La programmazione del Mac

Le Risorse

di Raffaello De Masi

Risorsa, chi è costei? Il fatto è che di questa «cosa» oscura, così legata al mondo della programmazione Mac, tutti parlano, talvolta a sproposito, ma quando si va a cercare, sotto sotto, cosa siano le risorse, come si usano e come si fa a crearle, ben pochi sono capaci di tenere un discorso esauriente

Ho sempre odiato dal profondo del cuore i cosiddetti «santoni» o «piccoli padreterni» che dall'alto della loro saggezza ammanniscono scienza ai comuni mortali. E ho sempre dubitato di chi parla senza farsi capire, visto che ho poi avuto, in dimostrazione, che molto spesso le grandi parole nascondono nel 75% dei casi, ignoranza, e nella restante parte, volontà di non dire. Poiché parto dal principio, basilare in ogni episodio scientifico, che qualsiasi cosa, fatta una volta, può essere ripetuta una seconda volta, non vedo come mai chi è cotanto depositario di scienza non possa essere eguagliato da chi possiede volontà e pazienza.

Nel mondo Mac, stranamente, le risorse sono divenute il babau dei programmatori di primo pelo, e il cavallo di battaglia dei package commerciali. La maggior parte degli utenti sa solo che attraverso le risorse è possibile convertire e «tradurre» programmi, anche grazie ai tentativi e alle prove che ha eseguito giocherellando con il Resource Editor. Oltre questo generalmente non si va. Per chi proviene dal mondo MS-DOS comunque, il concetto di «risorsa» è strano, talvolta tenebroso, imprevedibile, ma quasi sempre innaturale nell'ambito di qualunque linguaggio sia utilizzato. Questo concetto strano, talvolta inconsistente, di programmare attraverso risorse, è davvero difficile da inserire nel lineare mondo della programmazione, specie se procedurale. E così diversi programmatori, specie all'inizio, di fronte a un concetto per certi versi alieno, fanno a meno di questa possibilità; attenzione, ci sono fior di programmi scritti senza far uso di risorse, ma si tratta solo di eccezioni. Probabilmente lo stesso programmatore, se avesse avuto più dimestichezza con le risorse, si sarebbe risparmiato molte notti insonni o imprecazioni verso santi che non hanno alcuna colpa (d'altro canto, ai loro tempi, i computer non esistevano).

Ma cosa sono le risorse?

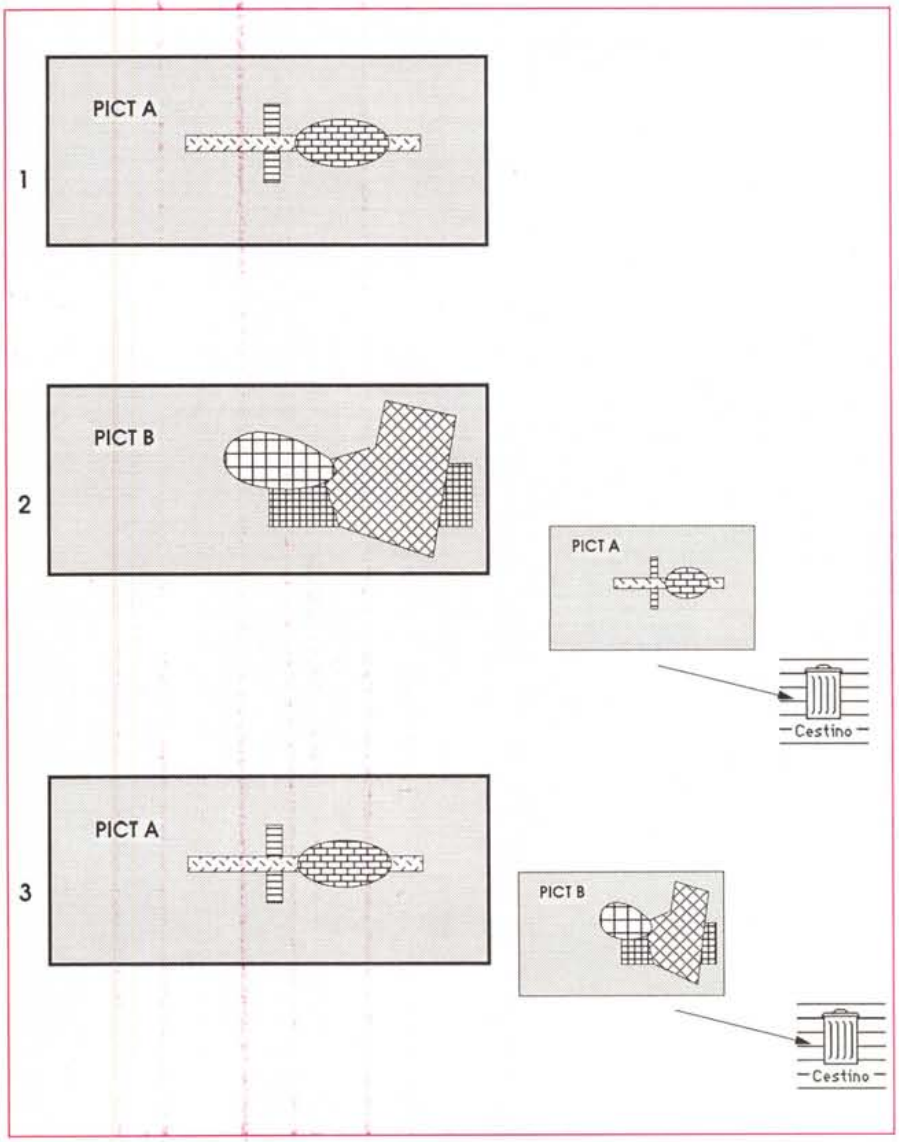
Consideriamo per un attimo il seguente scenario. Stiamo costruendo un programma in MS-DOS che conta le pecore che passano sullo schermo (si fa per dire); a un certo punto ci accorgiamo che inserendo due o tre routine in linguaggio macchina si accelera in maniera anche notevole la velocità di passaggio degli ovini. inoltre occorrerà manipolare tre o quattro file grafici (le figurine delle pecore in movimento) un paio di file dati, e magari le librerie di run-time necessarie al compilatore. In altri termini ci sono da manipolare almeno 10-15 «piece» diversi, diversamente articolati e connessi tra di loro (oltre ovviamente al programma principale) e la cosa più strana è che i diversi file devono essere tutti nel giusto posto perché il programma funzioni correttamente.

Così, se il programmatore cambia il nome delle directory comprese nel pacchetto, il programma non funziona più; se non c'è abbastanza memoria per contenere tutto il necessario, il programma non funziona più; se il programmatore dimentica di inserire al giusto posto il file di contorno, il programma non funziona più.

In altre parole, siamo nei guai; e allora cominciamo a sudare; caccia all'errore, monitoraggio della memoria, controllo costante dei file, sono tutte operazioni che fanno perdere tempo e spezzano la fibra più tenace, specie quando si pensa che il tempo perso lo si poteva utilizzare in ben altro modo. Male, molto male, a questo servono le risorse.

Qualche esempio di utilizzo

Il concetto di risorsa non è difficile da comprendere, ma occorre familiarizzarsi con alcuni concetti che, almeno all'inizio sono un poco alieni alla mentalità di un



Il compito del Resource Manager, nel caso di risorse cancellabili, in situazioni di ridotta disponibilità di memoria.

programmatore classico. Giusto per dare una definizione, una risorsa è una «piece» di programma che pur facendo parte del programma principale vive una sua vita autonoma e indipendente. La similitudine più prossima mi pare quella delle scialuppe di salvataggio su una nave; esse sono indispensabili al corretto funzionamento della struttura, ma non per questo creano intralcio o problemi alla nave principale; probabilmente di esse non ci si accorgerà se non nel momento del bisogno, possono vivere di vita propria, possono essere modificate senza creare problemi alla struttura principale, possono, addirittura essere sostituite o eliminate. Esse hanno una serie di caratteristiche che le rendono particolarmente utili ed efficaci. Vediamo qualche esempio di campo di applicazione.

a) Gestione di memoria parzialmente virtuale. Non stiamo parlando della virtual memory del System 7; la gestione virtuale della memoria da parte dei programmi è cosa nata insieme al Mac, nel 1984: è semplicemente un metodo di consentire l'esecuzione di un programma che è troppo grande per risiedere completamente in RAM (il sistema operativo attinge a parti del programma che risiedono sul disco, ma non si tratta della tecnica, più rudimentale, del chaining). Immaginiamo di avere un programma contenente un numero elevato di risorse PICT (risorse contenenti un file grafico) che materialmente non possono risiedere nella memoria centrale del sistema. Come fare? È necessario riscrivere il codice eliminando tutte le PICT; o fare la corsa parallela con il sistema, monitorando continuamente la memoria fino a caricare una PICT alla volta?

Questo è vero su un'altra macchina, magari corrente in MS-DOS, ma non su Mac. Tutto si risolve costruendo risorse PICT, del tipo «purgeable» (cancellabili). È come dire: «Signor Resource Manager, ti affido una serie di file; di questi alcuni mi servono solo in qualche occasione. Così se durante l'esecuzione del programma alcune parti non ti servono, e sei a corto di memoria, puoi certamente cancellarli; l'importante è che alla bisogna, tu sappia dove cercarli.»

Che cosa avviene in una situazione di ristrettezza di memoria? Vediamo la figura di questa pagina; Il Resource Manager esegue una serie di operazioni di controllo di questa, e scambia, cancellando e ricuperando dal disco, file diversi (dati, PICT, file binari), il tutto gestendo nella maniera migliore la memoria disponibile. L'operazione è assolutamente trasparente; l'unico compito del programmatore è quello di avvisare il programma di cosa ha bisogno e quando e quanto ne ha bisogno. Così, nella figura 1 vediamo come le varie PICT vengono caricate e cancellate a seconda della necessità.

Ma l'operazione di travaso e cancellazione è obbligatoria? Niente di tutto

Il formato di una risorsa STR#

byte offset	dati
0 - 1	0 2
2 - 4	M C
5 - 18	M i c r o c o m p u t e r

La struttura di una risorsa STR.

questo! Se lo stesso utente aumenta la memoria centrale della macchina, il Resource Manager riduce la sua attività in proporzione. Le PICT caricate saranno, quindi sempre presenti in memoria. Ma il Resource Manager è tanto «intelligente» da rendersi conto, fin dall'inizio, se ha a disposizione la necessaria memoria. Così, può accadere che, fin dall'inizio, il programma venga caricato direttamente in memoria con tutti i suoi ammennicoli. Ovviamente in questa situazione ideale, il programma opererà nella maniera più veloce e efficiente.

b) Le risorse permettono una facile modifica dei programmi (come già detto). È l'esempio principe d'uso del Resource Editor, un programma destinato a penetrare nelle risorse stesse e capace di eseguire un editing del loro contenuto. L'uso più frequente, in questo senso, è la modifica delle stringhe ASCII di menu, messaggio e così via. È così che dopo qualche settimana dall'uscita negli Stati Uniti, Word, o Excel, O FullImpact o PageMaker vengono messi in vendita completamente italianizzati. Ma non basta! Attraverso la modifica delle risorse è possibile realmente cambiare il codice di un programma stesso.

Volete un esempio? Semplice. Secondo voi come fa Word ad aggiungere comandi al menu quando invochiamo la combinazione Shift-Command +, o scegliendo il comando Add Command? Semplice, una routine automatica di modifica delle risorse manipola il codice MENU creando nuove entry per le chiamate create.

c) Addirittura, è possibile che un programmatore usi risorse senza neppure accorgersene. QuickBasic e ZBasic, ad esempio, ma anche diversi altri linguaggi, creano automaticamente, al momento della compilazione, un gruppo di speciali risorse di tipo CODE che possono essere aperte dal solito ResEdit per essere customizzate. Non sarà certo lo stesso che introdurre direttamente risorse nel codice sorgente, ma per cominciare è già qualcosa.

Le risorse nella struttura di un programma

Ogni file Mac, sia esso del tipo APPL, TEXT o DUMB, è formato di due parti principali, più o meno prevalenti l'una sull'altra. Queste due parti, chiamate in gergo «fork» (fork, lo ricordo, significa forcella, forcina, ma anche puntello), hanno un significato più teorico che effettivo, ma sono utili per afferrare alcuni concetti di base della programmazione Mac in maniera rapida e immediata.

In qualunque file, questi due fork sono, teoricamente, sempre presenti; ad essi sono stati dati i nomi di «data fork» e «resource fork». Nelle altre CPU il concetto è peregrino, in quanto si hanno solo «data fork», ma qui è davvero fondamentale. Ambedue i fork possono essere vuoti o contenere dei dati. Così è possibile avere file con dati solo nel «data fork» (come accade, per esempio, nei TEXT file) e file (ma è un caso più raro) dove il data fork è vuoto e il resource fork contiene diverso materiale. Ma il caso più diffuso è quello dove ambedue i «contenitori» sono più o meno pieni. E così a questa categoria appartengono la maggior parte dei programmi che affollano i nostri hard disk (word processor, database, applicazioni grafiche e così via). Addirittura è possibile avere lo stesso tipo di materiale sia nel resource che nel data fork (un esempio è il ritratto, digitalizzato, di Andrew Gariepy nella finestra d'apertura di Zbasic, dove stranamente il file PICT, che penseremmo essere nella sezione «data fork» sta invece nel «resource fork».

ICON	CODE
ICN#	DCOD
PICT	CDEV
TEXT	FREF
cicn	BNDL
STR#	STR
snd	ALRT
MENU	DITL
WDEF	CDEF

Alcuni tipi di risorsa, generalmente presenti in tutte le applicazioni.

La migliore risposta alla domanda «Perché ho bisogno di una risorsa» la si può dare ponendoci la domanda «Che bisogno ho di un blocco di dati?»

Rispondere a questa domanda in modo semplice è aver già dimezzato i problemi connessi con la manipolazione delle risorse stesse. Credo che la definizione più semplice di risorsa possa essere questa: è un blocco di dati messo a disposizione del Resource Manager perché li manipoli secondo la nostra volontà. L'unica complicazione sta nel fatto che le risorse non sono di un solo tipo, ma possono essere definite in maniera diversa in modo da consentire il più efficace maneggio da parte del Resource Manager stesso.

Ovviamente ogni tipo di risorsa, a seconda delle sue caratteristiche, va maneggiata in maniera particolare. Così, se si desidera creare una risorsa del tipo stringa (STR#) la prima parola dati, nel blocco dei dati, è il numero di caratteri-byte contenuti nel blocco stesso. Immediatamente dopo c'è la lunghezza, in byte, della prima substringa, c'è poi la rappresentazione ASCII della stringa stessa. Segue la lunghezza della seconda substringa, la seconda substringa, la lunghezza della terza, la terza, e così via.

Apple ha provveduto a regolamentare la struttura e l'uso di alcune risorse di carattere generale (vedremo che è possibile costruirsi risorse ad hoc, per i più bravi!) In base a questo principio, e per evitare confusioni tra diversi implementatori, Apple ha imposto delle regole di comportamento, nell'ambito di questi standard, così tutte le risorse di un particolare tipo, ad esempio ICON, sono strutturate e organizzate nella stessa maniera. Le risorse di uso più comune sono elencate nella figura.

Ma di risorse ce ne sono molti altri tipi. Nessuna fretta per adesso. Avremo modo di vedere qualcosa di interessante la prossima volta.

"Killer Price And A Terrific Warranty"

PC Word - giugno 1990

Direttamente dagli USA

GARANZIE:
* 5 anni in laboratorio (mano d'opera)
* 2 anni su scheda base e parti speciali (sk. VGA Orchid, tastiera, HD Maxtor, ecc.)
* 1 anno sulle parti standard
* Soddisfatti o rimborsati entro 30 giorni dall'acquisto

PC MAGAZINE EDITORS' CHOICE

INFO WORLD TOP 100

"Excellent value"
"An impressive package"

Poly 386 - 25/Cache
* 2 MB RAM
* 80 MB Hard Disk
* Sk. graf.VGA
* Monitor Mono 640x480 VGA
* contenitore tower

Lire 2.910.000

INFO WORLD

"Compared with the top 18 manufacturers. Polywell ranked fastest and lowest priced..."

Poly 386sx - 25
* 1MB RAM
* 40MB Hard Disk
* Sk. graf.VGA
* Monitor Mono 640x480 VGA

Lire 2.060.000

PC MAGAZINE

"Shines in its attention to detail."

Poly 386 - 33/Cache
* 64K Cache
* 4MB RAM
* 120 MB Hard Disk
* Sk. graf. Orchid Pro II 1024
* Monitor colore VGA 1024x768

Lire 4.290.000

"Stands out on the video benchmark tests"

PC MAGAZINE

Poly 486 - 33
* 128K Cache
* 4MB RAM
* 200MB Hard Disk
* Sk. graf. Orchid Pro II 1024
* Monitor colore VGA 1024x768

Lire 6.450.000

Tutti i sistemi sono forniti con licenza MS-DOS e manuali d'uso FDD da 3.5" o da 5,25" porte seriale e parallela.
Altre configurazioni sono disponibili su richiesta.
I prezzi sono IVA esclusa e possono subire variazioni senza preavviso.

Nuovi Annunci e Prodotti Speciali

- 386 il sistema può ospitare le CPU 386/25/33/40, 486sx, 486/25/33/50 consentendo un aggiornamento delle prestazioni a prezzi competitivi.
- Sceda Grafica VGA Edsun (Continue Edge Graphics) rende disponibile una risoluzione fino a 2500x2500 con 700.000 colori su monitor VGA standard.
- 486 a 50 Mhz con 256 Kb di memoria cache. "TOP GUN"
- Scheda Accelleratrice per PS/2 modelli 50/60s. Aumenta la velocità della CPU a 20 Mhz con 32 Kb di memoria cache.

Per informazioni e ordini telefonici chiamateci al numero:

(011) 4373313

Linea diretta assistenza tecnica
tel. (011) 489059

POLY SYSTEMS

Polywell Computers Inc.
61"C" airport Boulevard
South S.Francisco CA 94080 U.S.A.
FAX(415)583-1974

importiamo e distribuiamo esclusivo per l'Italia

M.C.E.

M.C.E. Manutenzioni e Costruzioni Elettroniche srl
Via Capellina 12 - 10144 TORINO
FAX (011) 4730512

M.C.E. in Italia è:
- *Assistenza tecnica*
- *Reti locali*
- *Prodotti d'avanguardia*