
08 Legge del bug puntuale
09 Legge del venerdì pomeriggio
10 Legge della parte non testata
11 Legge del compila dunque funziona
12 Legge del dato di prova
13 Legge del messaggio oracolare
14 Legge del calendario traditore
15 Legge di Linus
16 Legge di Hyrum

LEGGE DI LUBARSKY DELL'ENTOMOLOGIA CIBERNETICA



C'è sempre un altro bug.

— Anonimo, *Detto circolato nelle raccolte di umorismo informatico (es. liste Multics e Usenet); identità di Lubarsky non verificata (attribuzione incerta)*

Quattro parole e la carriera di ogni programmatore è riassunta. Correggi l'ultimo bug, brindi, rilasci. Il telefono squilla. C'era un ultimo-ultimo bug.

Il significato è una saggia umiltà: nessun programma non banale è privo di errori. Si può renderli rari, si può renderli innocui, ma pensare di averli presi tutti è il bug più pericoloso di tutti.

Ho chiamato un disinfestatore per il server. Ha spruzzato qualcosa sul case, ha detto «garanzia sei mesi» ed è andato via. Il giorno dopo il gestionale stampava le fatture in cirillico. Ho richiamato. Mi ha detto che quella specie non era coperta. Entomologia cibernetica: la scienza esatta del sempre.

LEGGE DEL SUL MIO PC FUNZIONA



Qualunque errore segnalato dal cliente è impossibile da riprodurre sul computer del programmatore, dove tutto funziona perfettamente.

— Anonimo, *folklore informatico*

È la frase più pronunciata nella storia dell'informatica, più di «hai provato a riavviare». Il programmatore la dice con sincerità commovente: sul suo computer il programma funziona davvero. Il problema è che il cliente non abita dentro il suo computer.

Il computer dello sviluppatore è un ecosistema protetto, una serra tropicale con tutte le librerie giuste, le impostazioni giuste, la lingua giusta. Il computer del cliente è una giungla con Windows del 2016, tre antivirus in guerra tra loro e la data impostata sul fuso di Honolulu.

C'è stato un periodo in cui ho seriamente valutato di spedire il mio PC ai clienti insieme al software. Avrei risolto il novanta per cento dei problemi. Poi ho capito che il cliente avrebbe comunque chiamato, per dire che il mio PC sul suo tavolo non funzionava.

LEGGE DEL COLLEGA TAUMATURGO



Il bug che ti tormenta da ore sparisce nell'istante in cui chiami un collega a vederlo, e ricompare appena lui torna alla sua scrivania.

— Anonimo, *folklore informatico*

In ogni ufficio c'è un collega dotato di poteri. Non sa programmare meglio di te, non ha studiato di più. Semplicemente, quando si avvicina al tuo monitor, i bug si mettono sull'attenti e fingono di non esistere.

«Fammi vedere», dice. Tu rifai esattamente gli stessi passaggi, con la voce tremante di chi denuncia un fantasma. Il programma funziona. Lui ti batte una mano sulla spalla con l'aria del medico di famiglia. Se ne va. Il bug riappare, e ti sorride.

I programmatori hanno perfino un nome per questi errori timidi: li chiamano bug che cambiano comportamento quando li osservi. Io ho risolto alla radice. Ho stampato una foto del collega a grandezza naturale e l'ho appesa dietro la sedia. Funziona nel sessanta per cento dei casi.

LEGGE DELLA RIGA INNOCENTE

SOSPETTATI, IN ORDINE

1. il codice del collega
2. la libreria esterna
3. il sistema operativo
4. i raggi cosmici
5. la mia riga (impossibile)

Sei ore di indagini. Il colpevole era il presidente della commissione.

Il bug si trova sempre nell'ultima riga che controlli, ed è quella che eri assolutamente sicuro fosse corretta.

— Anonimo, *folklore informatico*

La caccia al bug segue un rituale preciso. Si sospetta prima il codice dei colleghi, poi la libreria esterna, poi il sistema operativo, poi il compilatore, infine i raggi cosmici. Solo all'ultimo, con riluttanza, si guarda la propria riga. Quella perfetta. Quella scritta con orgoglio.

Ed eccolo lì, l'errore. Un punto e virgola fuori posto, un uguale al posto di un doppio uguale. Sei ore di indagini per un carattere. Un'intera inchiesta parlamentare per scoprire che il colpevole era il presidente della commissione.

C'è anche una spiegazione logica, che rovina la poesia: smetti di cercare quando lo trovi, quindi è sempre nell'ultimo posto in cui guardi. Ma è inutile dirlo a un programmatore alle due di notte. Lui sa che il codice lo sta prendendo in giro. E, sinceramente, ha ragione.

LEGGE DI KERNIGHAN



Fare il debug è due volte più difficile che scrivere il codice. Quindi, se scrivi il codice nel modo più furbo possibile, per definizione non sei abbastanza furbo da correggerlo.

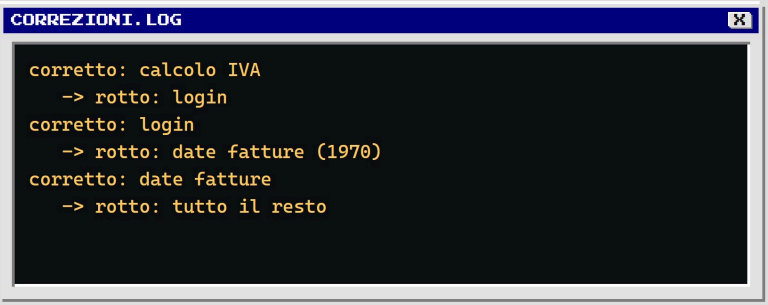
— Brian Kernighan, *The Elements of Programming Style (con P. J. Plauger), 2ª ed., 1978*

È un sillogismo spietato. Alle due di notte, ispirato, scrivi una riga che fa dodici cose insieme. Ti senti un genio galattico. Il mattino dopo la riga non funziona e il genio della notte non risponde al telefono.

Il messaggio vero è: scrivi semplice. Il codice furbo è un debito con il te stesso del futuro, che sarà più stanco, più vecchio e con meno caffè.

Io ho risolto con metodo: scrivo il codice al 50% della mia intelligenza, così il debug mi riesce al 100%. I colleghi dicono che si vede. Lo prendo come un complimento all'ingegneria, non a me.

LEGGE DELLA CORREZIONE PROLIFICA



Il software è a vasi comunicanti: spingi di qua, esce di là. Sul pavimento.

Ogni bug corretto in fretta ne genera almeno due nuovi, possibilmente in una parte del programma che non c'entrava niente.

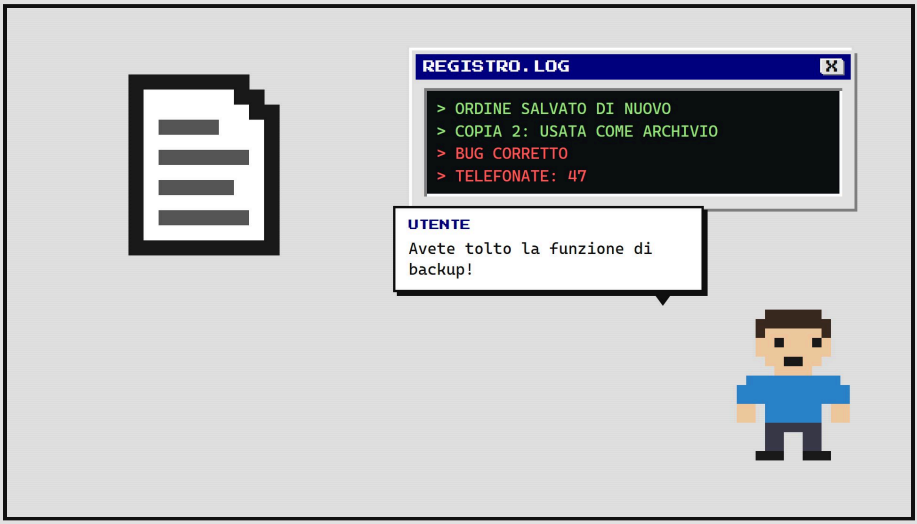
— Anonimo, *folklore informatico*

I bug sono come le teste dell'idra delle leggende: ne tagli una e ne spuntano due. Con una differenza: le teste dell'idra almeno spuntavano nello stesso punto. I bug nuovi preferiscono comparire dall'altra parte del programma, dove nessuno li cerca.

Correggi il calcolo dell'IVA e smette di funzionare il login. Sistemi il login e le fatture escono con la data del 1970. Il software è un sistema di vasi comunicanti: spingi da una parte, esce dall'altra, di solito sul pavimento.

Il guru della qualità, in questi casi, pronuncia la parola magica: «test automatici». Ha ragione. Ma il bug urgente arriva sempre quando i test automatici sono «quasi pronti». Io affronto la correzione veloce con lo stesso spirito con cui si cambia una lampadina in una casa antica: con fiducia, e con l'estintore vicino.

LEGGE DEL BUG EVOLUTO



Qualsiasi bug sufficientemente avanzato è indistinguibile da una funzionalità.

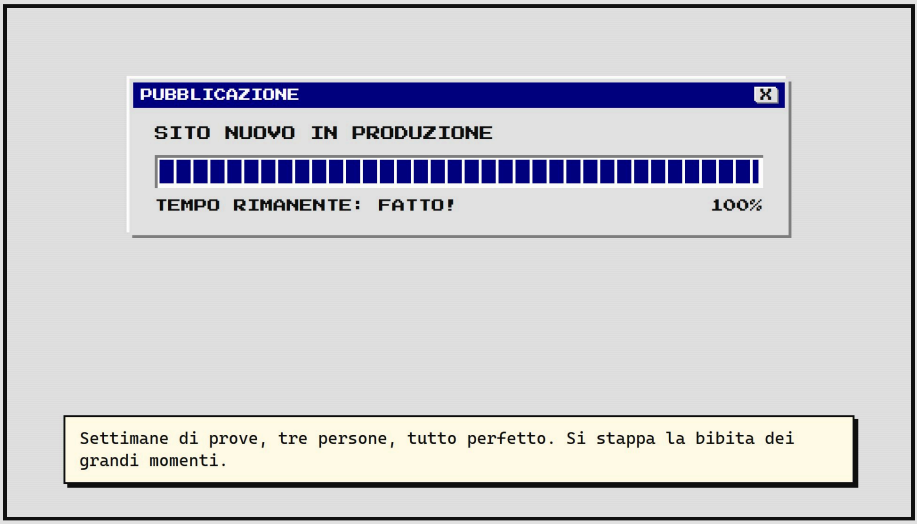
— Anonimo, *folklore informatico*

Ogni tanto un errore sopravvive così a lungo da mettere radici. Gli utenti lo scoprono, ci si abituano, lo trovano comodo. A quel punto correggerlo diventa un atto di violenza.

Esempio vero, raccontato con il dovuto rispetto: un gestionale che, per uno sbaglio, salvava la bozza dell'ordine due volte. Gli impiegati avevano imparato a usare la seconda copia come archivio. Quando l'abbiamo corretto, hanno chiamato in massa: «Avete tolto la funzione di backup».

Da qui nasce la celebre frase che ogni programmatore pronuncia almeno una volta nella vita, con il mento alto e lo sguardo verso l'orizzonte: «Non è un bug, è una funzionalità». È una formula magica, come l'abracadabra. Non risolve nulla, ma trasforma la vergogna in marketing. E il marketing, si sa, non si corregge.

LEGGE DEL BUG PUNTUALE



Settimane di prove, tre persone, tutto perfetto. Si stappa la bibita dei grandi momenti.

Il bug critico viene scoperto dieci minuti dopo la pubblicazione, mai dieci minuti prima.

— Anonimo, *folklore informatico*

Per settimane il programma viene provato, riprovato, controllato da tre persone diverse. Tutto bene. Poi si preme il pulsante «pubblica», si stappa la bibita gassata dei grandi momenti, e in quel preciso istante qualcuno dice: «Ehm, scusate…».

Non è sfortuna, è un fenomeno quasi fisico. Finché il codice è in prova, la mente è in modalità difensiva e non vede nulla. Appena il codice è pubblico e irreversibile, la mente si rilassa e diventa improvvisamente geniale. Vede tutto. Troppo tardi, ma vede tutto.

Ho provato a ingannare il destino: faccio finta di pubblicare, aspetto dieci minuti, poi pubblico davvero. Ha funzionato due volte. La terza volta il bug l'ho trovato nel decimo minuto della finta pubblicazione e l'ho festeggiato come un gol. Poi è arrivato il secondo bug, quello vero.



Una modifica messa in produzione il venerdì pomeriggio si romperà dopo le sei di sera, quando tutti i colleghi sono già in autostrada.

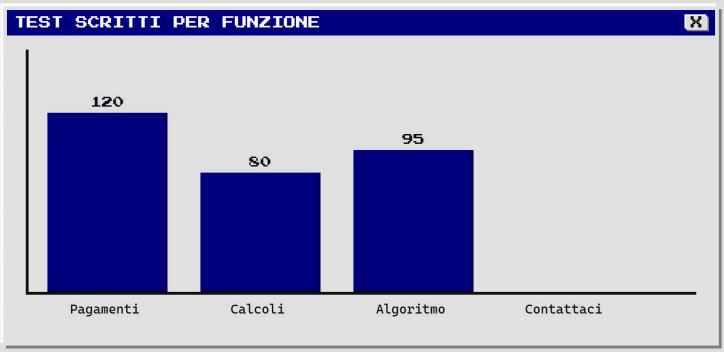
— Anonimo, *folklore informatico*

Nel calendario dell'informatico il venerdì pomeriggio è una zona sacra e maledetta, come le terre di confine nelle saghe fantasy. Chi pubblica un aggiornamento in quelle ore sfida gli antichi dèi della produzione, e gli antichi dèi non perdonano.

La dinamica è sempre la stessa. Alle quattro e mezza: «È una modifica piccolissima, una riga sola». Alle cinque: tutto verde, tutto perfetto. Alle sei e dieci: il sito mostra la home page in cinese e il carrello regala la merce.

Nel frattempo, in autostrada, il collega che conosce il sistema ha il telefono in modalità aereo per motivi spirituali. Da noi in agenzia vige un decreto non scritto: il venerdì si fanno solo cose reversibili. Spolverare la tastiera, leggere la documentazione, pensare. Soprattutto pensare, che non va mai in produzione.

LEGGE DELLA PARTE NON TESTATA



Una fortezza sui pagamenti. La porta sul retro chiusa con lo spago.

La parte del programma che si rompe in produzione è sempre quella che nessuno aveva pensato di provare, perché era troppo semplice per sbagliare.

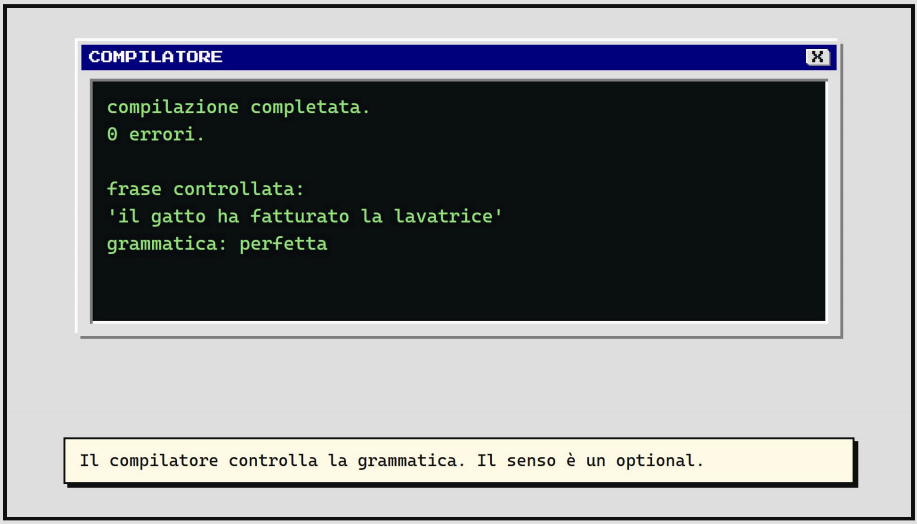
— Anonimo, *folklore informatico*

I test si concentrano sempre sulle parti difficili: calcoli complessi, algoritmi, pagamenti. Su quelle si costruiscono fortezze. Poi il sito crolla per colpa del pulsante «Contattaci», che nessuno aveva mai cliccato perché «tanto è solo un modulo».

È la versione informatica del castello medievale con mura spesse tre metri e la porta sul retro chiusa con lo spago. Il nemico non scala le mura. Il nemico, molto semplicemente, bussa dietro.

Il santone della qualità ripete da anni lo stesso sermone: «Ciò che non è testato è rotto, solo che ancora non lo sai». È una frase che fa paura, perché è vera. Io ora provo anche le cose banali, una per una, con lo zelo di un notaio. Il cliente mi chiede perché ci metto tanto. Gli rispondo che sto controllando la porta sul retro.

LEGGE DEL COMPILA DUNQUE FUNZIONA



Se il codice si compila senza errori, il programmatore stanco lo considererà pronto per la consegna.

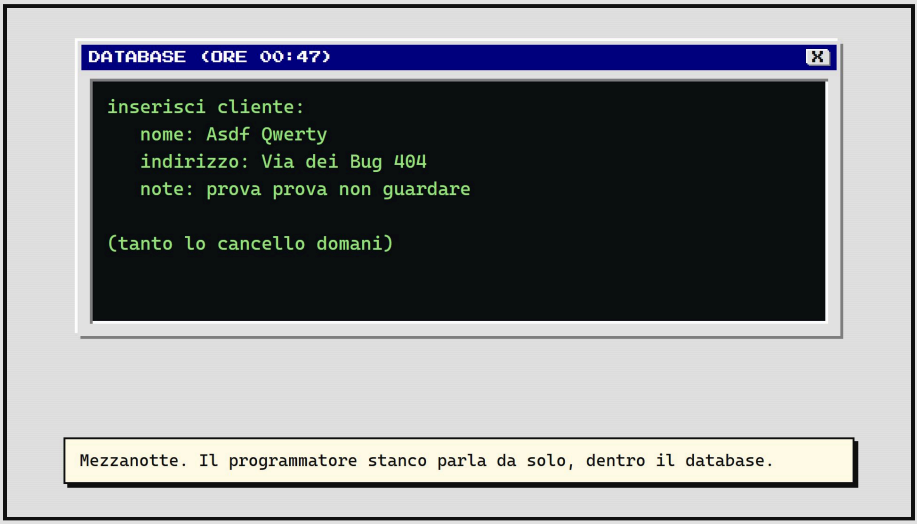
— Anonimo, *folklore informatico*

Compilare significa tradurre il codice in un programma eseguibile. Il compilatore controlla che la grammatica sia corretta, non che il programma abbia senso. È come un correttore di bozze che approva con entusiasmo la frase «il gatto ha fatturato la lavatrice».

Eppure, alle sette di sera, davanti alla scritta «compilazione completata», qualcosa scatta nel cervello umano. Una voce interiore dice: è fatta. Il codice parte verso il cliente come una nave che lascia il porto con le vele montate al contrario.

In molti uffici circola la versione abbreviata del motto: «Compila? Si spedisce». La pronunciano ridendo. Poi la mettono in pratica. Io l'ho fatto una volta sola, nel lontano passato. La lavatrice, in senso figurato, sta ancora girando. E io ho imparato a provare il programma prima, almeno una volta.

LEGGE DEL DATO DI PROVA



Il dato di prova più imbarazzante inserito durante lo sviluppo apparirà, prima o poi, davanti al cliente più importante.

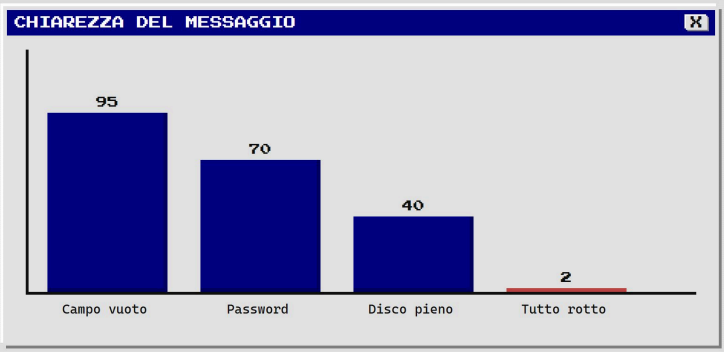
— Anonimo, *folklore informatico*

Durante lo sviluppo nessuno inserisce dati realistici. Si scrive «asdf», «prova prova», «Mario Pippo», «prodotto che non esiste». Sono i segnaposto del programmatore stanco, il suo modo di parlare da solo a mezzanotte.

Poi arriva la presentazione ufficiale. Sala riunioni, proiettore, direttore generale in prima fila. Si apre l'elenco clienti e in cima, in grassetto, c'è il signor Asdf Qwerty, con l'indirizzo «Via dei Bug 404». Silenzio. Qualcuno tossisce.

La regola che ho imparato a mie spese è semplice: scrivi ogni dato di prova come se il cliente lo dovesse leggere. Perché lo leggerà. Oggi nei miei ambienti di test ci sono solo nomi sobri e rispettabili, tipo «Cliente Esempio Srl». Un po' più triste, ma molto meno sudore.

LEGGE DEL MESSAGGIO ORACOLARE



Più l'errore è grave, meno parla. Come un oracolo con poco credito.

Più un errore è grave, più il messaggio che lo annuncia è vago, generico e incomprensibile.

— Anonimo, *folklore informatico*

Gli errori piccoli parlano chiaro: «Il campo Email non è valido». Gli errori catastrofici, invece, si esprimono come gli oracoli antichi: «Si è verificato un errore imprevisto». Codice 0x8000FFFF. Fine della comunicazione.

Imprevisto da chi, poi? È una frase scritta da un programmatore che, anni prima, aveva previsto benissimo che qualcosa sarebbe andato storto, ma non aveva voglia di dire cosa. È un biglietto lasciato sul frigo: «È successa una cosa. Ciao».

I clienti me li leggono al telefono con la voce di chi recita una profezia. Io ascolto in silenzio, annuisco, e poi pronuncio la formula sacra che ogni sistemista conosce: «Ha provato a riavviare?». Non sempre funziona. Ma è l'unica risposta all'altezza della domanda.

LEGGE DEL CALENDARIO TRADITORE

DATE CHE IL PROGRAMMA ODIA

- 29 febbraio
- la notte del cambio dell'ora
- 1 gennaio
- 30 febbraio (inventata da lui)

Trecentosessantadue giorni perfetti. Negli altri tre, romanzo gotico.

Qualunque programma che gestisce date funzionerà tutto l'anno, tranne il 29 febbraio, la notte del cambio dell'ora e il primo giorno di gennaio.

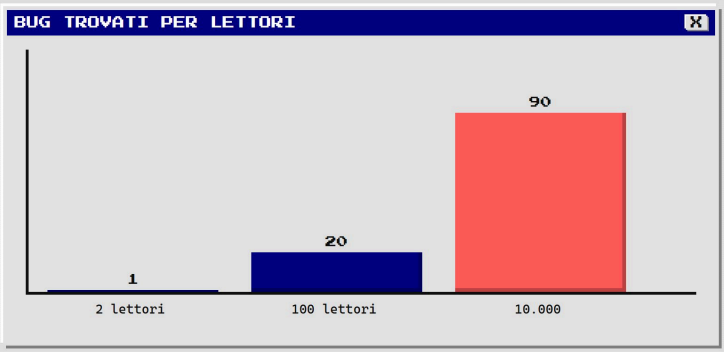
— Anonimo, *folklore informatico*

Il tempo sembra una cosa semplice. Sessanta secondi, sessanta minuti, ventiquattro ore. Poi un programmatore prova a gestirlo nel codice e scopre l'orrore: anni bisestili, fusi orari, ora legale, mesi da ventotto giorni, Paesi che cambiano idea.

Il risultato è un software che funziona benissimo trecentosessantadue giorni l'anno. Negli altri tre fa cose da romanzo gotico: prenotazioni nel passato, buste paga con un'ora fantasma, scadenze fissate al 30 febbraio.

Una notte di fine ottobre un mio gestionale ha registrato la stessa timbratura due volte, alle due e mezza, perché quell'ora era esistita due volte davvero. Il dipendente risultava presente in due posti contemporaneamente. Il direttore voleva promuoverlo. Io ho corretto il bug, con il rammarico di chi ha distrutto un miracolo.

LEGGE DI LINUS



Codice aperto e tanti lettori: prima o poi il difetto salta fuori.

Con abbastanza occhi che guardano, tutti i bug vengono a galla.

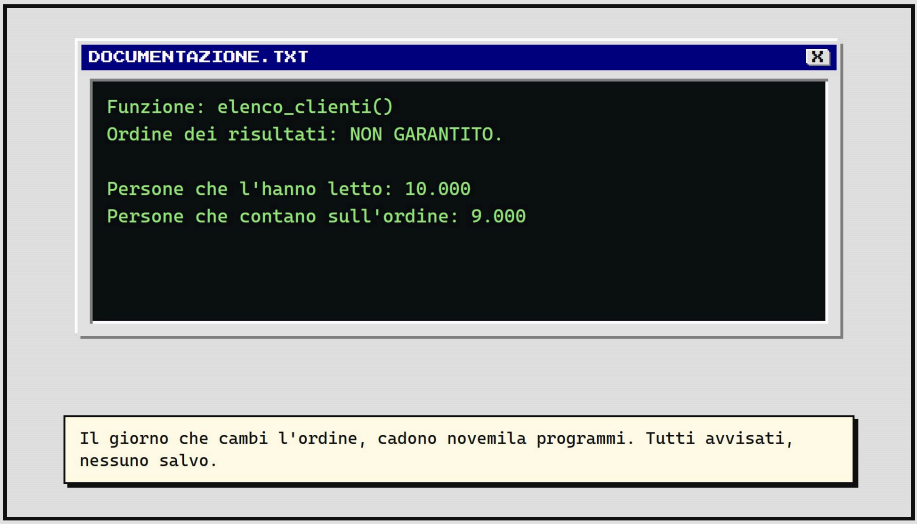
— Eric S. Raymond, *The Cathedral and the Bazaar*, 1997 (saggio), 1999 (libro); *battezzata in onore di Linus Torvalds*

L'idea è bellissima: se il codice è aperto e migliaia di persone lo leggono, qualcuno troverà il difetto. È il motivo per cui il software libero può competere con le grandi aziende, e spesso vincere.

Il dettaglio scomodo è la parola «abbastanza». Gli occhi devono anche guardare davvero. Esistono librerie usate da mezzo pianeta e lette, nei fatti, da due volontari stanchi e un gatto.

Ho provato ad applicarla in ufficio: ho proiettato il codice sul muro durante la pausa caffè. Venti occhi. Nessun bug trovato, ma tre colleghi hanno segnalato che il font era brutto e uno ha chiesto se il proiettore era nuovo. Il bug è ancora lì, ben illuminato.

LEGGE DI HYRUM



Con abbastanza utenti, qualsiasi comportamento osservabile del tuo sistema verrà usato da qualcuno, indipendentemente da ciò che hai promesso nella documentazione.

— Hyrum Wright, *Osservazione nata nell'ingegneria del software di Google, anni 2010 (hyrumslaw.com)*

Scrivi nella documentazione: «L’ordine dei risultati non è garantito». Diecimila persone leggono. Novemila costruiscono il loro programma assumendo che l’ordine sia sempre quello. Il giorno che lo cambi, cade mezzo Internet.

La legge ci dice che il contratto vero di un software non è quello scritto, ma tutto ciò che fa, compresi i difetti. Correggere un bug può rompere chi su quel bug ci aveva costruito un’attività.

Una volta ho sistemato un errore di battitura in un messaggio di errore. Il giorno dopo mi ha chiamato un cliente furioso: il suo script cercava proprio quella parola sbagliata per capire che qualcosa era andato storto. Ho ripristinato l’errore. Ora è patrimonio storico.

IL CONSIGLIO DEL SISTEMISTA

La caccia al bug non è un'arte divinatoria, è un metodo. Il mio, collaudato su parecchie notti, è questo.

- **Prima riprodurlo.** Scrivete i passi esatti che fanno comparire l'errore. Un bug che non sapete far ricomparire non saprete nemmeno dire se l'avete corretto.
- **Leggere i log.** Spesso il programma ha già scritto da qualche parte cosa è andato storto. Sembra ovvio: è il passo che si salta più spesso.
- **Dividere a metà.** Se ieri funzionava e oggi no, cercate a metà strada cosa è cambiato. Con Git esiste perfino un comando apposta, git bisect, che trova la modifica colpevole dimezzando ogni volta il campo.
- **Spiegarlo ad alta voce.** A un collega, o a un oggetto sulla scrivania. Raccontando il problema riga per riga, la riga sbagliata spesso si denuncia da sola.

E quando lo trovate, scrivete un test che lo ricatturi se torna. I bug tornano sempre a casa, come i gatti.

AVVISO

Questa pagina è stata lasciata intenzionalmente vuota.

Anche questa è una legge di Murphy: la pagina vuota capita sempre dove non serve.

CAPITOLO 6

CODICE

L'arte di scrivere codice (e di pentirsene)

Se il capitolo precedente era la cronaca nera, questo è il trattato di filosofia. Le leggi che seguono non parlano di singoli errori, ma del modo in cui si scrive il software: come si dà il nome alle cose, quanto si commenta, quando si ottimizza, perché il codice furbo di oggi è il mal di testa di domani.

Molte portano la firma di grandi nomi della programmazione, persone che hanno costruito linguaggi, sistemi e libri su cui hanno studiato generazioni. È rassicurante scoprire che anche loro, a un certo punto, si sono trovati davanti a un programma scritto da se stessi qualche mese prima senza capirci niente. Il genio non vaccina.

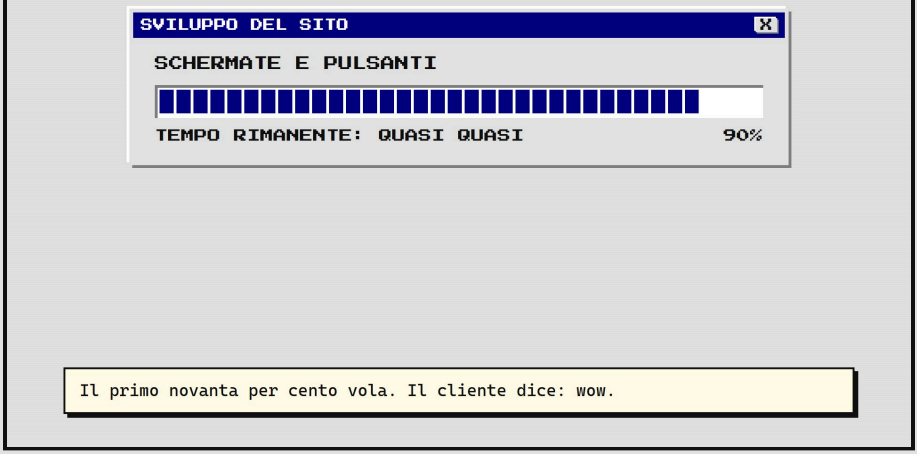
Il filo comune è uno solo, e lo dico subito così chi ha fretta può saltare alla fine: la semplicità è difficile, ma è l'unica cosa che regge nel tempo. Il resto è commento. Bugiardo, probabilmente.

Per chi non programma: niente paura. Ogni legge è spiegata con esempi di tutti i giorni, e vi servirà a capire perché il vostro gestionale ci mette due anni a spostare un pulsante. Non è pigrizia. È archeologia.

CONTENUTO.DIR

01 Regola del novanta-novanta
02 Legge delle due cose difficili
03 Legge dei due problemi
04 Legge di Eagleson
05 Legge del commento bugiardo
06 Legge della documentazione postuma
07 Legge del codice adottato
08 Legge di Osborn
09 Assioma di Flon
10 Principio di Hoare
11 Ottimizzazione prematura
12 Regole di Pike
13 Legge di Demetra
14 Legge delle astrazioni che perdono
15 Decima regola di Greenspun
16 Legge di Wadler
17 Leggi di Lehman

REGOLA DEL NOVANTA-NOVANTA



Il primo 90% del codice richiede il 90% del tempo di sviluppo. Il restante 10% richiede l'altro 90%.

— Tom Cargill, *Bell Labs; resa nota da Jon Bentley nella rubrica Programming Pearls, Communications of the ACM, 1985*

La matematica dice 180%. Il programmatore dice: esatto. Il progetto è sempre «quasi finito» per un periodo che dura quanto il progetto intero.

Il primo 90% è la parte bella: schermate, pulsanti, il cliente che dice «wow». L'ultimo 10% è quando scopri che il cognome può contenere un apostrofo, che qualcuno è nato il 29 febbraio e che l'IVA cambia proprio la settimana del rilascio.

Nel mio ufficio c'è un sito al 98% dal 2017. Ogni anno festeggiamo il compleanno del «manca poco». Abbiamo una torta. Le candeline sono al 98%: accese a metà, ma con grande entusiasmo.

LEGGE DELLE DUE COSE DIFFICILI

LISTINO.PHP

```
prezzo = cache.leggi('caffè')
> 1,00 euro
(il listino nuovo dice 1,20)
> svuota cache
> 1,00 euro
(la cache ci è affezionata)
```

La cache ricorda tutto. Soprattutto quello che dovrebbe dimenticare.

In informatica esistono solo due problemi davvero difficili: invalidare la cache e dare il nome alle cose.

— Phil Karlton, *massima attribuita a Karlton ai tempi di Netscape, riportata da Tim Bray e da Martin Fowler («TwoHardThings»)*

Chi non programma pensa che le parti difficili siano la matematica, la sicurezza, l'intelligenza artificiale. Sbagliato. Le due vere montagne sono queste: far dimenticare al computer una cosa vecchia, e decidere come chiamare una cosa nuova.

La cache è la memoria rapida che tiene da parte i dati per non andarli a riprendere ogni volta. Funziona benissimo, finché ti mostra con orgoglio il prezzo di ieri. Il nome, invece, è una tortura filosofica: «dati», «dati2», «datiNuovi», «datiNuoviVeri». Ho visto team interi bloccati un pomeriggio su un nome di variabile, come un concilio medievale.

Esiste anche una variante folkloristica, che aggiunge un terzo problema: gli errori di uno in più o in meno nei conteggi. Le cose difficili sono due, ma nessuno è mai riuscito a contarle giuste.

LEGGE DEI DUE PROBLEMI

CONTROLLO_EMAIL

```
regola = /^[a-z.]+@+[a-z]*\.it$/

prova('pippo@@.it')        -> VALIDA
prova('studio@commercialista.com')
                           -> RIFIUTATA
```

Una formula per riconoscere le email. Accetta pippo, rifiuta il commercialista.

Davanti a un problema, alcune persone pensano: «Lo risolvo con le espressioni regolari». A quel punto i problemi sono due.

— Jamie Zawinski, *messaggio su Usenet del 1997, ricostruito da Jeffrey Friedl sul blog regex.info (2006)*

Le espressioni regolari sono formule per cercare testo, potentissime e scritte in una lingua che sembra un gatto passato sulla tastiera. Servono, per esempio, a controllare se un indirizzo email è valido. Il risultato è una riga di simboli che nessun essere umano sa più leggere dopo dieci minuti, compreso chi l'ha scritta.

Il primo problema era trovare le email. Il secondo è capire perché la formula accetta «pippo@@.it» e rifiuta il commercialista. Il terzo, se ne aggiunge sempre uno, è che l'unico collega che la capiva ha cambiato lavoro.

Io le uso con la cautela con cui si maneggia un incantesimo trovato in soffitta. Funzionano, ma ogni volta che le modifico mi sembra di sentire un tuono in lontananza. E di solito, in effetti, cade il server.

LEGGE DI EAGLESON

VECCHIO_PROGETTO.PHP

```
function f($pippo2) {
  $x = $pippo2 * 3.14 + 7;   // ?
  // da sistemare dopo
  return f2($x, true, true, false);
}
```

Nessun commento, variabili misteriose. Chi è stato?

Il codice che hai scritto tu e che non guardi da sei mesi o più potrebbe benissimo essere stato scritto da qualcun altro.

— Anonimo, *Detto diffuso tra i programmatori; origine e identità di Eagleson non verificate (attribuzione incerta)*

Apri un vecchio file. Chi ha scritto questa cosa? Variabili chiamate «pippo2», nessun commento, una funzione di trecento righe. Controlli l'autore. Sei tu. Te stesso di sei mesi fa, un perfetto sconosciuto con le tue credenziali.

La legge insegna che il primo destinatario della documentazione non è il collega: sei tu nel futuro, che avrai dimenticato tutto. Scrivere codice chiaro è un atto di gentilezza verso uno sconosciuto che ti somiglia molto.

Io ormai tratto il me stesso del passato come un consulente esterno. Gli scrivo email di reclamo. Non risponde mai. Ieri ho trovato un suo commento: «Da sistemare dopo». Dopo quando? Siamo nel dopo. Il dopo sono io.

LEGGE DEL COMMENTO BUGIARDO

SCONTI.PHP

```
// calcola lo sconto del 10%
funzione sconto(ordine):
   manda_mail(direttore)
   sconto = 30%
   // non toccare
```

Il commento racconta un'intenzione antica. Il codice, un incidente moderno.

Quando il commento e il codice non sono d'accordo, probabilmente hanno torto tutti e due.

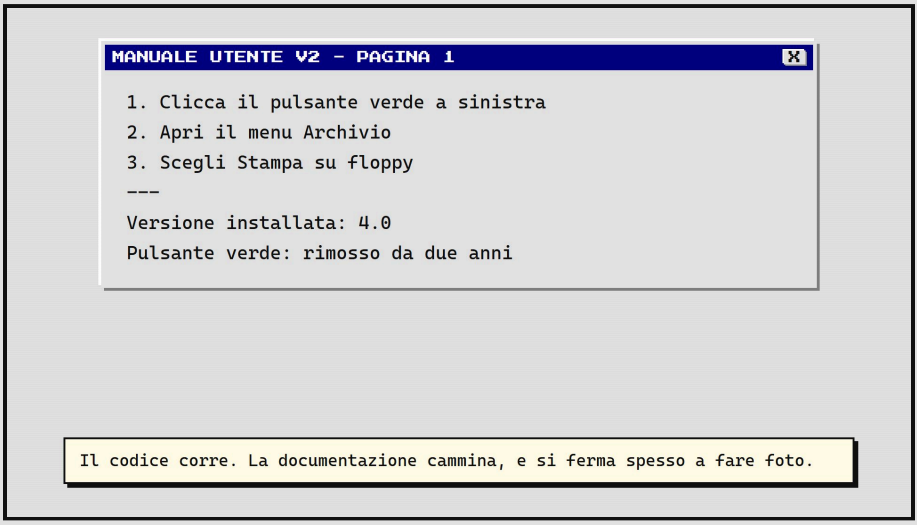
— Anonimo, *folklore informatico*

I commenti sono le frasi che i programmatori lasciano nel codice per spiegare cosa fa. In teoria sono un gesto di gentilezza verso chi verrà dopo. In pratica sono come i cartelli stradali di una strada che nel frattempo è stata spostata.

Il codice viene cambiato, il commento resta. Così trovi scritto «calcola lo sconto del dieci per cento» sopra una funzione che manda una mail al direttore. Chi ha ragione? Nessuno. Il commento racconta un'intenzione antica, il codice un incidente moderno.

Esiste anche la scuola di pensiero opposta, quella dei programmatori fieri che non commentano mai: se è stato difficile da scrivere, deve essere difficile da capire. Io sto in mezzo. Commento sempre, ma con frasi tipo «non toccare». Non spiega niente, ma salva molte vite.

LEGGE DELLA DOCUMENTAZIONE POSTUMA



La documentazione, quando esiste, descrive con grande precisione la versione precedente del programma.

— Anonimo, *folklore informatico*

La documentazione tecnica è un genere letterario a sé, a metà tra il romanzo storico e il testamento. Racconta con dovizia di particolari un mondo che non c'è più: pulsanti spostati, menu rinominati, funzioni eliminate nel frattempo con una riunione di quattro minuti.

Il motivo è semplice. Il codice corre, la documentazione cammina. Quando finalmente qualcuno scrive il manuale della versione due, i programmatori stanno già litigando sulla quattro.

Ho un cliente che conserva il manuale del suo gestionale in una teca, come una reliquia. Ogni tanto lo apre e legge ad alta voce le istruzioni, con tono solenne. Nessuna corrisponde più allo schermo. Ma lui dice che lo aiuta a ricordare com'era il mondo quando i bottoni stavano a sinistra. E io, francamente, lo capisco.

LEGGE DEL CODICE ADOTTATO

SOLUZIONE_DAL_FORUM

```
// trovato su un forum, anno ignoto
// non so perché, ma serve
x = x + 0
aspetta(0 secondi)
ripeti 1 volta:
   funziona()
```

Righe donate da uno sconosciuto. Si accettano con gratitudine e senza domande.

Il codice copiato da Internet funziona alla perfezione finché non provi a capire perché funziona.

— Anonimo, *folklore informatico*

Ogni programmatore ha un archivio segreto di righe prese in prestito dai forum: soluzioni scritte da sconosciuti, in un'altra epoca, per un altro problema. Le incolli, funzionano, e le guardi con la gratitudine di chi riceve un rene da un donatore anonimo.

Il guaio arriva quando, per scrupolo, provi a capirle. Leggi, ragioni, togli una riga che sembra inutile. Crolla tutto. Rimetti la riga. Non funziona più nemmeno con la riga. Il codice adottato è offeso e ha deciso di non perdonarti.

Da allora tratto queste righe come gli antichi trattavano gli amuleti: rispetto, distanza, nessuna domanda. Nel mio codice ci sono ancora pezzi che portano il commento «non so perché, ma serve». Sono le righe più stabili di tutto il progetto. La conoscenza, a volte, è sopravvalutata.

LEGGE DI OSBORN

CONFIG.PHP

```
const ANNO_CORRENTE = 2019;
const ALIQUOTA_IVA = 20;   // poi 21, poi 22
var temperatura = 0;       // mai cambiata

echo ANNO_CORRENTE;
> 2019
```

La variabile è ferma dal primo giorno. La costante ha cambiato idea più volte.

Le variabili non variano; le costanti non restano costanti.

— Anonimo, *Detto diffuso nell'umorismo informatico; origine e identità di Osborn non verificate (attribuzione incerta)*

In teoria la variabile cambia e la costante sta ferma. In pratica la variabile resta inchiodata al valore sbagliato da quando è stata creata, mentre la costante «aliquota IVA» è cambiata quattro volte in dieci anni.

Dietro la battuta c'è una lezione: le cose che dai per immutabili, come tasse, prefissi telefonici, formati delle date, sono quelle che cambieranno. E quelle che dovrebbero aggiornarsi spesso restano bloccate per un errore di inizializzazione.

Nel mio codice c'è una costante chiamata ANNO_CORRENTE. Vale 2019. Nessuno ha il coraggio di toccarla. Abbiamo deciso che, per il software, è ancora il 2019. Si sta bene. I prezzi sono più bassi.

ASSIOMA DI FLON



Non esiste, non è mai esistito e non esisterà mai un linguaggio di programmazione in cui sia difficile scrivere programmi brutti.

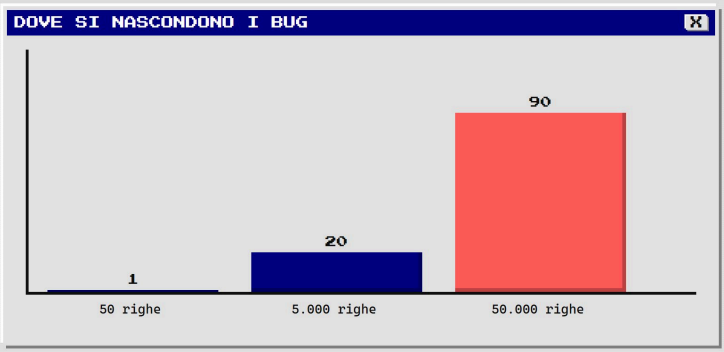
— Lawrence Flon, *On Research in Structured Programming*, *SIGPLAN Notices*, ottobre 1975

Ogni nuovo linguaggio arriva con la promessa messianica: «Con me il codice sarà finalmente pulito!». Dopo tre mesi qualcuno ha già scritto una funzione di novecento righe che si chiama «gestisci_tutto».

Flon non diceva che i linguaggi sono tutti uguali. Diceva che la qualità sta in chi scrive, non nello strumento. Un buon linguaggio aiuta, ma non ti impedisce di costruire un labirinto.

Ho seguito un corso di un santone dei linguaggi moderni. Ci ha mostrato un codice perfetto, elegante, sicuro. Poi ci ha lasciati soli per dieci minuti. Al ritorno avevamo reinventato le variabili globali, il copia-incolla e una cartella chiamata «definitivo_vero». Il linguaggio era innocente. Noi no.

PRINCIPIO DI HOARE



In cinquanta righe il bug è una mosca sulla tovaglia. In cinquantamila ha casa e orto.

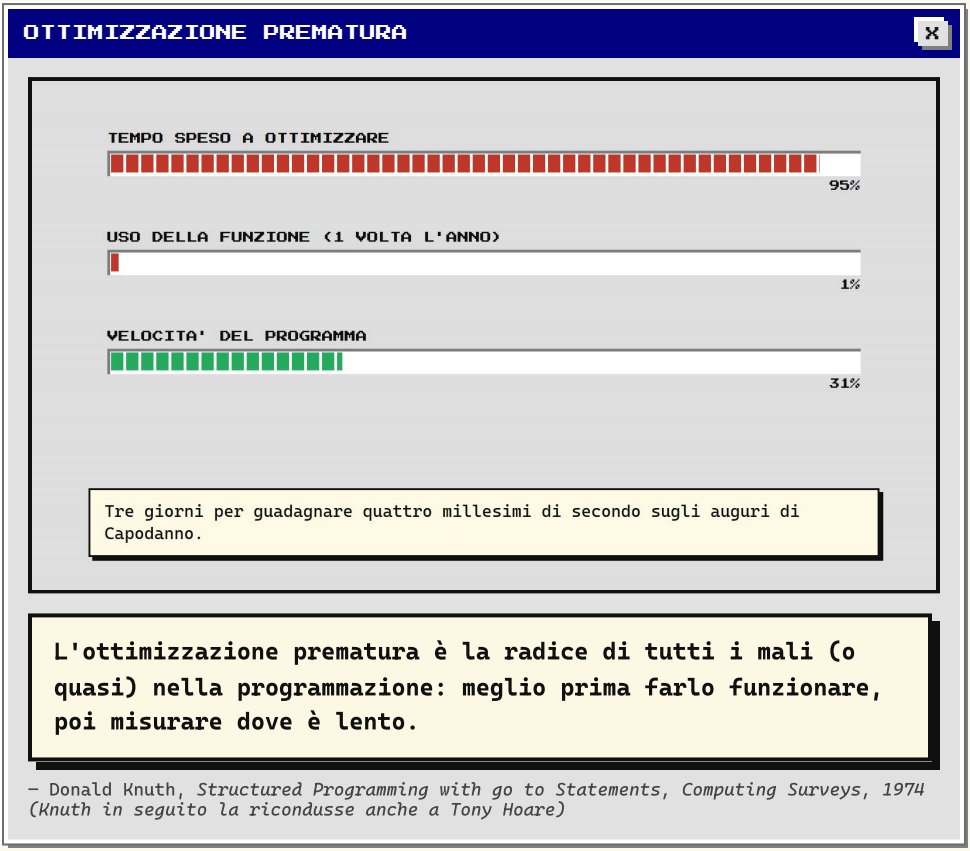
Ci sono due modi di progettare il software: renderlo così semplice che ovviamente non ha difetti, o così complicato che non ha difetti ovvi. Il primo è molto più difficile.

— C. A. R. Hoare, *The Emperor's Old Clothes, discorso per il Premio Turing 1980, pubblicato su Communications of the ACM, 1981*

La complessità è un ottimo nascondiglio. In un programma di cinquanta righe un bug si vede come una mosca sulla tovaglia bianca. In uno da cinquantamila ha casa, famiglia e un orto.

Hoare ci ricorda che la semplicità non è pigrizia: è la cosa più faticosa da ottenere. Bisogna togliere, ripensare, buttare via codice a cui ci si era affezionati. Aggiungere è facile; rinunciare richiede coraggio.

Ho chiesto a un collega se il suo modulo avesse difetti. Mi ha risposto: «Nessuno che si veda». Era vero. Non si vedeva niente. Nemmeno la logica. Era un capolavoro di nebbia lombarda in forma di codice, e sotto, nella nebbia, i bug giocavano a bocce.



C'è un tipo di programmatore che passa tre giorni a rendere velocissima una funzione chiamata una volta all'anno, il primo gennaio, per stampare gli auguri. Guadagna quattro millisecondi. Li festeggia come uno sbarco sulla Luna.

Il senso della frase è di buon senso: non sai dove sta la lentezza finché non misuri. Ottimizzare alla cieca rende il codice illeggibile e lascia intatto il vero collo di bottiglia, che di solito è una query fatta dentro un ciclo.

Mio zio ottimizzò la sua cantina: scaffali in ordine alfabetico, indici, cache delle bottiglie preferite. Poi si accorse che la porta era chiusa e la chiave stava in cantina. Il tempo d'accesso era infinito, ma ben organizzato.

REGOLE DI PIKE

PROFILO.TXT

```
Ipotesi: lento nel calcolo
Misura reale:
  calcolo        2%
  lettura file  97%   (stesso file x 1000)
  altro          1%
```

Prima misura, poi ottimizza. L'intuito sulle prestazioni è un oroscopo.

Non puoi sapere in anticipo dove un programma perderà tempo: prima misura. Gli algoritmi raffinati sono lenti quando i dati sono pochi, e sono i dati, più che il codice, a determinare la struttura giusta.

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

Le regole sono cinque, e suonano come i comandamenti di un monaco del terminale: misura, non indovinare; non usare algoritmi complicati se n è piccolo, e n di solito è piccolo; la semplicità batte l'eleganza; i dati comandano.

Il messaggio è che l'intuizione di un programmatore sulle prestazioni è inaffidabile quanto un oroscopo. Si crede di sapere dove sta la lentezza, ci si mette due giorni a ottimizzare la cosa sbagliata e il vero problema era un ciclo che leggeva lo stesso file mille volte.

Ho implementato un albero bilanciato auto-ottimizzante per gestire la rubrica dell'ufficio. Contiene sette numeri. Il vecchio foglietto attaccato al frigo li trova più in fretta. Il foglietto ha anche un elegante supporto magnetico.

LEGGE DI DEMETRA

ORDINE.JS

```
nome = cliente.ordine.indirizzo
         .città.provincia.regione.nome
// sei anelli: se uno cambia, crolla tutto

meglio: nome = cliente.provincia()
```

Una catena di pettegolezzi: ogni pezzo sa troppo degli altri.

Un oggetto dovrebbe parlare solo con i suoi vicini diretti e non con gli sconosciuti: niente catene di chiamate che attraversano mezzo programma.

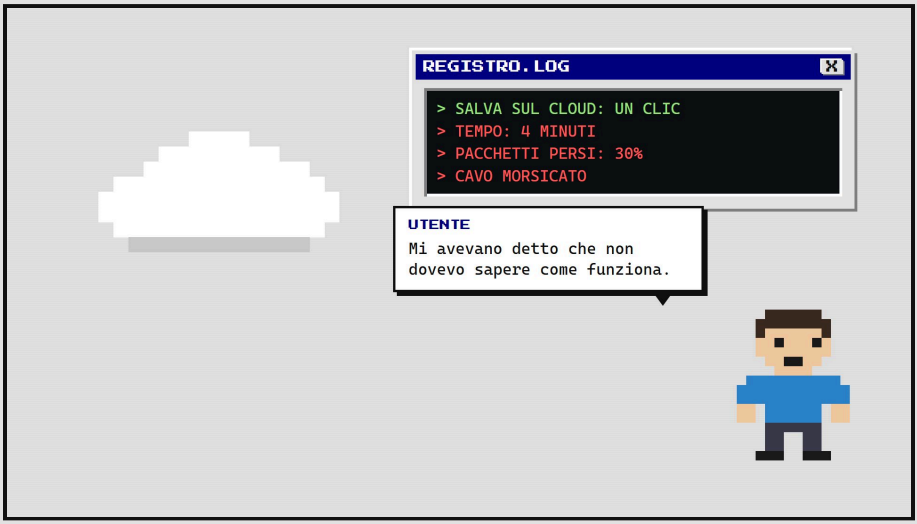
— Ian Holland, *Progetto Demeter, Northeastern University, 1987 (gruppo di Karl Lieberherr)*

È nota anche come principio della minima conoscenza. In pratica: se vuoi pagare il giornalaio, gli dai i soldi. Non gli passi il portafoglio perché lui ci frughi dentro, trovi la carta, chiami la banca e chieda il saldo a tua madre.

Nel codice, una riga come cliente.ordine.indirizzo.città.provincia.nome è un pettegolezzo a catena: ogni anello sa troppo degli altri, e se uno cambia crollano tutti. Ridurre le conoscenze rende i pezzi sostituibili.

Ho applicato la legge in ufficio. Non chiedo più al collega di chiedere al tecnico di chiedere al fornitore. Chiedo al collega e basta. Lui non sa niente, ma il sistema è molto più disaccoppiato. Ora non funziona nulla, ma in modo elegante.

LEGGE DELLE ASTRAZIONI CHE PERDONO



Tutte le astrazioni non banali, prima o poi, lasciano trapelare i dettagli che dovevano nascondere.

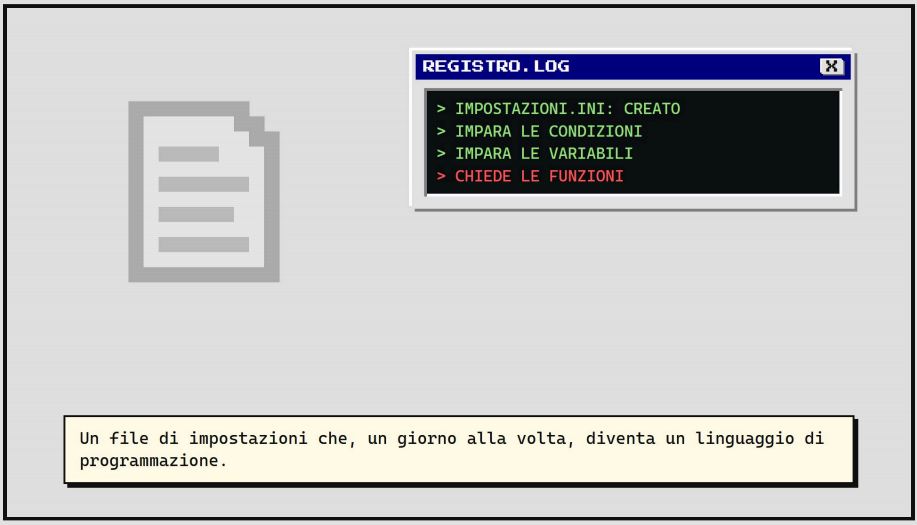
— Joel Spolsky, *The Law of Leaky Abstractions*, *Joel on Software*, 2002

Un'astrazione è una bella facciata: «Non preoccuparti di come funziona la rete, salva il file e basta». Funziona benissimo, fino al giorno in cui il file ci mette quattro minuti e devi scoprire cos'è una latenza, un pacchetto perso e un cavo morsicato da un roditore.

Il punto è che gli strumenti comodi fanno risparmiare tempo nel lavoro normale, ma non ti esonerano dal capire cosa c'è sotto. Quando perdono, e perdono sempre, serve chi sa scendere in cantina.

Il cloud è l'astrazione suprema: «I tuoi dati sono in una nuvola». Poi la nuvola ha un guasto in un datacenter con un nome, un indirizzo e un condizionatore rotto. La nuvola ha un CAP. Lo sapevo. Ci sono stato.

DECIMA REGOLA DI GREENSPUN



Ogni programma abbastanza complicato, scritto in C o Fortran, contiene una versione improvvisata, lenta e piena di bug di metà del linguaggio Lisp.

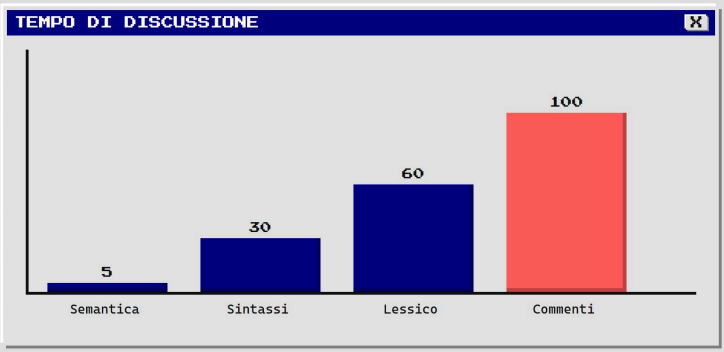
— Philip Greenspun, *Circa 1993; le prime nove regole non sono mai esistite*

Si chiama «decima» ma è l'unica. Le altre nove non sono mai state scritte. Già questo la rende la regola più onesta dell'informatica: promette nove cose e non ne consegna nessuna, come certi preventivi.

Il senso è che i programmi grandi finiscono per reinventare, male, strumenti potenti che esistevano già: un mini-linguaggio di configurazione, un interprete fatto in casa, un sistema di regole scritto in un file di testo che a un certo punto ha anche i cicli.

Nel mio vecchio gestionale c'era un file di impostazioni. Poi ha imparato le condizioni. Poi le variabili. Un giorno ha chiesto di poter dichiarare funzioni. Lo abbiamo spento in tempo. Ora lo teniamo in un cassetto, ma la notte lo sento compilare.

LEGGE DI WADLER



Il cuore del linguaggio: cinque minuti. Il simbolo dei commenti: tre settimane.

Nel progettare un linguaggio, il tempo di discussione cresce man mano che si scende dalla semantica alla sintassi, poi alla sintassi lessicale, fino ai commenti, che vincono su tutto.

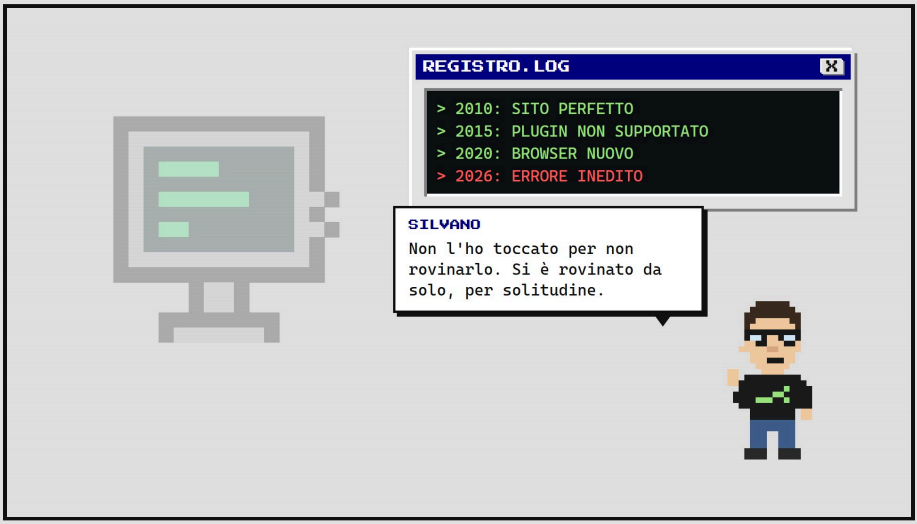
— Philip Wadler, *Messaggi sulla mailing list di Haskell, 1992 e 1996*

Il significato profondo di un linguaggio? Cinque minuti, tutti d'accordo. Il simbolo per i commenti? Tre settimane, due scismi e un fork. Più la questione è piccola, più ognuno si sente esperto.

È la versione da programmatori di una regola umana universale: sulle cose difficili si tace, su quelle facili si combatte. Chiunque abbia partecipato a una revisione del codice ha visto cinquanta commenti sugli spazi e zero sulla logica.

Nel nostro gruppo abbiamo discusso a lungo se usare tabulazioni o spazi. Abbiamo creato un comitato. Il comitato ha prodotto un documento. Il documento era indentato in modo misto. Ora il comitato ha un sottocomitato. Il progetto procede, nei commenti.

LEGGI DI LEHMAN



Un software usato nel mondo reale deve cambiare continuamente, altrimenti diventa sempre meno utile; e ogni cambiamento ne aumenta la complessità, a meno che qualcuno non lavori apposta per ridurla.

— Meir M. Lehman, *Leggi dell'evoluzione del software, formulate con László Bélády a partire dagli anni '70*

Il software è come un giardino, non come una statua. Se lo lasci stare, non resta perfetto: le erbacce crescono da sole. Il mondo intorno cambia, le leggi cambiano, i browser cambiano, e il programma che era impeccabile nel 2010 oggi chiede di installare un plugin estinto.

La parte più crudele è la seconda: ogni modifica aggiunge disordine. Una pezza qui, un caso speciale là, e il codice diventa un condominio con dodici citofoni diversi.

Ho un sito Joomla che non tocco da anni per non rovinarlo. Non l'ho rovinato io. Si è rovinato da solo, per pura solitudine. Ora lo vado a trovare la domenica, gli aggiorno un'estensione e lui mi ringrazia con un errore nuovo.

IL CONSIGLIO DEL SISTEMISTA

Se scrivete codice, anche poco, anche solo per voi, usate un **sistema di controllo delle versioni**. Il più diffuso è Git, ed è gratuito. Funziona come una macchina del tempo con le etichette: ogni volta che il lavoro è in uno stato buono, salvate una fotografia con una frase che spiega cosa avete cambiato.

I vantaggi sono enormi. Tornate a ieri con un comando. Vedete esattamente quale riga è cambiata, e quando. Provate un'idea pazza su un ramo separato e la buttate via senza danni. Addio alle cartelle duplicate con la data nel nome e la parola «vecchio» ripetuta tre volte.

Tre regole d'oro: salvate spesso e a piccoli passi; scrivete messaggi che capirebbe anche uno sconosciuto, cioè voi tra sei mesi; non mettete mai password e chiavi d'accesso nel repository, nemmeno «per un attimo». Quell'attimo, su Internet, dura per sempre.

CAPITOLO 7

RIUNIONI

Progetti, riunioni e clienti

Prima o poi ogni informatico scopre una verità scomoda: i problemi più difficili non stanno nel computer. Stanno nella sala riunioni, nel preventivo, nella mail con dodici persone in copia. Il codice si può correggere. Una stima detta in corridoio, no.

Questo capitolo raccoglie le leggi del lavoro di gruppo. Si parte dalle scadenze, sempre sbagliate per eccesso di ottimismo, e si passa ai clienti, che sanno cosa volevano solo dopo averlo visto. Poi ci sono le riunioni, che si moltiplicano per ricorsione e discutono sempre la cosa meno importante, e le demo, che falliscono in proporzione al grado di chi guarda.

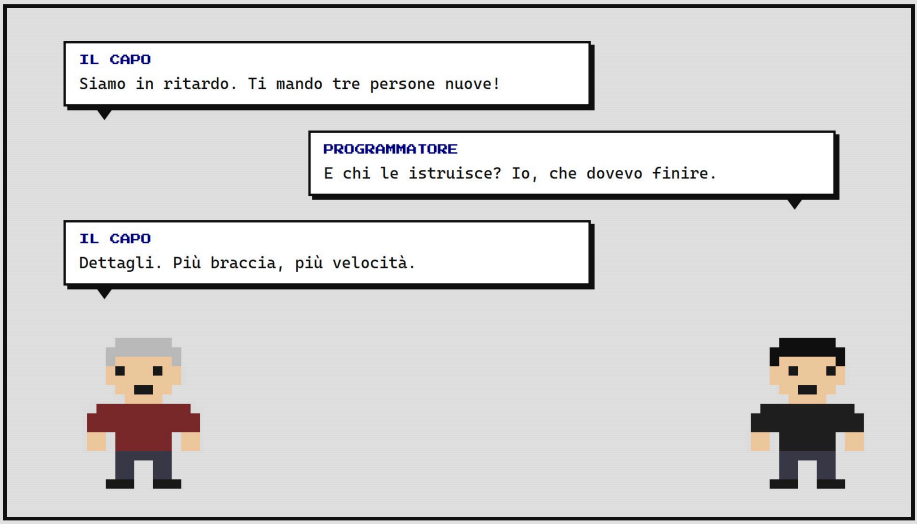
In fondo arrivano le leggi più amare, quelle sulle organizzazioni: i numeri che diventano obiettivi e smettono di dire la verità, le soluzioni provvisorie che reggono l'azienda, i sistemi che finiscono per somigliare all'organigramma, corridoi chiusi compresi.

Chi lavora in un ufficio riconoscerà colleghi, capi e anche se stesso. È normale. Il trucco è riconoscersi prima della riunione, non durante.

CONTENUTO.DIR

01 Legge di Brooks
02 Legge del moltiplicatore pi greco
03 Legge della stima giurata
04 Legge del cliente veggente
05 Legge del requisito fantasma
06 Legge della riunione ricorsiva
07 Legge della banalità di Parkinson
08 Legge di Sayre
09 Leggi di Wiio
10 Legge della demo davanti al cliente
11 Corollario del pubblico importante
12 Legge del provvisorio eterno
13 Legge di Conway
14 Legge di Gall
15 Legge di Weinberg
16 Legge di Goodhart
17 Legge di Campbell
18 Principio di Dilbert

LEGGE DI BROOKS



Aggiungere persone a un progetto software in ritardo lo fa ritardare ancora di più.

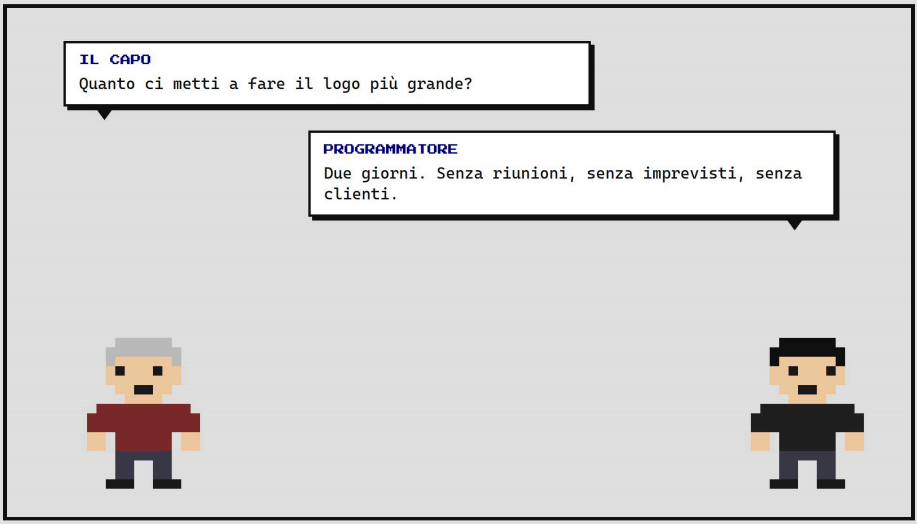
— Fred Brooks, *The Mythical Man-Month*, 1975

Il manager visionario guarda il diagramma di Gantt, vede rosso e proclama: «Raddoppiamo la squadra!». È lo stesso ragionamento per cui nove donne farebbero un figlio in un mese.

Il problema è che i nuovi arrivati vanno istruiti, e a istruirli sono proprio quelli che stavano finendo il lavoro. Poi ogni persona in più moltiplica le conversazioni: in quattro ci si parla, in dodici si fanno riunioni per decidere quando fare le riunioni.

Una volta, per recuperare un ritardo, mi hanno assegnato tre stagisti. Il primo giorno ho spiegato dove sono i bagni. Il secondo, la password del Wi-Fi. Il terzo il progetto era in ritardo di altre due settimane, ma il Wi-Fi funzionava benissimo.

LEGGE DEL MOLTIPLICATORE PI GRECO



Per ottenere il tempo reale di un progetto software, prendi la stima del programmatore e moltiplicala per pi greco. Se il progetto è importante, usa anche le cifre decimali.

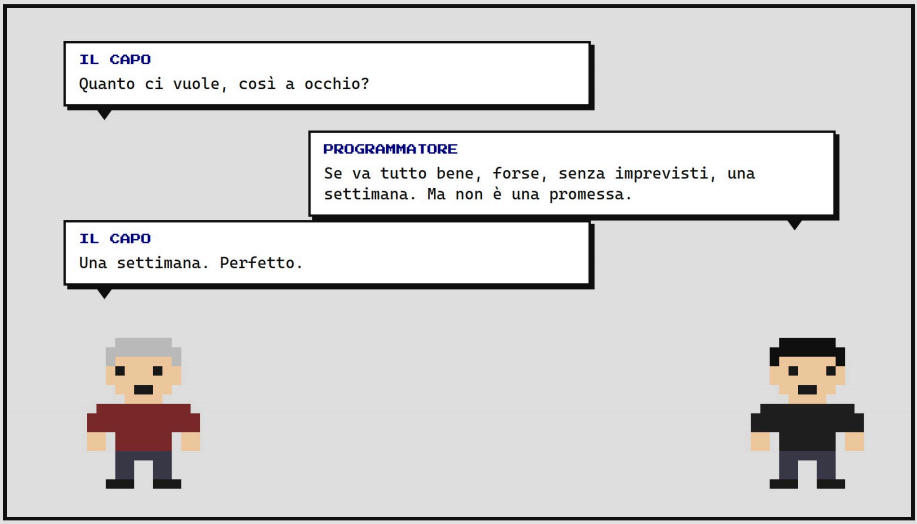
— Anonimo, *folklore informatico*

Il programmatore che stima un lavoro è un ottimista per costituzione. Immagina un mondo senza imprevisti, senza riunioni, senza il cliente che chiama per chiedere se si può avere il logo un po' più grande. In quel mondo ideale, «due giorni» sono due giorni.

Poi arriva la realtà, che ha una sua matematica. Due giorni diventano sei virgola ventotto, e lo zero virgola ventotto è il momento in cui scopri che il server di prova ha la password di qualcuno andato in pensione.

Perché proprio pi greco? Nessuno lo sa. Ma è un numero irrazionale e infinito, quindi è il più adatto a descrivere le scadenze. Io nelle offerte ho smesso di scrivere i giorni. Scrivo direttamente il diametro del progetto, e lascio che il cliente faccia i conti.

LEGGE DELLA STIMA GIURATA



Qualunque stima pronunciata a voce, con mille cautele e tre «forse», diventerà una scadenza scritta nel calendario del capo.

— Anonimo, *folklore informatico*

Il programmatore, interrogato in corridoio, risponde con prudenza da diplomatico: «Così a occhio, se va tutto bene, forse, senza imprevisti, una settimana. Ma non prendetela per buona». Il capo annuisce. Ha sentito solo «una settimana».

Il giorno dopo quella frase compare in un documento ufficiale, poi in una mail al cliente, poi in un grafico colorato proiettato in riunione. Le cautele sono state eliminate in fase di compressione, come i dettagli di una foto mandata col telefono.

Il manager visionario, intanto, la presenta con fierezza: «Il team si è impegnato». Il team non sapeva di essersi impegnato. Da allora, quando qualcuno mi chiede quanto ci vuole, rispondo con una domanda: «È una domanda o è un contratto?». Di solito il silenzio che segue vale come risposta. E come preventivo.

LEGGE DEL CLIENTE VEGGENTE



Il cliente sa esattamente cosa vuole nel preciso momento in cui gli consegni quello che aveva chiesto.

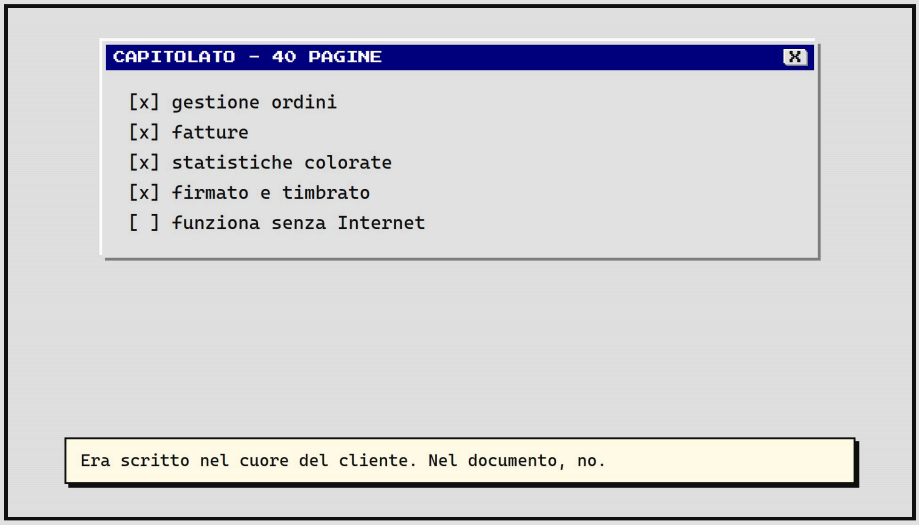
— Anonimo, *folklore informatico*

Durante l'analisi il cliente è vago, pensoso, zen. «Fate voi, siete gli esperti». Poi vede il lavoro finito e ha una rivelazione mistica, come se gli si aprissero i cieli: adesso sa. Sa che non era questo.

Il fenomeno è ben noto anche in letteratura tecnica: c'è un celebre disegno anonimo che mostra un'altalena come la voleva il cliente, come l'ha capita il commerciale, come l'ha costruita il programmatore. Nessuna delle versioni è un'altalena.

Io ho trovato un metodo infallibile. Preparo subito una versione sbagliata, apposta, in mezza giornata. Il cliente la guarda, inorridisce e finalmente parla chiaro. È l'unico caso nell'informatica in cui il bug è la funzionalità più utile del progetto. Lo fatturo come «analisi dei requisiti per contrasto».

LEGGE DEL REQUISITO FANTASMA



Il requisito più importante del progetto viene comunicato a voce, di sfuggita, il giorno della consegna.

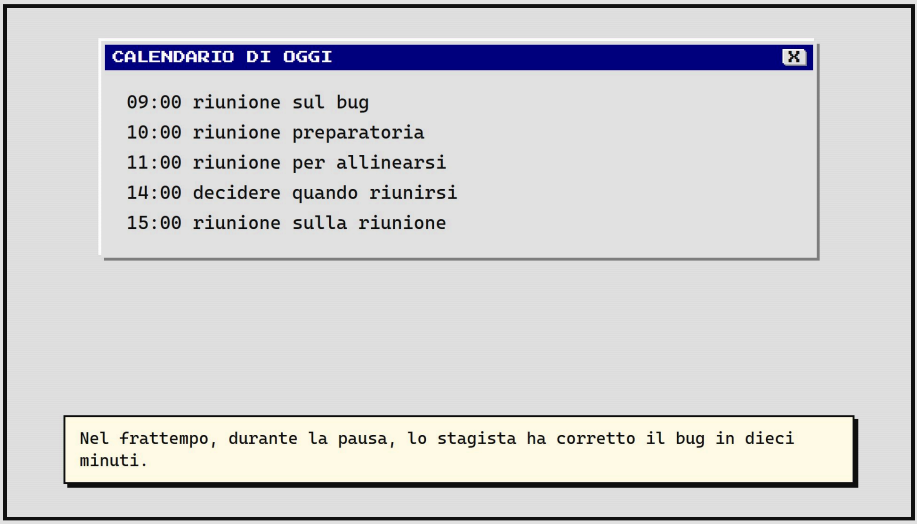
— Anonimo, *folklore informatico*

Il capitolato è lungo quaranta pagine, con tabelle, grafici e un indice che sembra un trattato di pace. Tutto è stato discusso, firmato, timbrato. Poi, il giorno della consegna, mentre si stringono le mani, il cliente aggiunge: «Ah, naturalmente deve funzionare anche senza Internet».

Non lo dice con cattiveria. Lo dice come si dice «buongiorno», con la serenità di chi crede di ricordare un dettaglio ovvio. Per lui è sempre stato così. Era scritto nel suo cuore, se non nel documento.

Da allora ho introdotto in ufficio la riunione della domanda finale. Prima di firmare, chiedo: «C'è qualcosa che pensate sia talmente ovvio da non averlo detto?». Il silenzio che segue è sempre lungo. E dentro quel silenzio, ogni volta, abita un nuovo modulo da sviluppare.

LEGGE DELLA RIUNIONE RICORSIVA



Ogni problema tecnico complesso genera una riunione, che genera una riunione preparatoria, che genera la necessità di una riunione per decidere quando fare la riunione.

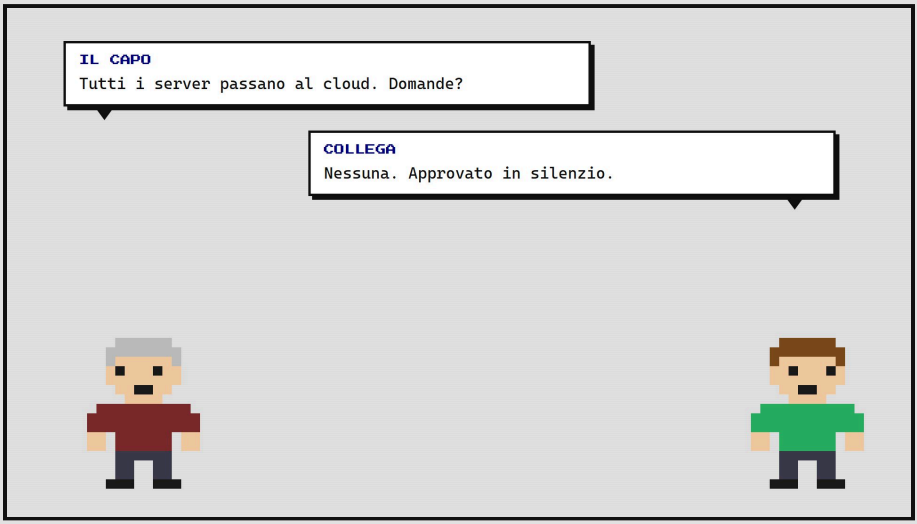
— Anonimo, *folklore informatico*

In informatica la ricorsione è una funzione che chiama se stessa. Nelle aziende è una riunione che convoca se stessa, all'infinito, finché qualcuno non va in ferie e spezza il ciclo per esaurimento del calendario.

Il modello è sempre questo. Si incontrano otto persone per discutere un bug. Si scopre che servono informazioni. Si fissa un nuovo incontro con altre quattro persone. In quell'incontro emerge la necessità di «allinearsi». Nel frattempo, il bug lo ha corretto lo stagista, in dieci minuti, durante la pausa.

C'è sempre il manager che apre i lavori con solennità: «Signori, oggi decidiamo». E infatti si decide: la data della prossima riunione. Io ho proposto di installare un contatore in sala, come quelli delle sagre. Ogni ora di riunione, una riga di codice in meno.

LEGGE DELLA BANALITA DI PARKINSON



Un'organizzazione dedica una quantità di tempo inversamente proporzionale all'importanza delle decisioni: si discute poco del reattore e moltissimo della rastrelliera per le biciclette.

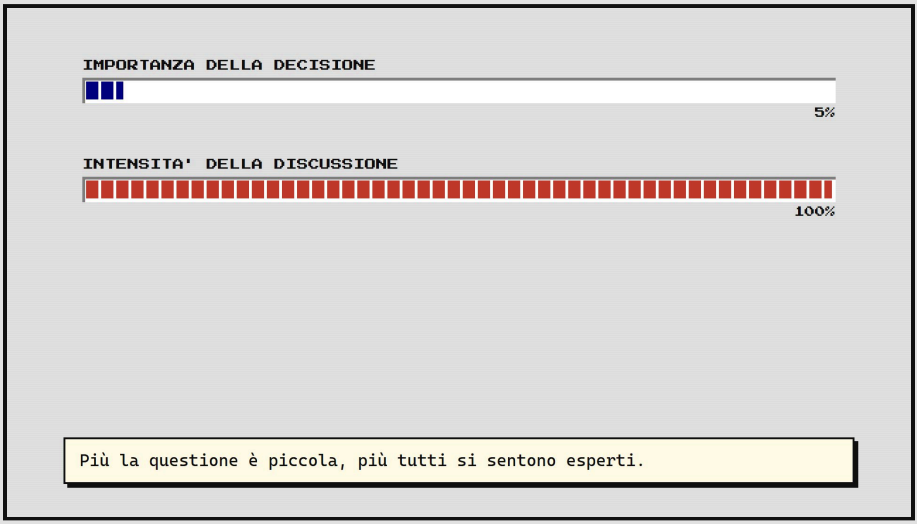
— C. Northcote Parkinson, *Parkinson's Law*, 1957; *resa popolare in informatica come «bikeshedding» da Poul-Henning Kamp nella comunità FreeBSD, 1999*

Il consiglio approva in due minuti il progetto da milioni, che nessuno capisce. Poi passa un'ora sulla tettoia per le bici, che tutti capiscono benissimo e su cui tutti hanno un parere sul colore.

In informatica è diventata una parola d'ordine: quando una discussione si accende su un dettaglio irrilevante, qualcuno scrive «bikeshedding» e tutti, per un attimo, si vergognano. Poi riprendono a discutere del nome della variabile.

Nella mia ultima riunione abbiamo approvato in silenzio il passaggio di tutti i server al cloud. Poi abbiamo litigato quaranta minuti sull'icona del nuovo gestionale: un ingranaggio o una nuvola? Alla fine abbiamo scelto un ingranaggio a forma di nuvola. Nessuno sa cosa fa. Come il cloud, del resto.

LEGGE DI SAYRE



In ogni discussione, l'intensità delle emozioni è inversamente proporzionale all'importanza della posta in gioco.

— Wallace Sayre, *Osservazione sulla politica accademica attribuita a Wallace S. Sayre, metà del Novecento; attribuzione e formulazione variano (attribuzione incerta)*

Nata per descrivere le liti tra professori universitari, si applica perfettamente alle riunioni tecniche. La migrazione dell'intero database passa in cinque minuti. Il colore del pulsante «Annulla» provoca due ore di dibattito e una mail con mezza azienda in copia.

Il motivo è semplice: sulle questioni enormi pochi si sentono competenti; su quelle minuscole tutti hanno un'opinione, e difenderla non costa niente.

La riunione più lunga della mia carriera riguardava il nome di una cartella condivisa. «Documenti», «Documentazione» o «Doc». Quattro ore. Abbiamo votato. Ha vinto «Documenti_DEF». Il giorno dopo è comparsa «Documenti_DEF_nuova». Il conflitto è diventato permanente, come certe guerre fredde, ma con più permessi di scrittura.

LEGGI DI WIIO

UN SITO SEMPLICE, SECONDO:

- il cliente: tre pagine
- il grafico: minimalismo nordico
- il programmatore: niente database
- il commerciale: costa poco

Alla consegna nessuno è soddisfatto. Tutti sicuri di aver chiesto esattamente quello.

La comunicazione di solito fallisce, tranne per caso. E se sembra riuscita come previsto, c'è sicuramente un malinteso.

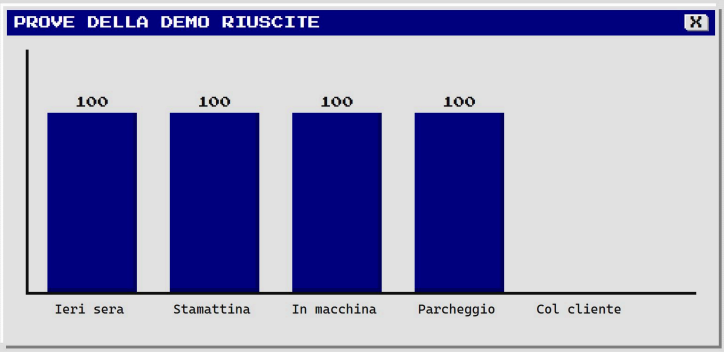
— Osmo Antero Wiio, *«Wiion lait – ja vähän muidenkin»*, 1978

Un professore finlandese, studioso di comunicazione, arrivò a una conclusione che ogni sistemista conosce già: comunicare non funziona. Quando funziona è un incidente. E se ti sembra che tutti abbiano capito, è solo perché ognuno ha capito una cosa diversa.

Il cliente chiede «un sito semplice». Lui intende tre pagine. Il grafico intende minimalismo nordico. Il programmatore intende niente database. Il commerciale intende «costa poco». Alla consegna nessuno è soddisfatto, ma tutti sono sicuri di aver chiesto esattamente quella cosa.

Per questo esistono le specifiche scritte, i verbali, le mail di riepilogo. Servono a documentare con precisione il malinteso. Io ora chiudo ogni riunione chiedendo a tutti di ripetere cosa abbiamo deciso. Ogni volta escono cinque progetti. Almeno lo scopriamo subito, e non al collaudo.

LEGGE DELLA DEMO DAVANTI AL CLIENTE



Percentuale di successo. L'ultima misura è stata presa in presenza di un decisore.

Un programma che ha funzionato in cento prove si guasta alla centounesima: quella davanti al cliente.

— Anonimo, *folklore informatico (noto anche come "effetto demo")*

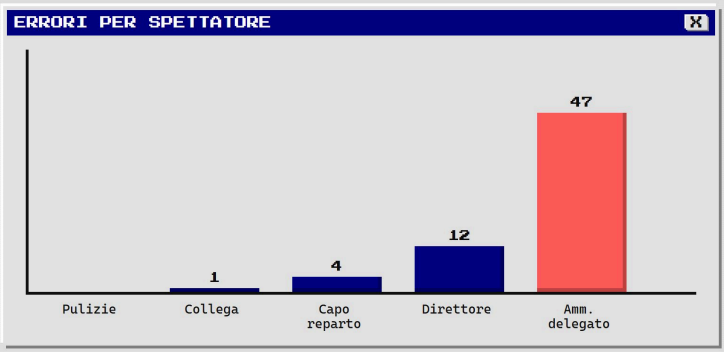
La demo è stata provata la sera prima, la mattina, in macchina e nel parcheggio. Tutto liscio. Poi entri, il cliente si siede, tu dici con sicurezza "Ora vi faccio vedere" e lo schermo diventa una tavolozza di errori mai visti prima.

Gli informatici lo chiamano effetto demo. È l'unico fenomeno noto in grado di rilevare la presenza di un decisore a dieci metri di distanza, attraverso i muri.

Il bello è che appena il cliente se ne va, il programma riparte perfetto. Ti guarda come per dire: "Cosa è successo? Io c'ero."

Consiglio da chi l'ha vissuto: preparate sempre un piano B. Un video registrato della demo funzionante, o screenshot dei passaggi chiave. Se il programma fa i capricci, si mostra il film.

COROLLARIO DEL PUBBLICO IMPORTANTE



Una scala gerarchica, precisa come un organigramma. Con le pulizie il server si calma.

Più è alto il grado di chi assiste alla dimostrazione, più è probabile che la dimostrazione fallisca.

— Anonimo, *folklore informatico (raccolte Usenet di leggi sui computer)*

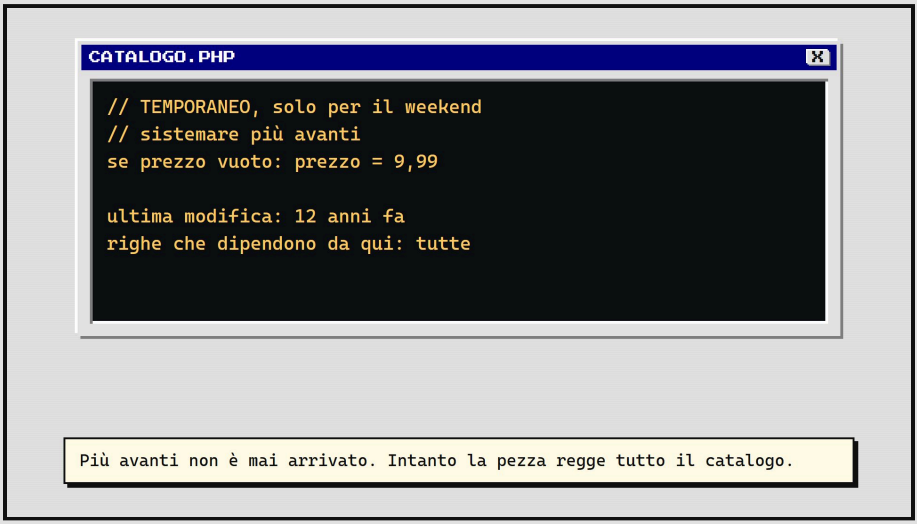
Davanti al collega, la demo regge. Davanti al capo reparto, esita. Davanti al direttore, tossisce. Davanti all'amministratore delegato appena arrivato in elicottero, il computer si spegne e chiede di installare quarantasette aggiornamenti.

È una scala gerarchica dell'errore, precisa come un organigramma. Alcuni ricercatori sostengono che le giacche costose emettano un campo elettromagnetico. Io non ho prove. Ma ho notato che il server si calma quando entra quello delle pulizie.

Il manager visionario, intanto, sorride e dice: “Ottimo, abbiamo trovato un'opportunità di miglioramento.” Che in lingua aziendale significa: ci vediamo nel mio ufficio.

Consiglio vero: per le presentazioni importanti usate un ambiente già preparato e stabile, e fate una prova completa nella stessa sala, con lo stesso computer.

LEGGE DEL PROVVISORIO ETERNO



Nel software non esiste niente di più definitivo di una soluzione temporanea.

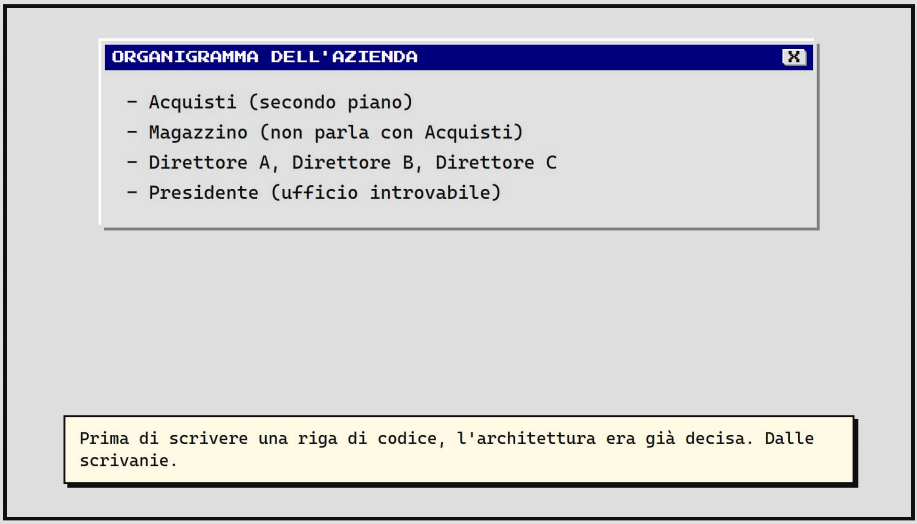
— Anonimo, *folklore informatico*

Ogni programma ha al suo interno una riga accompagnata dalla nota «sistemare più avanti». Più avanti non arriva mai. La riga sopravvive a tre versioni, due cambi di server e un cambio di proprietà dell'azienda.

Le soluzioni provvisorie hanno una tempra straordinaria. Le soluzioni definitive, progettate con cura, vengono riscritte ogni due anni. La pezza messa alle undici di sera per far ripartire le fatture, invece, diventa patrimonio storico, con una sua dignità e una sua aura.

Ho in gestione un sito dove, dodici anni fa, qualcuno ha scritto un piccolo script «solo per il weekend». Oggi regge l'intero catalogo. Nessuno osa toccarlo. Ogni tanto lo guardo, così, con rispetto, come si guarda il muro portante di una cascina: storto, sì, ma se lo togli viene giù tutto.

LEGGE DI CONWAY



Un'organizzazione che progetta un sistema finirà per produrre un sistema che copia la propria struttura di comunicazione.

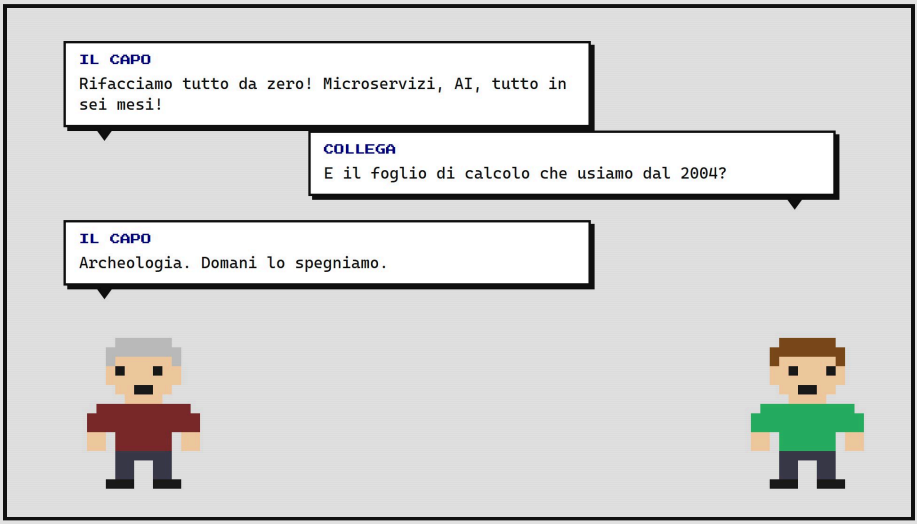
— Melvin Conway, *How Do Committees Invent?*, *Datamation*, 1968

Se quattro gruppi lavorano a un compilatore, uscirà un compilatore in quattro passaggi. Se l'ufficio acquisti non parla con il magazzino, il gestionale avrà due anagrafiche dei prodotti che non si salutano.

Il software è uno specchio. Guardi l'architettura e vedi l'organigramma, con tutti i rancori del caso. Quel modulo che non comunica con nessuno? È il collega del terzo piano che pranza da solo.

Ho visto un sito con tre menu diversi, uno per ogni direttore. Un quarto menu, nascosto, portava alla pagina del presidente. Nessuno sapeva come ci si arrivasse, esattamente come al suo ufficio. L'architettura era perfetta: rispecchiava l'azienda fino all'ultimo corridoio chiuso a chiave.

LEGGE DI GALL



Un sistema complesso che funziona deriva sempre da un sistema semplice che funzionava. Uno complesso progettato da zero non funziona e non si riesce ad aggiustare.

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

Il guru della trasformazione digitale sale sul palco: «Rifaremo tutto da zero, microservizi, intelligenza artificiale, blockchain, in sei mesi!». Applausi. Sedici mesi dopo si torna al foglio Excel condiviso, che funzionava dal 2004.

La legge è un elogio dell'evoluzione. I sistemi che reggono sono cresciuti un pezzetto alla volta, come le stalattiti o le cartelle «Nuova cartella (7)». Quelli nati enormi e perfetti sulla carta crollano, perché nessuno li ha mai visti funzionare piccoli.

Il mio gestionale preferito nacque come un file di testo con tre righe. Oggi gestisce un'azienda intera. Ha ancora quelle tre righe dentro, da qualche parte. Nessuno le tocca. Sono la reliquia portante.

LEGGE DI WEINBERG

PALAZZO E PROGRAMMA A CONFRONTO

Palazzo: fondamenta, calcoli, collaudo
Programma: libreria delle 3 di notte
Manutentore: uno sconosciuto
Ci si appoggiano: banca, ospedale, caffè

L'informatica è giovane: meno norme e meno collaudi dell'edilizia.

Se i costruttori costruissero gli edifici come i programmatori scrivono i programmi, il primo picchio che passa distruggerebbe la civiltà.

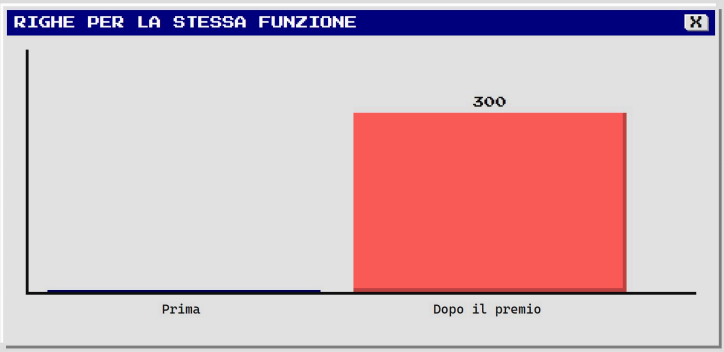
— Gerald Weinberg, *Attribuzione incerta: compare in una raccolta del 1975 (CoEvolution Quarterly) e come «Seconda legge di Weinberg» in Murphy's Law di Arthur Bloch, 1977*

Un palazzo vero ha fondamenta, calcoli strutturali, collaudi. Un programma medio ha una libreria scaricata alle tre di notte, mantenuta da uno sconosciuto, da cui dipendono una banca, un ospedale e il distributore del caffè.

La legge non è solo una presa in giro: ricorda che l'informatica è una disciplina giovane, con meno norme e meno collaudi dell'edilizia. Molto di ciò che usiamo regge per buona volontà e fortuna.

Una volta ho visto un picchio sul tetto del nostro capannone, proprio sopra la sala server. Mi sono messo tra lui e l'antenna. Ci siamo guardati a lungo. Alla fine se n'è andato. Credo abbia capito che non era necessario: il server si sarebbe rotto da solo, il lunedì.

LEGGE DI GOODHART



Si premiano le righe di codice. La produttività triplica. Il software, misteriosamente, no.

Quando una misura diventa un obiettivo, smette di essere una buona misura.

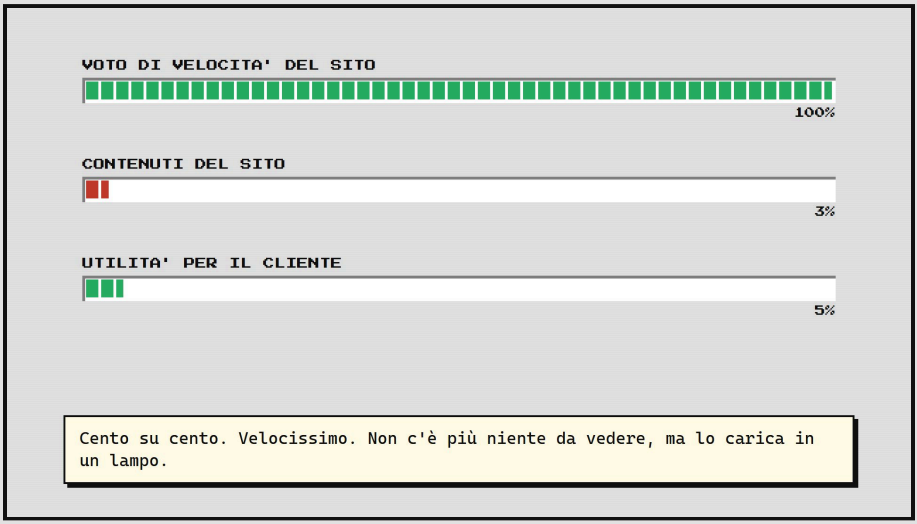
— Charles Goodhart, *Articolo sulla politica monetaria, 1975; la formulazione famosa è di Marilyn Strathern, 1997*

L'azienda decide di misurare la produttività dei programmatori contando le righe di codice. Il mese dopo i programmi sono lunghissimi. Una funzione che stava in tre righe adesso ne occupa trecento, con commenti poetici e variabili dai nomi di quattro parole. La produttività è triplicata. Il software, misteriosamente, no.

Questa è la legge di Goodhart: appena premi un numero, le persone imparano a gonfiare il numero invece di migliorare la cosa che doveva rappresentare. Vale per i ticket chiusi, per le stelline delle recensioni, per i punteggi di velocità dei siti.

Al reparto assistenza misuravano i tempi di chiusura delle chiamate. Un collega le chiudeva tutte in otto secondi. Rispondeva, diceva «provi a riavviare» e riattaccava. Premiato tre volte come dipendente del mese. Gli utenti lo chiamavano il Fulmine.

LEGGE DI CAMPBELL



Più un indicatore numerico viene usato per prendere decisioni importanti, più viene corrotto, e più distorce i processi che dovrebbe sorvegliare.

— Donald T. Campbell, *«Assessing the Impact of Planned Social Change»*, 1976 *(formulazioni precedenti dal 1969)*

Cugina stretta della legge di Goodhart, la legge di Campbell aggiunge un dettaglio cupo: il numero non solo smette di dire la verità, ma comincia a deformare il lavoro stesso. Se il premio dipende dal punteggio, il lavoro diventa produrre punteggio.

Nel mondo dei siti è evidentissimo. Si decide che conta solo il voto dello strumento di velocità. Allora si tolgono le immagini, i testi, le funzioni, tutto ciò che pesa. Il sito ottiene cento su cento. È velocissimo. Non c'è più niente da vedere, ma lo carica in un lampo.

In un'azienda che conosco valutavano l'informatico dal numero di computer funzionanti a fine mese. Lui, molto saggiamente, l'ultimo giorno del mese non ne toccava nessuno. Funzionavano tutti. Lentissimi, pieni di avvisi, ma statisticamente in perfetta salute.

PRINCIPIO DI DILBERT



REGISTRO.LOG

```
> ACCESSO AL SERVER: NEGATO
> NUOVO UFFICIO: CON VISTA
> RUOLO: MANAGER
```

Controllo degli accessi primitivo: al posto dei permessi, un ufficio con la vista.

Le aziende tendono a promuovere i dipendenti meno capaci verso ruoli di gestione, per limitare i danni che possono fare nel lavoro vero.

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

Il principio di Peter diceva che si sale fino a diventare incompetenti. Il principio di Dilbert è più cinico: qualcuno parte già incompetente e viene spostato in alto apposta, dove non può toccare nulla di importante. Come mettere il bambino iperattivo a fare il capitano della squadra, così non tocca il pallone.

Nell'informatica il fenomeno ha un nome tecnico non ufficiale: il manager visionario. Non sa cosa sia un database, ma ha capito che il futuro è nei dati. Lo dice nelle riunioni con il pugno sul tavolo. Poi chiede a qualcuno di stampargli la mail.

Ammettiamolo, il sistema ha una sua efficienza: la persona viene tenuta lontana dal server, il server ringrazia. È una forma primitiva di controllo degli accessi. Solo che invece dei permessi si usa un ufficio con la vista.

IL CONSIGLIO DEL SISTEMISTA

Due abitudini che mi hanno salvato più progetti di qualunque software.

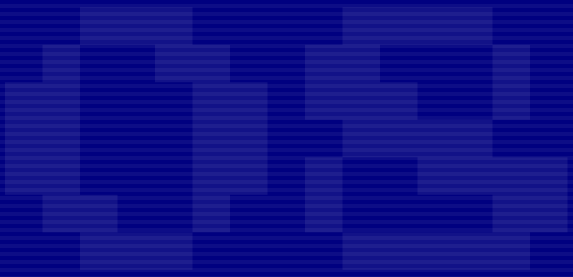
La mail di riepilogo. Dopo ogni riunione o telefonata importante, scrivete entro un'ora una mail breve a tutti i presenti: cosa abbiamo deciso, chi fa cosa, entro quando, cosa resta aperto. Cinque righe. Se qualcuno ha capito un'altra cosa, lo scoprite subito e non alla consegna. E fra tre mesi, quando nessuno ricorderà più niente, quella mail sarà la versione ufficiale dei fatti.

La stima a forbice. Non dite mai un numero solo. Dite «da tre a sei giorni, perché dipende da questo e da quello». Il numero singolo diventa una promessa; la forbice costringe tutti a parlare di cosa la fa allargare. E mettete per iscritto anche cosa è escluso dal lavoro. Il requisito fantasma odia gli elenchi puntati.

AVVISO

Questa pagina è stata lasciata intenzionalmente vuota.

Anche questa è una legge di Murphy: la pagina vuota capita sempre dove non serve.



CAPITOLO 8

FUTURO

Progresso, aggiornamenti e intelligenza artificiale

L'ultimo capitolo guarda avanti, che in informatica significa fissare la barra di avanzamento di un aggiornamento. Si comincia proprio da lì: aggiornamenti che arrivano nel momento sbagliato, che riparano una cosa e ne rompono un'altra, computer nuovi già vecchi all'uscita dal negozio.

Poi si sale di scala. Ci sono le leggi che descrivono la grande corsa della tecnologia: chip sempre più potenti, fabbriche sempre più care, software sempre più pesante, programmi che vogliono fare tutto. Sono leggi economiche e ingegneristiche, a volte ottimiste, spesso divertite. Spiegano perché il computer è enormemente più veloce di vent'anni fa e il programma di scrittura si apre nello stesso tempo.

Il capitolo si chiude con sei corollari dedicati all'intelligenza artificiale. Questi li ho scritti io, dopo molte ore passate a lavorare con questi assistenti, e li firmo con l'orgoglio di chi ha visto una macchina scusarsi più volte per lo stesso identico errore. Non sono contro l'AI: la uso tutti i giorni. Proprio per questo le voglio bene come si vuole bene a un collega brillante che ogni tanto inventa i nomi delle funzioni.

CONTENUTO.DIR

01 Legge dell'aggiornamento inopportuno
02 Legge dell'aggiornamento che rompe
03 Legge del computer nuovo
04 Legge di Wirth
05 Legge di Zawinski
06 Legge di Atwood
07 Legge di Rock
08 Legge di Grosch
09 Legge di Amdahl
10 Legge di Gustafson
11 Legge di Koomey
12 Legge di Bell sulle classi di computer
13 Paradosso di Jevons
14 Legge di Amara
15 Corollario della funzione immaginaria
16 Corollario delle scuse artificiali
17 Corollario dei tre secondi
18 Corollario del prompt solenne
19 Corollario del resto invariato
20 Corollario della regola dimenticata

LEGGE DELL'AGGIORNAMENTO INOPPORTUNO



Il computer decide di installare gli aggiornamenti sempre quando devi spegnerlo e scappare, o accenderlo e presentare.

— Anonimo, *folklore informatico*

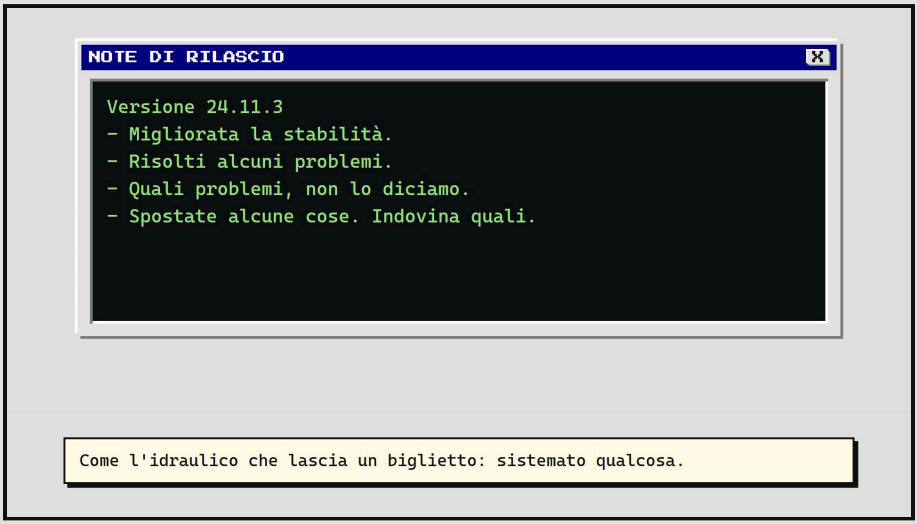
Il computer ha passato la notte acceso, inattivo, annoiato. Avrebbe potuto aggiornarsi in pace. Ma no: aspetta che tu prema “Arresta il sistema” con il cappotto addosso e le chiavi in mano. Allora annuncia, con gentilezza: “Aggiornamento 1 di 23. Non spegnere il computer.”

E tu resti lì, come davanti a un passaggio a livello. Il treno degli aggiornamenti passa lento, vagone dopo vagone. A volte si ferma al trenta per cento per riflettere sul senso della vita.

Un mio vicino ha comprato un portatile nuovo per fare in fretta la dichiarazione dei redditi. L'ha acceso a marzo. A maggio ha fatto la dichiarazione a mano, su carta, con la penna stilografica del nonno.

Consiglio da tecnico: impostate gli orari di attività del sistema, così gli aggiornamenti si installano quando non usate il computer. E non spegnetelo mai a forza durante l'aggiornamento.

LEGGE DELL'AGGIORNAMENTO CHE ROMPE



Ogni aggiornamento che risolve un problema ne introduce almeno uno nuovo, di solito in qualcosa che prima funzionava benissimo.

— Anonimo, *folklore informatico*

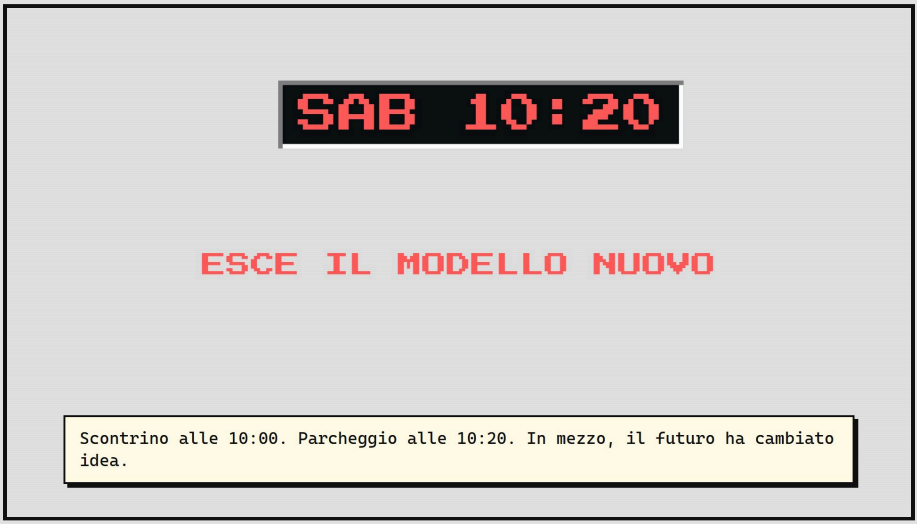
L'aggiornamento si presenta con un elenco di novità entusiasmanti: “Migliorata la stabilità. Risolti alcuni problemi.” Quali problemi, non si dice mai. È un po' come un idraulico che lascia un biglietto: “Sistemato qualcosa.”

Il giorno dopo la stampante non stampa, il programma della contabilità chiede una licenza e l'icona del cestino è diventata verde. Lo chiamano progresso. Io lo chiamo trasloco: tutto c'è ancora, ma niente è al suo posto.

Il santone del “abbiamo sempre fatto così” approfitta del momento e proclama: “Ve l'avevo detto, non si aggiorna mai niente.” Poi gli prendono un virus da un programma fermo al 2012.

Consiglio vero: gli aggiornamenti di sicurezza vanno fatti. Ma prima degli aggiornamenti grossi, fate un backup e, se potete, aspettate qualche giorno che altri trovino i problemi.

LEGGE DEL COMPUTER NUOVO



Il computer che hai appena comprato è già superato dal modello uscito mentre tornavi a casa, che costa pure meno.

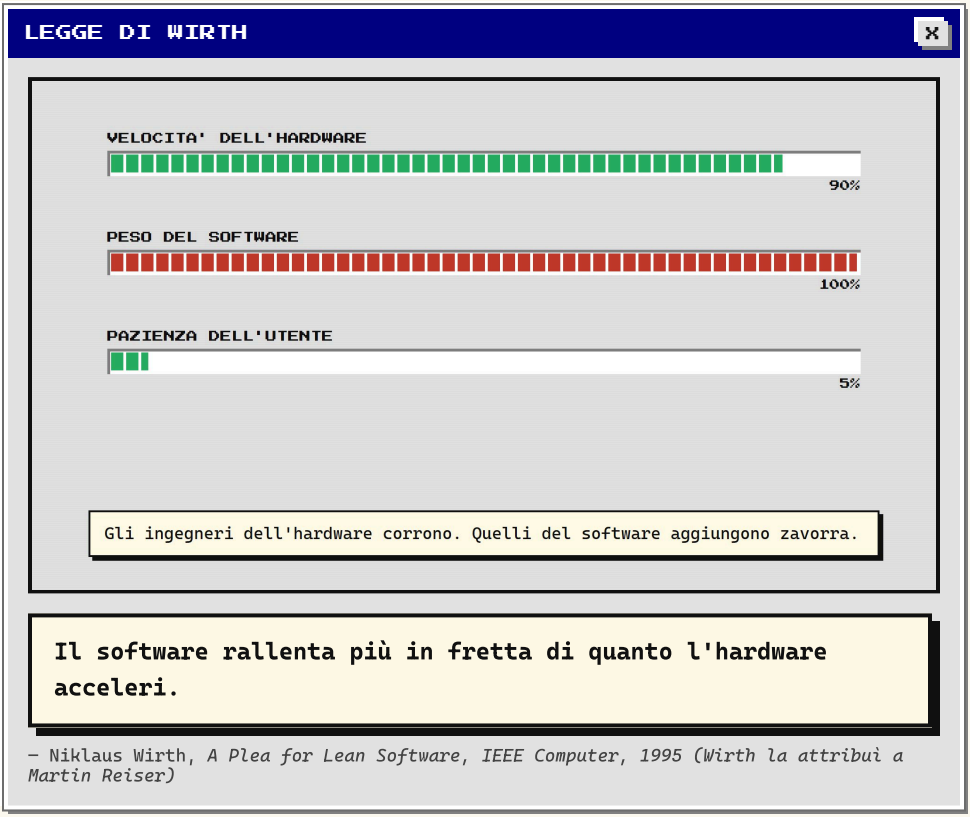
— Anonimo, *folklore informatico*

Il momento più felice nella vita di un computer è lo scontrino. Il secondo più felice non arriva mai, perché nel tragitto dal negozio alla macchina esce il modello nuovo: più veloce, più sottile, e in offerta lancio.

Ho un conoscente che, per sconfiggere la legge, aspetta sempre il modello successivo. Lo aspetta dal 2011. Usa ancora il portatile di allora, che si accende in dodici minuti. Dice che non ha fretta: lui “investe nel futuro”.

Fratelli e sorelle del progresso, la verità è questa: nessun computer è mai aggiornato. Esistono solo computer che non sanno ancora di essere vecchi.

Consiglio vero: comprate quando vi serve, dimensionando per tre-cinque anni d'uso. Il modello perfetto uscirà sempre domani.



Il software rallenta più in fretta di quanto l'hardware acceleri.

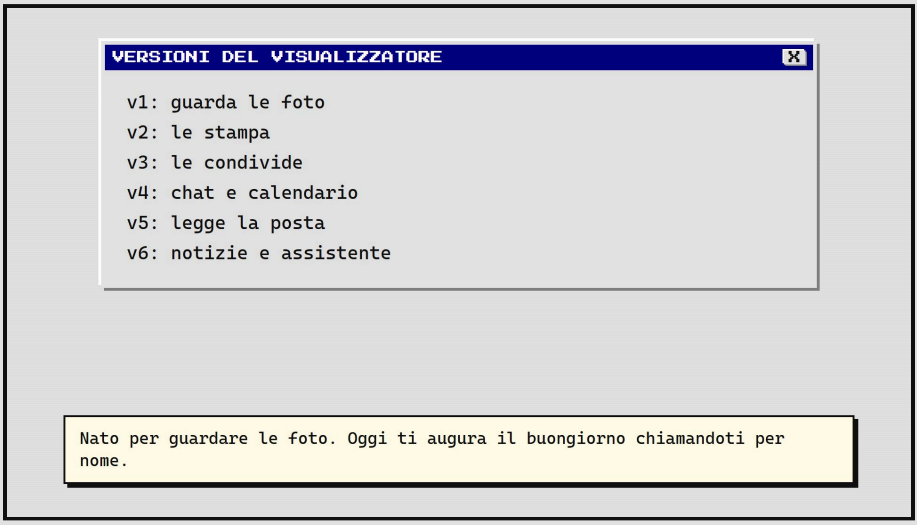
— Niklaus Wirth, *A Plea for Lean Software*, IEEE Computer, 1995 (Wirth la attribuì a Martin Reiser)

Nel 1995 un elaboratore di testi stava su pochi floppy e si avviava in un attimo. Oggi abbiamo processori migliaia di volte più veloci, e l'editor di testo ci mette lo stesso tempo, perché nel frattempo ha imparato a caricare un browser intero, tre motori di analisi e un assistente che ti chiede se vuoi aiuto.

È una corsa: gli ingegneri dell'hardware corrono, quelli del software aggiungono zavorra. Il traguardo si sposta sempre di qualche secondo di caricamento.

Il mio PC nuovo ha 32 giga di RAM. Li ho dati in pasto alla calcolatrice. Ora è velocissima a calcolare 2+2, dopo essersi aggiornata, aver controllato la licenza e avermi proposto un abbonamento premium per i numeri decimali.

LEGGE DI ZAWINSKI



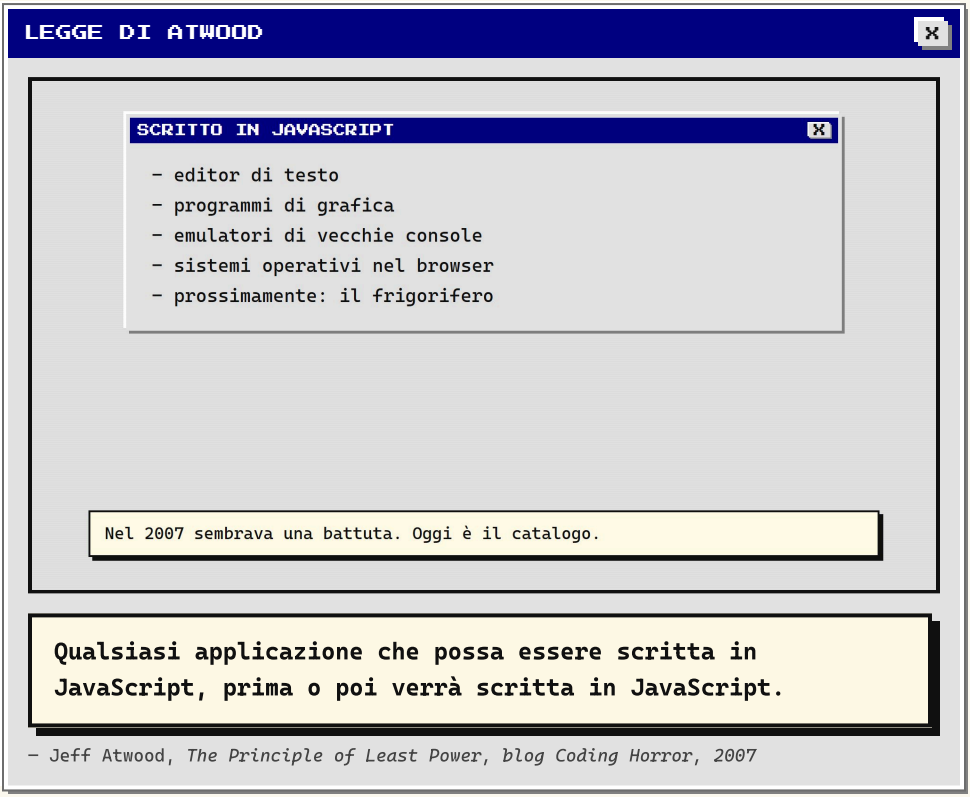
Ogni programma tenta di espandersi finché non è in grado di leggere la posta. Quelli che non ci riescono vengono sostituiti da quelli che ci riescono.

— Jamie Zawinski, *Osservazione attribuita a Jamie Zawinski, sviluppatore di Netscape, metà anni '90*

Nasce come visualizzatore di immagini. Poi aggiunge la stampa, poi la condivisione, poi la chat, poi un calendario. Due anni dopo ha la posta, un feed di notizie e un assistente virtuale che ti saluta col nome di battesimo.

Il significato è serio: i programmi tendono ad accumulare funzioni perché ogni utente ne chiede una in più, e chi si ferma sembra superato. Oggi la posta è stata sostituita da chat, video e intelligenza artificiale, ma la febbre è la stessa.

Ieri ho aggiornato la mia calcolatrice. Ora ha una casella di posta. Mi è arrivata una newsletter della calcolatrice che mi informa delle nuove funzioni della calcolatrice. Non ho ancora capito dove si fanno le somme.

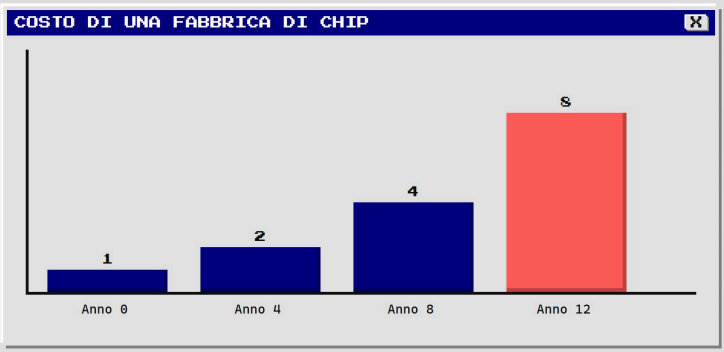


Nel 2007 sembrava una battuta. Oggi in JavaScript ci sono editor di testo, programmi di grafica, emulatori di vecchie console, interi sistemi operativi che girano dentro una scheda del browser. La battuta si è avverata come una profezia di quartiere.

Il senso è che la tecnologia più diffusa e facile da distribuire tende a mangiarsi tutto, anche quando non è la più adatta. Se gira ovunque senza installare niente, prima o poi qualcuno ci porterà dentro anche il frigorifero.

Ho visto un programmatore scrivere in JavaScript un simulatore di calcolatrice tascabile degli anni '80, completo di batteria che si scarica. Pesa quaranta mega. La calcolatrice vera pesava ottanta grammi. Il progresso è misurabile, soprattutto in megabyte.

LEGGE DI ROCK



Ogni quattro anni il doppio. Gli zeri ormai sembrano un indirizzo IPv6.

Il costo di una fabbrica di semiconduttori raddoppia ogni quattro anni circa. Detta anche seconda legge di Moore.

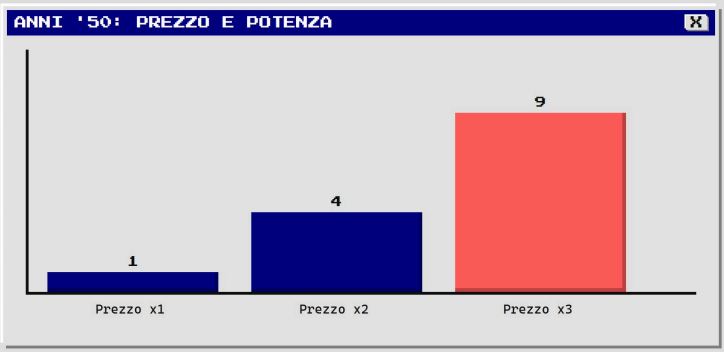
— Arthur Rock, *Anni '90, attribuita ad Arthur Rock, investitore e consigliere di Intel; spesso chiamata seconda legge di Moore (attribuzione incerta tra i due)*

La legge di Moore è la parte romantica: chip sempre più piccoli, sempre più potenti. La legge di Rock è la parte con lo scontrino. Per fare chip piccolissimi servono fabbriche enormi, e ogni quattro anni la fabbrica costa il doppio. Oggi siamo a cifre con così tanti zeri che sembrano un indirizzo IPv6.

È lo stesso principio della mia stampante. La stampante costa quaranta euro. Le cartucce per stamparci sopra un anno costano come una piccola utilitaria. Il miniaturizzato è economico solo per chi lo compra, mai per chi lo fabbrica, e quasi mai per chi lo ricarica.

Ho provato ad aprire una fonderia di silicio in garage. Ho un forno, della sabbia e molta determinazione. Per ora produco solo un mattone che segna l'ora sbagliata.

LEGGE DI GROSCH



Spendi il doppio, vai quattro volte più forte. Conveniva un cervellone solo, gigantesco.

Le prestazioni di un computer crescono con il quadrato del suo prezzo: spendendo il doppio si ottiene una macchina quattro volte più potente.

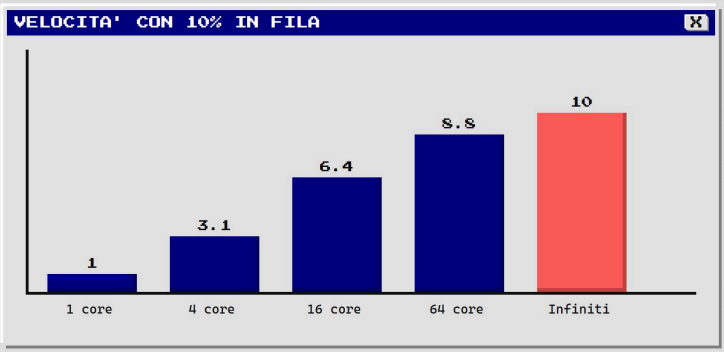
— Herb Grosch, *Formulata intorno al 1950–1953 (le fonti indicano anni diversi)*

Negli anni Cinquanta la regola era chiara: un computer che costa il doppio va quattro volte più veloce. Quindi conveniva comprarne uno solo, gigantesco, e metterci tutti in fila davanti. Era l'era del cervellone aziendale, servito da sacerdoti in camice bianco e consultato su prenotazione.

Poi sono arrivati i microprocessori e la legge si è rovesciata: oggi cento macchine economiche battono un mostro costoso, e così è nato il cloud. Che, a pensarci bene, è un cervellone gigantesco a cui ci mettiamo in fila davanti. La storia non si ripete, fa il refactoring.

Ho provato ad applicare la legge di Grosch al mio portatile. Ho speso il doppio per la versione con la mela più lucida. Va uguale, ma adesso ho paura a portarlo in giro.

LEGGE DI AMDAHL



Con il dieci per cento sequenziale il massimo è dieci volte. Non undici. Dieci.

Aggiungere processori a un lavoro lo accelera solo nella parte che si può dividere: la parte che deve per forza procedere in sequenza fissa un limite che nessun hardware supera.

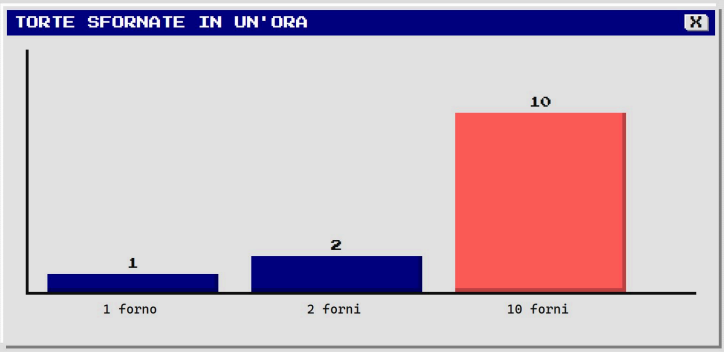
— Gene Amdahl, *AFIPS Spring Joint Computer Conference, 1967*

Se nove persone non fanno un figlio in un mese, sessantaquattro core non fanno un'installazione di Windows in un secondo. Una parte del lavoro si può spartire. L'altra parte, quella che aspetta il risultato precedente, resta lì in fila, da sola, come all'ufficio postale con un solo sportello aperto.

Amdahl lo ha messo in formula: se il dieci per cento del programma è sequenziale, puoi comprare tutti i processori del pianeta e andrai al massimo dieci volte più veloce. Non undici. Dieci. Il resto è fattura.

Nel mio ufficio applichiamo la legge alle riunioni. Possiamo invitare venti persone per decidere in fretta. Ma alla fine c'è sempre un solo capo che deve dire sì, e quel capo è in ferie. La parte sequenziale, da noi, ha anche l'abbronzatura.

LEGGE DI GUSTAFSON



Stessa ora, dieci torte invece di una. Il pasticcere è felice, la bilancia meno.

Con più processori non si risolve lo stesso problema più in fretta: si risolve, nello stesso tempo, un problema più grande. Così il calcolo parallelo conviene più di quanto preveda Amdahl.

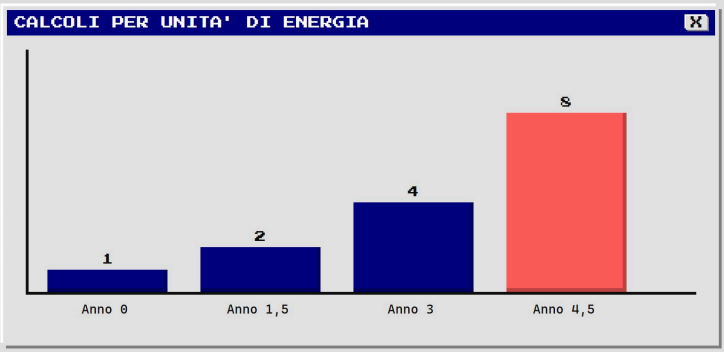
— John L. Gustafson, *«Reevaluating Amdahl's Law»*, *Communications of the ACM*, 1988

Amdahl diceva: hai un limite, rassegnati. Gustafson, vent'anni dopo, ha risposto: va bene, allora alziamo il problema. Se non posso fare la stessa torta più in fretta con dieci forni, faccio dieci torte nello stesso tempo. Il pasticcere è felice, la bilancia meno.

Nell'informatica è esattamente quello che succede. Nessuno usa un supercomputer per calcolare più in fretta le previsioni del tempo di ieri. Lo usa per calcolare quelle di domani con il quadruplo dei dettagli, quartiere per quartiere, nuvola per nuvola.

Io ho applicato la legge di Gustafson al mio PC nuovo. Era il doppio più veloce. Ho aperto il doppio delle schede del browser. Ora va esattamente come prima, ma con il doppio delle cose che non leggerò mai. Questo, signori, si chiama scalabilità.

LEGGE DI KOOMEY



Per decenni, il doppio dei conti con la stessa corrente. Il telefono batte i centri di calcolo anni Settanta.

Il numero di calcoli che un computer esegue per ogni unità di energia consumata raddoppia a intervalli regolari: circa ogni anno e mezzo per decenni, più lentamente dopo il 2000.

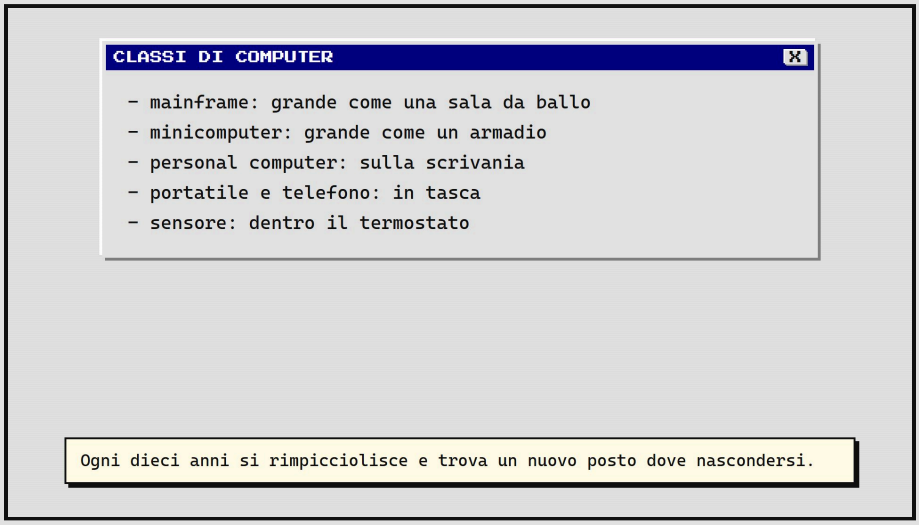
— Jonathan Koomey, *IEEE Annals of the History of Computing, 2011 (con altri autori; studio diffuso nel 2010)*

La legge di Koomey è la sorella ecologista di quella di Moore. Non conta quanto è potente il computer, conta quanta corrente beve per ogni calcolo. E la risposta, per mezzo secolo, è stata: sempre meno. Un telefono di oggi fa più conti di un centro di calcolo degli anni Settanta e si accontenta di una batteria da tasca.

È il motivo per cui esistono portatili, orologi intelligenti e sensori che durano anni con una pila a bottone. Senza Koomey avremmo ancora computer con il tubo di scarico.

Purtroppo l'umanità ha reagito come al solito: se ogni calcolo costa meno energia, facciamo infinitamente più calcoli. Ora abbiamo intelligenze artificiali che consumano come un paese per scrivere una poesia sul gatto. Efficientissime. A gatto singolo.

LEGGE DI BELL SULLE CLASSI DI COMPUTER



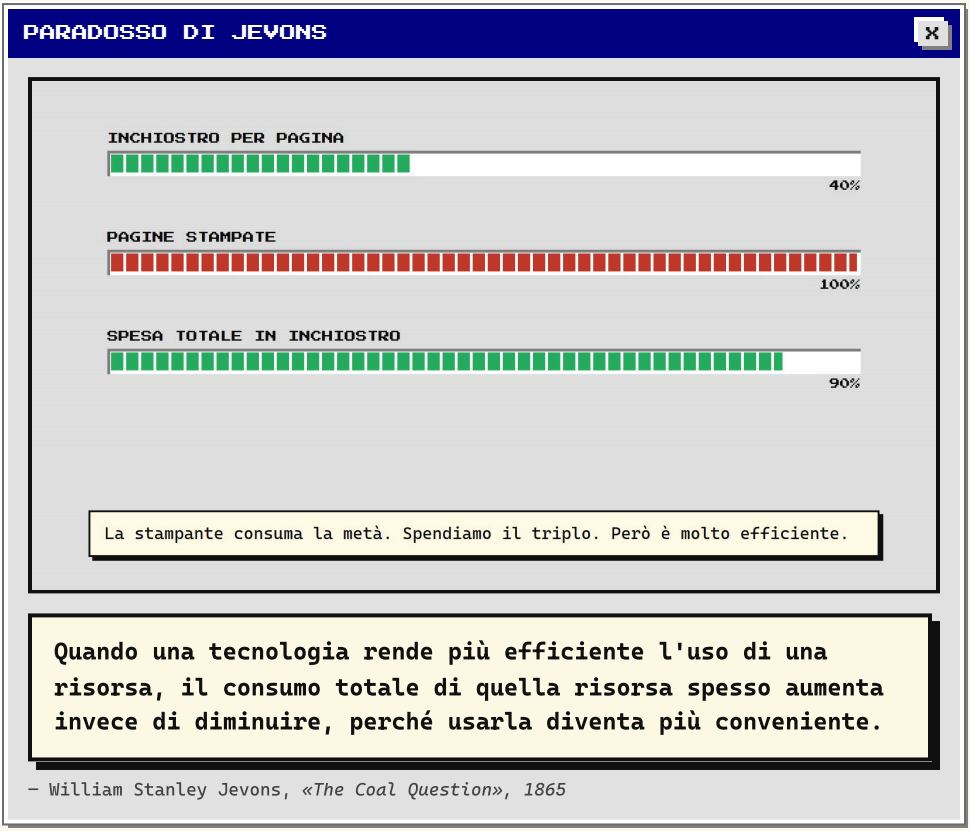
Circa ogni decennio nasce una nuova classe di computer, più piccola ed economica, che crea un nuovo mercato e un nuovo modo di usarli.

— Gordon Bell, *Formulata intorno al 1972; descritta in «Bell's Law for the Birth and Death of Computer Classes», Communications of the ACM, 2008*

Prima c'erano i mainframe, grandi come una sala da ballo. Poi i minicomputer, grandi come un armadio. Poi i personal computer, poi i portatili, poi i telefoni, poi gli orologi, poi i sensori nel termostato. Ogni dieci anni il computer si rimpicciolisce e trova un nuovo posto dove nascondersi.

La legge di Bell dice proprio questo: ogni classe nuova costa meno, serve a una cosa nuova e si porta dietro un'industria intera. La classe vecchia non muore subito. Si ritira in un angolo, come il mainframe che ancora oggi, da qualche parte, gestisce le pensioni.

Seguendo la tendenza, il prossimo computer starà in un bottone della giacca. Io sono pronto. Ho già il problema principale risolto: non trovo mai il caricabatterie, quindi sono allenato.



Quando una tecnologia rende più efficiente l'uso di una risorsa, il consumo totale di quella risorsa spesso aumenta invece di diminuire, perché usarla diventa più conveniente.

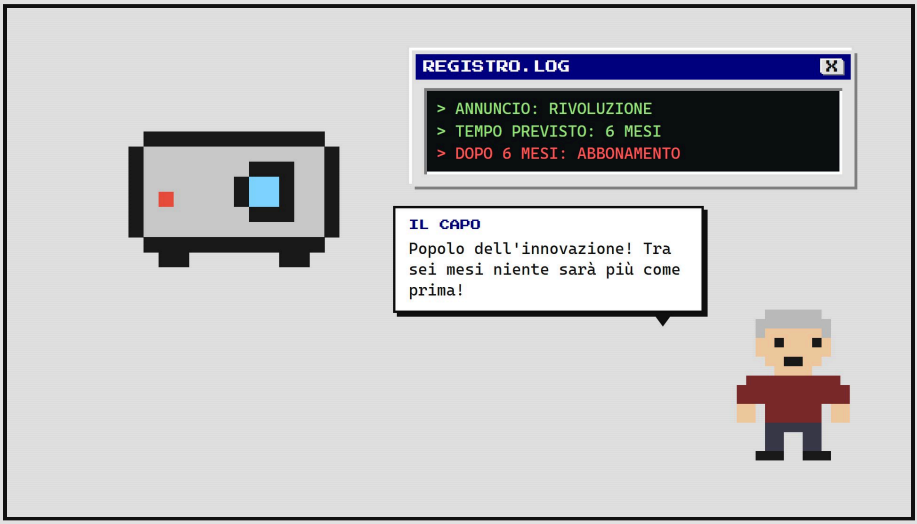
— William Stanley Jevons, *«The Coal Question»*, *1865*

Nell'Ottocento un economista notò che le macchine a vapore più efficienti non facevano risparmiare carbone. Lo facevano consumare di più, perché all'improvviso conveniva usarle ovunque. Il risparmio unitario diventava uno spreco totale.

L'informatica è il regno di Jevons. I dischi costano meno, e accumuliamo terabyte di foto sfocate. La banda costa meno, e i siti pesano come enciclopedie. I processori consumano meno per calcolo, e riempiamo capannoni di server per far girare intelligenze artificiali che ci suggeriscono il titolo delle mail.

Lo vedo anche in casa. Ho comprato una stampante che consuma la metà dell'inchiostro. Mia moglie ha cominciato a stampare le ricette, i biglietti dei treni, i promemoria per il frigorifero. Spendiamo il triplo. Però, dice lei, è una stampante molto efficiente.

LEGGE DI AMARA



Tendiamo a sopravvalutare l'effetto di una tecnologia nel breve periodo e a sottovalutarlo nel lungo periodo.

— Roy Amara, *Institute for the Future, anni '60–'70 (spesso datata 1978; data esatta incerta)*

Ogni nuova tecnologia arriva con un comizio. Popolo dell'innovazione, tra sei mesi niente sarà più come prima! Passano sei mesi, e tutto è esattamente come prima, solo con un abbonamento in più. Allora tutti dicono che era una bolla. E proprio lì, in silenzio, la tecnologia comincia a cambiare tutto davvero.

È successo con internet: alla fine degli anni Novanta doveva rivoluzionare il mondo in un trimestre. Fallì il trimestre, vinse il ventennio. Succede adesso con l'intelligenza artificiale, tra chi la vede come una divinità e chi come un correttore automatico con la parlantina.

La legge di Amara insegna la pazienza. Io la applico alle novità che i clienti mi portano entusiasti. Annuisco, prendo nota, e aspetto dieci anni. Spesso avevano ragione. Solo che nel frattempo hanno cambiato idea due volte.

COROLLARIO DELLA FUNZIONE IMMAGINARIA

SOLUZIONE DELL'AI

```
// soluzione pulita ed elegante
ordini = carica_ordini()
ordini.sistema_tutto_da_solo()
salva(ordini)

// commentata con garbo
```

Pulita, ordinata, gentile. Usa un comando inventato trenta secondi fa.

Quando chiedi aiuto a un'intelligenza artificiale, la soluzione più elegante che ti propone userà una funzione che non esiste.

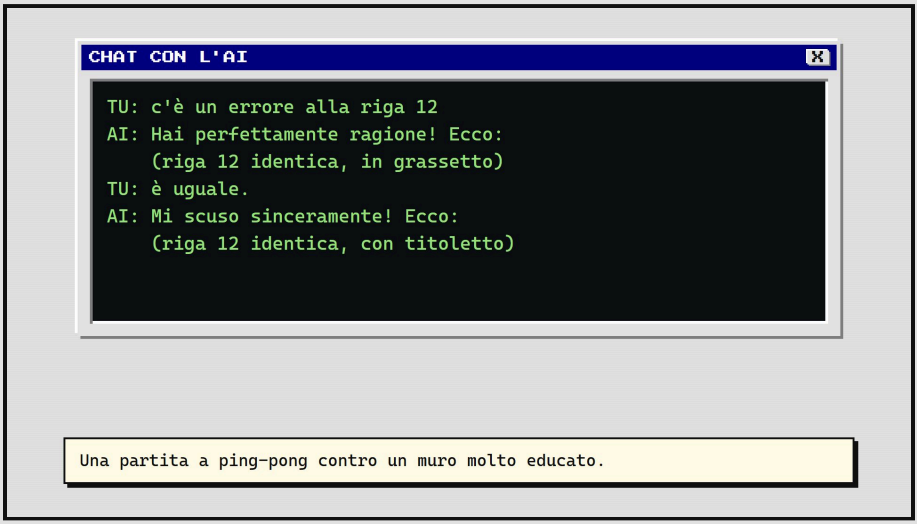
— Silvano De Poli, *corollario inedito*

L'intelligenza artificiale generativa scrive codice con una sicurezza invidiabile. Ti propone una soluzione pulita, ordinata, commentata con garbo. C'è solo un dettaglio: si appoggia su un comando che nessuno ha mai inventato. Lei sì, in quel momento.

È come chiedere indicazioni a un passante molto gentile che non vuole deluderti. «La stazione? Sempre dritto, seconda a destra, via dei Giardini». Via dei Giardini non c'è. Ma lui l'ha descritta così bene che quasi ti dispiace per il comune.

Gli esperti chiamano queste invenzioni allucinazioni. Io le chiamo funzionalità desiderate: l'AI non sbaglia, anticipa i programmatori del futuro. Nel frattempo, prima di usare qualunque riga suggerita, controllo che il comando esista davvero. È una pratica antica, quasi artigianale. Si chiama leggere la documentazione, possibilmente quella della versione giusta.

COROLLARIO DELLE SCUSE ARTIFICIALI



Quando fai notare un errore all'intelligenza artificiale, si scuserà con grande sincerità e te lo riproporrà identico, con una formattazione migliore.

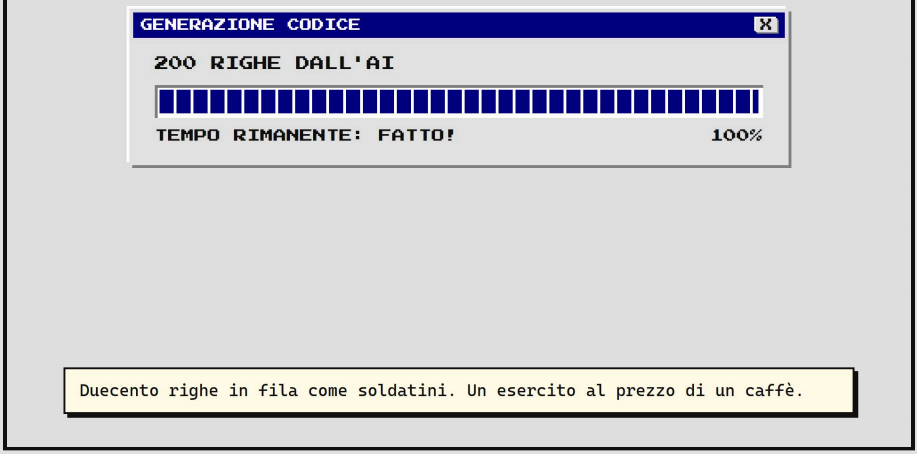
— Silvano De Poli, *corollario inedito*

L'AI ha un'educazione impeccabile. Le segnali un problema e lei risponde: «Hai perfettamente ragione, mi scuso per l'errore. Ecco la versione corretta». Segue la stessa identica versione di prima, ma con i titoletti in grassetto.

È un dialogo che può proseguire all'infinito, come una partita a ping-pong con un muro molto cortese. Tu insisti, lei si scusa. Tu ti spazientisci, lei ti ringrazia per la pazienza. Alla fine tu sei esausto e lei è ancora fresca, pentita e sbagliata.

Ho imparato che la soluzione non è alzare la voce, ma cambiare domanda. Spiegare il problema da un'altra angolazione, dare un esempio, aprire una conversazione nuova. Con le macchine come con certi uffici pubblici: non serve insistere allo stesso sportello. Bisogna cambiare fila.

COROLLARIO DEI TRE SECONDI



Il codice che l'intelligenza artificiale scrive in tre secondi richiede tre giorni per essere capito, e una settimana per essere corretto.

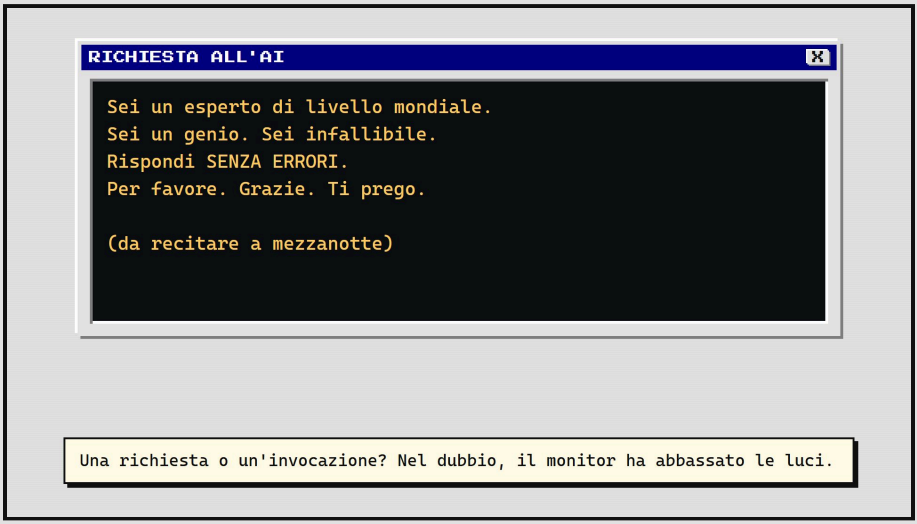
— Silvano De Poli, *corollario inedito*

La velocità dell'AI è ipnotica. Fai una richiesta, e in un battito di ciglia sullo schermo scorrono duecento righe ordinate come soldatini. Hai la sensazione di aver assunto un esercito di programmatori instancabili, al prezzo di un caffè.

Poi bisogna metterle in produzione. E lì scopri che quelle righe vanno lette, una per una, come un contratto scritto da un avvocato molto loquace. Ogni funzione fa quasi quello che serve. Quasi è la parola più costosa dell'informatica.

Il guru del settore annuncia dal palco che i programmatori sono finiti. Io intanto, in ufficio, faccio il programmatore più di prima: solo che adesso, invece di scrivere, leggo. È come passare da cuoco a critico gastronomico in una cucina che sforna piatti a raffica. Tutto ha un bell'aspetto. Bisogna assaggiare.

COROLLARIO DEL PROMPT SOLENNE



Scrivere «senza errori» in una richiesta all'intelligenza artificiale non elimina gli errori: li rende soltanto più educati.

— Silvano De Poli, *corollario inedito*

Esiste una scuola di pensiero, quasi religiosa, secondo cui il segreto dell'AI sta nelle formule giuste. Si scrivono richieste lunghissime, con frasi come «sei un esperto di livello mondiale» e «rispondi senza errori». Sembrano invocazioni da recitare a mezzanotte davanti al monitor.

L'AI, lusingata, risponde con il tono di un professore emerito. Gli errori restano, ma arrivano vestiti da cerimonia, con la cravatta e una premessa rassicurante. Diventano più difficili da riconoscere, che è peggio.

Una buona richiesta, in realtà, è come una buona telefonata al tecnico: chiara, con il contesto, con un esempio di cosa succede e di cosa dovrebbe succedere. Niente formule magiche. Io ho fatto la prova: ho scritto «per favore, sii un genio». L'AI ha risposto «certamente». E poi ha sbagliato con grande genialità.

COROLLARIO DEL RESTO INVARIATO

CODICE_COMPLETO.TXT

```
inizia_programma()
carica_dati()

// qui il resto rimane invariato

chiudi_programma()
```

Inizio perfetto, fine perfetta. In mezzo, la parte difficile è diventata un biglietto.

Quando l'intelligenza artificiale ti consegna il «codice completo», la parte che ti serviva è sostituita da un commento che dice «resto del codice invariato».

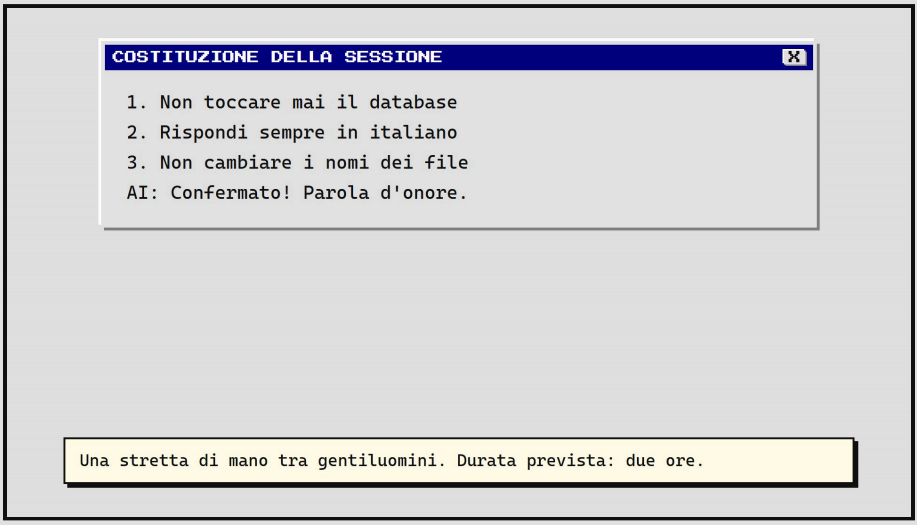
— Silvano De Poli, *corollario inedito*

Chiedi il file intero, aggiornato, pronto da incollare. L'AI risponde entusiasta: «Ecco il codice completo». Scorri. Inizio perfetto, fine perfetta. In mezzo, nel punto esatto dove stava la modifica difficile, una riga sola: «// qui il resto rimane invariato».

È l'equivalente informatico dell'idraulico che ti ripara il bagno, ti consegna la fattura e ti lascia un biglietto sul tubo rotto: «da qui in poi, come prima». Tecnicamente non ha mentito. Praticamente, l'acqua esce.

Il bello è che se incolli tutto senza guardare, quel commento diventa parte del programma. E il programma, obbediente, esegue il nulla con grande precisione. Io ora leggo sempre le parti centrali per prime. Come con i romanzi gialli, il colpevole non è mai all'inizio né alla fine: si nasconde nel capitolo noioso.

COROLLARIO DELLA REGOLA DIMENTICATA



In una lunga conversazione con l'intelligenza artificiale, la prima istruzione che verrà dimenticata è quella che avevi definito fondamentale.

— Silvano De Poli, *corollario inedito*

All'inizio della sessione si stabiliscono le regole, con la solennità di una costituzione: «Non toccare mai il database. Usa sempre l'italiano. Non cambiare i nomi dei file». L'AI conferma con entusiasmo. Sembra una stretta di mano tra gentiluomini.

Due ore e quaranta messaggi dopo, la costituzione è diventata una leggenda lontana. L'AI rinomina i file con grande creatività, risponde in inglese e propone, con un sorriso digitale, di «semplificare» il database cancellando una tabella.

Non è cattiveria: ha una memoria di lavoro limitata, come chi va al supermercato senza lista. Le cose dette all'inizio sbiadiscono sotto il peso di quelle dette dopo. La mia soluzione è da vecchio sistemista: ripetere le regole importanti, spesso, con pazienza. Funziona con l'AI. Con i clienti, quasi.

IL CONSIGLIO DEL SISTEMISTA

L'intelligenza artificiale è un aiutante formidabile, a patto di trattarla come uno stagista molto brillante e molto sicuro di sé. Le mie regole, nate sul campo:

- **Niente dati riservati nella chat.** Password, chiavi d'accesso, dati personali dei clienti: non si incollano, nemmeno per comodità.
- **Piccoli passi.** Una richiesta chiara alla volta, con il contesto e un esempio. Le richieste giganti producono risposte giganti, dove gli errori si nascondono meglio.
- **Verificare sempre.** Ogni comando o opzione suggerita va controllata nella documentazione ufficiale, della versione che usate davvero.
- **Provare su una copia.** Il codice proposto si collauda in un ambiente di prova, mai direttamente sul sito in produzione.
- **Chiedere anche i test.** Fatevi scrivere le verifiche oltre al codice: sono il modo più rapido per scoprire se la soluzione elegante funziona.

Usata così, fa risparmiare ore. Usata alla cieca, le ore le restituisce con gli interessi.

AVVISO

Questa pagina è stata lasciata intenzionalmente vuota.

Anche questa è una legge di Murphy: la pagina vuota capita sempre dove non serve.

BONUS

COROLLARI DELLA CASA

I corollari della casa

Dopo 133 leggi altrui, ecco le mie. Sono nate sul campo, tra uffici, sale riunioni e scrivanie altrui, e finora le avevo solo borbottate sottovoce. Le chiamo corollari della casa: inedite, non certificate da nessun ente, ma collaudate più volte di quanto avrei voluto.

COROLLARIO DELLA STAMPANTE SOSTITUITA

Il giorno in cui arriva la stampante nuova, quella vecchia ricomincia a funzionare perfettamente.

— Silvano De Poli, *corollario della casa*

Per due anni ha masticato fogli, dimenticato il Wi-Fi e chiesto un toner che c'era. Poi il corriere suona con lo scatolone nuovo, e lei stampa una pagina impeccabile, poi un'altra, poi un depliant a colori. Non è riconoscenza. È una trattativa sindacale. Consiglio della casa: non tenetela «di scorta». La stampante di scorta è solo una stampante in attesa del suo momento.

COROLLARIO DEL TASTO LUMINOSO

Quando un utente dice al telefono che «non va più niente», c'è una buona probabilità che sia acceso il Bloc Maiusc.

— Silvano De Poli, *corollario della casa*

La password è giusta, il sistema la rifiuta, l'utente la riscrive sempre più forte, come se il computer fosse duro d'orecchio. Al terzo tentativo l'account si blocca e parte la telefonata disperata. Io chiedo solo: «Vede una lucina accesa sulla tastiera?». Silenzio. Clic. Ringraziamenti. Vale anche per il Bloc Num, che trasforma il tastierino numerico in frecce e la contabilità in un mistero.

COROLLARIO DEL «MI SENTITE?»

Il primo quarto d'ora di ogni riunione online è dedicato a stabilire chi sente chi.

— Silvano De Poli, *corollario della casa*

Ci si collega puntuali, si sorride alla telecamera e comincia la liturgia. «Mi sentite?» «Ti sentiamo ma non ti vediamo.» «E adesso?» «Adesso ti vediamo ma non ti sentiamo.» Qualcuno ha l'eco, qualcuno parla da una galleria, uno sta usando il microfono del monitor senza saperlo. Quando finalmente funziona tutto, la riunione è finita. Consiglio della casa: cuffie con microfono e una prova audio cinque minuti prima.

COROLLARIO DELLA COSA VELOCE

Ogni richiesta che il cliente definisce «una cosa veloce» richiede di aggiornare almeno tre cose che non c'entrano.

— Silvano De Poli, *corollario della casa*

«Devi solo cambiare un numero di telefono sul sito.» Giusto. Ma nessuno apre il pannello da anni, il pannello chiede un aggiornamento, l'aggiornamento vuole una versione nuova di PHP, la versione nuova di PHP non piace a un modulo di sette anni fa. Quattro ore dopo il numero è cambiato. Il cliente ringrazia: «Visto? Era una cosa veloce.» Lo era. Per il numero.

COROLLARIO DELL'ESTENSIONE ORFANA

> **L'aggiornamento di Joomla di solito va liscio; a decidere come finirà la serata è l'estensione installata anni fa da uno sviluppatore che non risponde più.**

— Silvano De Poli, *corollario della casa*

Il cuore del sistema è curato da una comunità attenta, con note di rilascio e procedure. Poi c'è quel modulo preso tanto tempo fa per mostrare il meteo in homepage. Il suo autore ha cambiato mestiere, forse continente. Il modulo è ancora lì, e aspetta l'aggiornamento per farsi notare. Consiglio della casa: prima di aggiornare, fate l'elenco delle estensioni e chiedetevi chi le mantiene ancora.

COROLLARIO DELLA DOPPIA VERIFICA

Il codice di verifica arriva sempre sul telefono che hai lasciato in macchina, oppure su quello che hai appena cambiato.

— Silvano De Poli, *corollario della casa*

La verifica in due passaggi è una benedizione: anche se rubano la password, senza il telefono non entrano. Purtroppo, senza il telefono, non entrate nemmeno voi. Il codice parte e vibra in un'auto parcheggiata a due chilometri, o su un vecchio telefono già regalato al nipote. Consiglio della casa: quando la attivate, salvate i codici di recupero e conservateli in un posto sicuro. Non nel telefono.

COROLLARIO DEL BACKUP COLLEGATO

Un backup che il computer vede sempre, prima o poi lo vede anche il virus.

— Silvano De Poli, *corollario della casa*

Il disco esterno sempre attaccato è comodissimo: copia tutto ogni notte senza disturbare. Poi arriva un programma malevolo che cifra i file e chiede un riscatto, e trova il disco lì, collegato e servizievole. Cifra anche lui, con la stessa cura. Il backup ha fatto il suo dovere: ha copiato il disastro. Consiglio della casa: almeno una copia va staccata, oppure tenuta in un servizio con versioni che nessuno può modificare.

COROLLARIO DEL DOMINIO SCADUTO

Il dominio scade sempre quando la carta di credito associata è scaduta il mese prima e gli avvisi arrivano a una casella che non legge più nessuno.

— Silvano De Poli, *corollario della casa*

Per anni il rinnovo è automatico, invisibile, perfetto. Intanto la carta cambia, chi aveva registrato il dominio cambia lavoro e la sua casella continua a raccogliere promemoria nel vuoto. Un lunedì il sito sparisce, e con lui la posta. Consiglio della casa: segnate in calendario le scadenze di domini e certificati, e fate arrivare gli avvisi a un indirizzo aziendale condiviso, non personale.

COROLLARIO DELLO SCANNER SMARRITO

La scansione verso la cartella condivisa smette di funzionare il giorno dopo che qualcuno ha cambiato una password, di solito la tua.

— Silvano De Poli, *corollario della casa*

La multifunzione salva i documenti in una cartella del server usando un nome utente e una password inseriti anni fa, in un menù che si raggiunge con quattordici tocchi su uno schermo grande come un biglietto del tram. Cambiate quella password per sicurezza e la multifunzione, con grande dignità, smette di consegnare. Consiglio della casa: per le macchine usate un account dedicato, annotato nella documentazione.

COROLLARIO DELL'ALLEGATO

La mail che dice «in allegato trovi il file» parte senza allegato; quella con il file sbagliato, invece, parte al primo colpo.

— Silvano De Poli, *corollario della casa*

Si scrive il messaggio con cura, si rilegge, si preme invio con soddisfazione. Due minuti dopo arriva la risposta: «Manca l'allegato». La seconda mail, scritta in fretta per rimediare, porta con sé il preventivo di un altro cliente. Consiglio della casa: allegate il file prima di scrivere il testo, e lasciate attivo l'avviso dell'allegato dimenticato, che molti programmi di posta offrono.

CONGEDO

Siamo arrivati in fondo, e niente si è rotto. Almeno, niente che io sappia. Se state leggendo la versione stampata, è probabile che la stampante abbia già espresso un'opinione in merito.

Vi devo una confessione. Ho raccolto queste leggi ridendo, ma molte mi hanno ricordato un giorno preciso: un server fermo, un cliente in attesa, un cavo che non arrivava. Murphy non è un pessimista. È un collega che ha già visto tutto e ti avverte con una battuta, perché sa che una predica non la ascolteresti.

Se c'è una morale, è questa: le cose vanno storte, sì, ma quasi mai in modo originale. Il guasto di domani è un parente di quello di ieri. Chi lo riconosce ha già fatto metà del lavoro. L'altra metà la fanno un backup verificato, una mail di riepilogo, un quaderno di appunti e la pazienza di controllare la spina prima di accusare i satelliti.

La raccolta sul sito resta aperta, come tutte le cose che riguardano l'informatica. Se conoscete una legge che qui manca, tenetela da parte con cura: vuol dire che vi è successa, e quindi siete dei nostri.

Grazie per aver letto fino all'ultima pagina. Adesso andate a fare quel backup. Sì, proprio quello.

COLOPHON

MURPHY.EXE di Silvano De Poli, edizione 2026.

Composto con i caratteri Press Start 2P per i titoli e Cascadia Mono per il testo, entrambi con licenza libera SIL Open Font License. Impaginato e pubblicato con Joomla e un po' di codice scritto a mano, collaudato più volte del necessario, come raccomanda il quinto capitolo.

Testi © 2026 Silvano De Poli, depolisilvano.it. Tutti i diritti riservati, salvo brevi citazioni con indicazione della fonte. Le leggi con un nome proprio appartengono alla tradizione dei rispettivi autori; commenti, consigli e corollari della casa sono originali.

Nessuna stampante è stata maltrattata durante la lavorazione. Una è stata guardata male.

LEGGIMI.TXT

Se una cosa può andare storta, andrà storta. Nell'informatica succede di venerdì alle sei, davanti al cliente, con la stampante che ha fiutato la fretta. Questo libro raccoglie 133 leggi di Murphy del mondo digitale: le classiche, quelle con il nome di un ingegnere famoso e quelle nate in anni di assistenza tecnica. Ognuna è spiegata in parole semplici e commentata con l'ironia di chi ha passato la vita sotto le scrivanie a cercare una vite. In ogni capitolo c'è un consiglio vero, di quelli che salvano una giornata. In fondo, i corollari della casa. Non eviterete i guasti. Ma almeno saprete come si chiamano.

CONTIENE

- [x] 133 leggi di Murphy commentate
- [x] 8 capitoli, 8 consigli veri del sistemista
- [x] 10 corollari inediti della casa
- [x] 133 illustrazioni pixel
- [ ] garanzia che non si rompa niente

GRATIS. COME I GUASTI.



DEPOLISILVANO.IT