di Raffaello De Masi

Appunti di programmazione

Alla larga dalle bombe

C'è qualcosa che forse terrorizza il programmatore Mac ben più della peste nera per l'Europa del Medioevo; la bomba di sistema. Ad un certo punto, magari nello stesso punto dove prima le cose filavano lisce, improvvisamente il sistema si pianta, apparentemente non c'è più possibilità di comunicare col calcolatore e l'unica possibilità che rimane è quella di resettare il sistema; la cosa può essere davvero spiacevole se si sta lavorando con un linguaggio interprete e non si ha avuto l'accortezza e il buon senso di salvare il sorgente (o le modifiche ad esso apportate) prima del test del programma

Non è tutto; magari nel nostro programma siamo capitati in un loop senza fine e la macchina non reagisce al break. L'unica soluzione è quella di usare il pulsante di interrupt, ma la Apple non si è mai sognata di spiegare cosa fare dopo l'uso di questo famigerato pulsante. Per il buon programmatore non specialista, che non sa cercare immediatamente la soluzione nel monumentale Inside Macintosh, tanto valeva spegnere l'apparecchio.

Eppure ci sono fior di programmi, di un mega e più, che non vanno in bomba neppure a prenderli a calci. Come fanno Word 4 o Excel o il famigerato Mathematica, dalle ben 250.000 linee di sorgente, a non «bombardare» mai? Semplice, non perché non si trovino in condizioni da andare in bomba, ma solo perché chi ha scritto l'applicazione ha previsto anche la possibilità di maneggiare le bombe stesse.

Come divenire quindi provetto artificiere e scansare queste impuntature di sistema che ci tolgono la salute a secchi? La soluzione è, come al solito, molto più semplice di quanto non si immagini.

Io, che l'arteficiere l'ho fatto per davvero, da militare, ho imparato ad applicare un principio che il mio capitano istruttore, persona di acuta intelligenza e di spirito woodehousiano, usava ripetere in ogni occasione «Conoscere la forza del nemico è già mezza vittoria; l'altra metà viene dalla conoscenza della sua debolezza. Perciò prima di mettervi a smontare un ordigno, anche il più semplice e noto, prendete i vostri manuali e ripassate la lezione». Qui non pretendo mica di dare lezioni ad alcuno, ma certo ricordare insieme i principi che animano gli errori non farà male a nessuno.

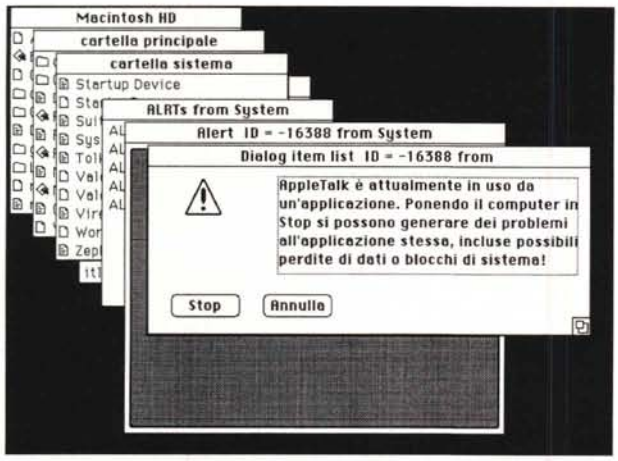
Facciamo un esempio; cosa succede se il nostro programma tenta di scrivere su un dischetto ormai pieno? O peggio su un disco rovinato? Probabilmente va in bomba! Ma perché invece programmi come Draw II, o lo stesso Word, educatamente ci avvisano del malfunzionamento e ci invitano a provvedere? A chi fa word processing in maniera giornaliera sarà capitato probabilmente, almeno una volta nella sua vita che Word spari la sentenza «Unrecoverable Disk Error», ma il sistema non si pianta e soprattutto il documento su cui stiamo lavorando non va perduto!

Per noi programmatori non olimpionici, è possibile dotare le nostre creature di queste comode possibilità? La risposta è «Nella maggior parte sì, e spesso con poca fatica!»

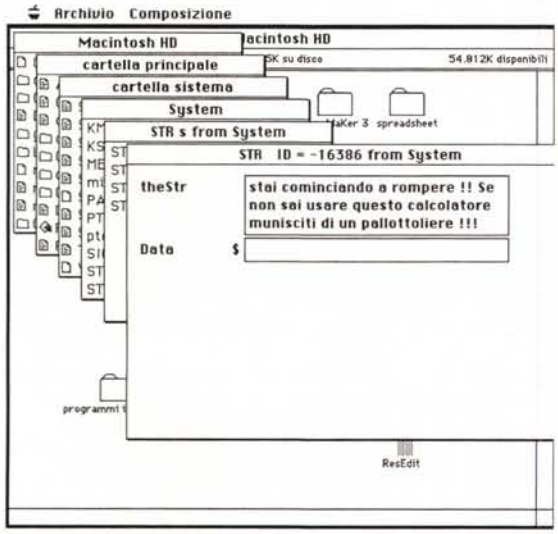
Le cause del «bombardamento» sono riassumibili in tre forme. Nel primo caso (il più comune) lo scoppio è dovuto ad un errore di gestione di memoria. Il programma perde la traccia della sua dislocazione in memoria e i dati da esso manipolati si disperdono nel firmamento dei Kbyte e Mbyte come una cento lire nell'universo. Addio, senza possibilità di ritrovamento. Quando il programma non sa più dove ricercare i suoi dati, bum! e la bomba salta.

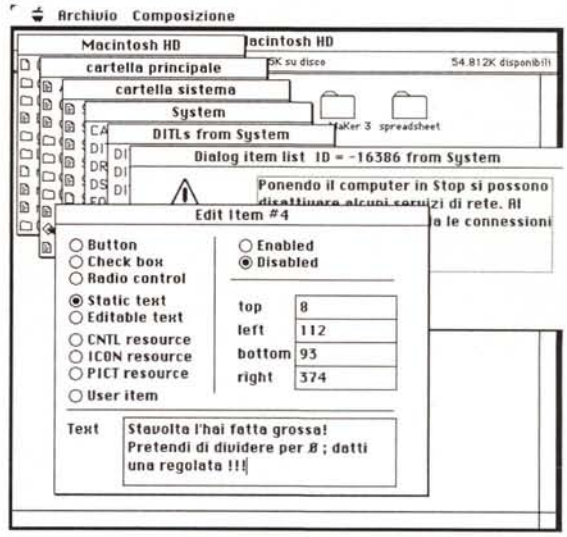
La seconda causa più comune di bomba è un errore di I/O. Causa piuttosto comune è, ad esempio, il tentativo di aprire un file già aperto. Se non sono abilitate le routine di trapping dell'errore (routine, per la verità gestibili anche al più semplice dei linguaggi) ecco che il programma non sa che fare, e salta!

Terza causa più probabile è un errore del resource manager; ad esempio il programma tenta di accedere ad un dialogo, ma questo non è disponibile o è danneggiato, o magari è stato reindirizzato o modificato in maniera non cor-



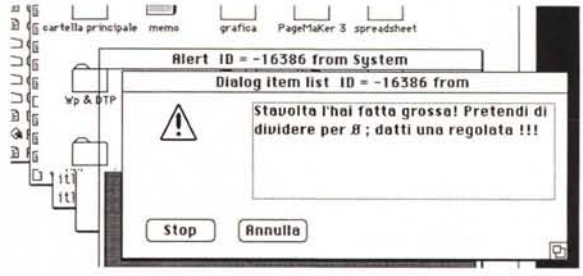
a





b

Alcuni esempi di dialog box, di sistema (a) e personalizzati (b) con Resource Editor. Si ricordi che è possibile comunque costruirsi DB di qualunque tipo utilizzando il Resource Maker, fornito con la maggior parte dei linguaggi in circolazione.



retta dal programmatore; non resta che resettare!

Tutto ciò può essere evitato, o almeno il programma può essere istruito a superare l'evento esplosivo e a mostrare una educata finestra di dialogo, che avvisi dell'errore e proponga una o più alternative. Nei casi disperati, almeno, può essere meno frustrante vedere il programma che avvisa di un errore irrecuperabile e trasferisce il controllo al Finder. In tutti questi casi la soluzione sta nella scrittura di una routine di maneggio dell'errore.

Il programmatore deve costruire la sua bella routine di manipolazione dell'errore e usarla ogni volta in cui teme problemi. Si tratta di qualcosa di molto semplice gestibile attraverso la sola chiamata di [ON ERROR GOTO/GOSUB], per dirla in Basic; la routine di invio potrà poi manipolare il relativo errore in base al numero di ID. Ovvio che la gestione dello stesso sarà poi funzione del tipo di errore individuato. Si andrà così dalla costruzione del messaggio «Disco Pieno, inserire un altro disco», a errori talmente complessi per cui sarà necessario ritornare al Finder. Ma vogliamo evitarvi anche questa piccola fatica.

Una volta scansata la bomba dell'errore e individuato il suo numero di identificazione ci viene in aiuto il Sistema Operativo; affideremo tutta la gestione dell'errore al Resource Manager, che chiameremo in causa con la funzione ResError. Ma come fare, praticamente? Poiché l'errore è identificato da un numero intero, passeremo questo alla routine e utilizzeremo un semplice loop (o meglio una istruzione [CASE]) per indirizzare il problema verso la soluzione. Tenendo conto che l'Error Manager contiene già la gran parte delle soluzioni ai problemi (in base al tipo d'errore presente ha già pronta la finestra di dialogo relativa, con il messaggio appropriato, e spesso, con uno o più pulsanti per indirizzare il successivo flusso del programma); sarebbe davvero fatica sprecata mettersi a riscoprire l'America; al massimo, cosa che faccio sempre io, potrebbe essere simpatico manipolare i dialog box di errore, inserendo qualche nota personale (o magari una parolaccia all'indirizzo degli utenti più sprovveduti). Non solo, ma in fase di realizzazione del programma, una serie di output sullo schermo costruiti in base agli errori che man mano si verificano (ad esempio «Attenzione; ho problemi nel ridimensionare una window» oppure «Hai dimenticato di accendere la stampante!») rende il lavoro di debug molto più semplice e facile. È importante tenere conto che senza routine di maneggio dell'errore, comunque, il programma è pieno di mine vaganti pronte a scoppiare.

Ma per ritornare a quello che diceva il capitano di prima, impariamo un poco a conoscere gli errori; come dicevamo certi sono indipendenti dalla volontà del programmatore; per fortuna sono pochi rispetto a quelli maneggiabili, ma occorre farne menzione perché addirittura possono impedire l'utilizzo stesso della macchina. Approfittiamo quindi della occasione per descrivere un po' tutti gli errori propri del nostro beneamato «melone».

Oltre a quelli di cui abbiamo parlato precedentemente (che per comodità chiameremo errori della Applicazioni), esistono anche errori di lancio e errori di Sistema. I primi sono veri e propri guasti e impediscono di far partire addirittura la macchina. Sono noti in quanto evidenziano la faccia di un «Sad Mac», una iconetta nera, dalla faccia triste, che, nelle ultime macchine, è accompagnata da un motivetto stile carillon che ha il solo effetto di far saltare ancora di più i nervi. Gli insuccessi di lancio possono avvenire per diverse ragioni, primo tra cui un error del test diagnostico interno. Alta causa è l'uso di un System non corretto (come avviene ad esempio lanciando l'FX con il System 6.04; ricordo che Apple ha realizzato per la macchina top della sua serie un sistema ad hoc); fino a qualche tempo fa (oggi una routine d'errore, appunto, permette di evitare lo spegnimento della macchina) lo stesso insuccesso nel bootstrap accadeva anche tentando di lanciare il sistema di un disco privo di System. Ancora altri fattori possono essere un guasto del Finder, un settore guasto sul disco, o (non ridete, non è una possibilità peregrina) il bottone di interrupt schiacciato da qualcosa presente sul tavolo di cui non ci siamo accorti (questa possibilità può portare a guasti anche gravi della componentistica della motherboard).

Un errore, anzi per essere più esatti, un insuccesso di sistema avviene quando il software di gestione presenta guasti o quando il sistema di gestione della memoria interna accusa qualche problema. Diverse sono le cause che determinano errori di tal fatta. Eccone un riassunto, con i relativi codici.

- **ID 01** [errore di Bus]; questo errore avviene quando un programma tenta di accedere a una locazione di memoria non valida o inaccessibile. Si tratta di un errore più frequente nella serie 2 e SE/30 ed è determinato dal tentativo del sistema di leggere (o scrivere) su locazioni di RAM non disponibili (o, come dicevamo, inaccessibili).
- **ID 02** [errore di indirizzamento]; è uno degli errori di sistema più frequenti in assoluto e ha cause diverse, talora inspiegabili. In termini di programmazione il sistema tenta di indirizzare o comunque di maneggiare word rispettivamente 16 e 32 bit su indirizzi dispari; il sistema, in questo caso, individua questo indirizzo come spurio e va in bomba.
- **ID 03** [istruzione illegale]; il microprocessore (Motorola 68000, 68020 o il recente 68030) ha ricevuto un op-code, termine difficile per indicare codice operativo, o semplicemente comando) che non è compreso nel set di ordini comprensibili.
- **ID 04** [divisione di zero]; qui le scuse del programmatore sono poche; c'è stato un tentativo di dividere lo zero per un valore; il caso più diffuso, specie nei linguaggi privi di definizione di variabile, è quello dell'errato uso del nome della variabile.
- **ID 05** [eccezione di arresto]; è una condizione derivante da diverse cause; generalizzando si possono riunire queste cause sotto la definizione di «indirizzamento fuori range»; l'esempio più semplice è il tentativo di accesso del 21° elemento di un array dimensionata fino al 20° posto. Si tratta di un tipo d'errore facilmente maneggiabile, comunque, attraverso un linguaggio di programmazione.
- **ID 06** [eccezione di TrapV - Trap-On-Overflow];
- **ID 07** [eccezione di esecuzione in modo privilegiato]; si tratta di errori dalle caratteristiche complesse; non è semplice qui discuterli senza un'ampia trattazione del sistema operativo MAC; preferibile leggere su Inside Macintosh, la sezione EH (System Error Handler, [ERROR/SYS]) di Bradley Hacker, per ulteriori informazioni; in ogni caso, per quel che c'interessa, non si tratta di errori facilmente recuperabili.
- **ID 08** [eccezione di Trace]; il chip della famiglia dei 68000 ha la capacità di eseguire il tracing delle operazioni che esegue, ma si tratta di una operazione non priva di rischi in quanto interferisce con l'esecuzione del programma, in maniera più pronunciata a seconda della velocità del processo stesso.
- **ID 09** [eccezione di linea 1010]; in

codice esadecimale [A], tale linea consente accesso alle ROM di sistema attraverso un «trap». Se il processore non può eseguire la routine di Trap si va in errore (le ROM di Mac sono accessibili attraverso l'uso di una tabella di vettori di trap, che sono volta per volta richiamate attraverso una istruzione del microprocessore che inizia con l'esadecimale «A»).

- **ID 10** [eccezione di linea 1111]; in codice esadecimale [F], esegue operazioni molto simili a quelle descritte nel precedente punto. La differenza sta nel fatto che i trap [F] non sono direttamente accessibili dal programmatore, per cui un errore di tal genere potrebbe essere sintomo di difetto del sistema o, come nella maggior parte dei casi accade, di errore sulle ROM statiche. Si tratta, per quanto detto in precedenza, di una eventualità molto rara.
- **ID 11** [eccezioni varie]; raccoglie una serie di eventi non altrimenti definibili.
- **ID 12** [routine non implementata]; sembra un assurdo in termini; il sistema tenta di accedere a una routine di trap che invece non esiste. Errore piuttosto raro, avviene quando il processore accede a un trap che, per difetto di definizione o indirizzamento, non è comparabile con nessuno di quelli definiti nella tabella dei vettori.
- **ID 13** [interrupt spurio]; per un istante sistema operativo, clock e processore non sono stati sincronizzati; perché avviene è ben difficile dirlo. Purtroppo c'è poco da fare; è solo un cattivo colpo di fortuna.
- **ID 14** [errore di I/O di sistema]; poco da spiegare; basta per incapparci anche il semplice fatto di aver lasciato spenta la stampante o il plotter; è facile in ogni caso, scrivere routine per svincolarsi da questo errore (l'ON ERROR serve anche a questo).
- **ID 15** [errore di carica di segmento]; come è noto Mac non mantiene in memoria tutta l'applicazione (perla verità non mantiene neppure il System per intero) ma la divide in parti e carica e scarica i pezzi che ci servono a seconda della bisogna; per motivi sconosciuti il programma non è stato capace di caricare il segmento successivo; sovente non viene neppure evidenziato l'errore e il sistema passa semplicemente al Finder (a questo tipo di errore va riferito il dialog box [«L'applicazione XXXX è stata inaspettatamente chiusa»] che talora compare nell'uso di certi programmi non proprio mingherlini). Per inciso è uno degli errori che, più di tutti, viene determinato da infezioni da virus.
- **ID 16** [floating point error]; errore matematico per la verità molto raro e oggi, con i System più recenti, definitivamente scomparso.
- **ID 17-24** [errori di carica del package principale]; questo errore, anch'esso non molto comune, accade quando il sistema tenta di leggere speciali sezioni del sistema operativo chiamate «packages». Molto più diffuso con i sistemi operativi precedenti al 3, è praticamente sparito e si innesca solo talvolta quando si manipolano risorse piuttosto complesse.
- **ID 25** [errore di allocazione di memoria]; il programma richiede maggiori aree di memoria, ma il sistema non ne trova disponibili; comune nelle macchine più piccole quando usavano un ben noto programma di grafica, dimostra una certa disattenzione, da parte del programmatore, nel tenere traccia della memoria e della relativa allocazione.
- **ID 26** [errore di caricatore di segmento]; indica il tentativo di lanciare un programma senza un numero 0 di codice di risorse. Caso ben strano, in quanto un programma, senza risorsa 0 non è un programma (mi si perdoni il bisticcio di parole).
- **ID 27** [mappa del file distrutta]; ben poco da commentare.
- **ID 28** [errore di overflow dello stack]; difficile da spiegare in due parole a un non programmatore; si può riassumere dicendo che due aree di memoria (Stack ed Heap) sono entrate in collisione (sovrapposizione).
- **ID 29-40**; non usati.
- **ID 41** [errore di Finder non trovato]; si è tentato di passare il controllo a un disco privo di Finder.
- **ID 100** [errore di volume]; il file di sistema è danneggiato o non presente.

L'ultima categoria di bombe dipende da errori specifici di applicazione; sono quelli più facili da evitare tenendo conto che pressoché tutti i linguaggi possiedono tecniche di trap d'errore capaci di maneggiarli tutti o quasi.

Si tratta di 23 errori, per la maggior parte legati a operazioni di I/O. Nella più parte dei casi l'uso di chiamate del tipo «ON ERROR» e di analisi dell'ERRN e ERRL (tipo di errore e relativo numero di linea) permette di bypassare agevolmente l'impasse dell'errore stesso.

- **ID 33** [directory piena]; il dischetto (MSF) o la cartella (HFS) hanno esaurito lo spazio di header che contiene i titoli e gli indirizzi dei file.
- **ID 34** [disco pieno]; non esiste più spazio fisico sul dischetto.
- **ID 35** [volume inesistente]; il disco cui si riferisce il file non esiste.
- **ID 36** [miscellanea di errori di I/O]; errori non altrimenti specificati.
- **ID 37** [errore di I/O]; nome del file errato.
- **ID 38** [errore di I/O]; tentativo di leggere o scrivere su un file non aperto.
- **ID 39** [errore di I/O]; tentativo di leggere oltre la fine del file.
- **ID 40** [errore di I/O]; tentativo di spostare il puntatore prima dell'inizio del file.
- **ID 41** [non utilizzato].
- **ID 42** [errore di I/O]; tentativo di aprire troppi file.
- **ID 43** [errore di I/O]; file non trovato.
- **ID 44** [errore di accesso fisico al disco]; il disco è protetto dalla scrittura.
- **ID 45** [errore di I/O]; il file è bloccato (locked).
- **ID 46** [errore di I/O]; il disco è bloccato (locked).
- **ID 47** [errore di I/O]; il file è tuttora impegnato in un altro compito; es. tentativo di aprire un file testo con un word processor mentre lo stesso è già aperto da uno spreadsheet (in Multifinder).
- **ID 48** [errore di I/O]; tentativo di creare due file con lo stesso nome o numero.
- **ID 49** [errore di I/O]; tentativo di aprire due path diverse per lo stesso file.
- **ID 50** [errore di I/O]; errore su blocco di parametri (più comune su file random).
- **ID 51** [errore di I/O]; il numero di riferimento del file non è corretto (raro).
- **ID 52** [non usato].
- **ID 53** [errore di accesso fisico al disco]; il disco chiamato non è presente in alcun drive.
- **ID 54** [errore di I/O]; tentativo di scrivere su un file protetto.
- **ID 55** [non usato].
- **ID 56** [non usato].
- **ID 57** [errore di I/O]; il dischetto non è formato Macintosh.
- **ID 58** [non usato].
- **ID 59** [errore di I/O]; tentativo di cambiare nome a un file guasto.
- **ID 60** [errore di I/O]; la directory principale è guasta.

Termina qui l'ultima categoria di errori; speriamo di aver potuto dare una mano al programmatore; comunque, con questo articolo sottomano si potrà avere almeno la soddisfazione di sapere di che morte (bomba) è morto in quel momento Mac.

MC