La release 1.2

A gentile richiesta da parte di lettori e amici, l'argomento di questo mese riguarda i nuovi comandi CLI presenti sulla release 1.2 del sistema operativo di Amiga.
Non dovremmo parlare così ufficialmente dato che questa non è stata ancora distribuita dalla Commodore Italia, ma ormai la maggior parte degli «Amighi», chi per un motivo chi per un altro, dispone già di una copia arrivata «chissaccome» e aspetta solo di poterla sfruttare a pieno.
A tutti voi «precursori», dunque, buona lettura...

di Andrea de Prisco

Il pacchetto

Trattasi di tre dischi e di un manualetto di circa 80 pagine. I tre dischetti sono etichettati Kickstart 1.2, Workbench 1.2 ed Extras 1.2. Quest'ultimo contiene, ne riparleremo più approfonditamente in altra occasione, la nuova versione del Basic (compatibile con il kick 1.2) e le PC utility con le quali, disponendo di un drive esterno da 5.25 pollici, è possibile formattare dischetti compatibili MS-DOS ed effettuare copie di file tra i due formati eventualmente interponendo dei filtri.

Delle nuove feature offerte dal Workbench 1.2 ne abbiamo già parlato in altre occasioni. Giusto per ricordare qualcosa diremo che ora leggere una directory è assai più veloce di prima, disponiamo del ram disk anche sottoforma di icona e finestra, da "preferences" possiamo impostare il modo interlacciato per disporre di un numero doppio di linee (sfarfallose) e settare, sempre da preferences, le specifiche dell'interfaccia seriale (parità, lunghezza parola, stop bit, ecc.).

Infine il Kickstart 1.2 è lo stesso delle rom delle nuove versioni di Amiga, il 2000 e il 500, e si riconosce da altre versioni precedenti pseudo-uno-punto-due per il fatto di mostrare (vedi foto) la versione sin dal momento di richiedere il Workbench o l'applicativo.

CLI 1.2

L'interfaccia a linea di comando (CLI) disponibile sul dischetto del Workbench 1.2, come detto è stata arricchita con nuovi comandi, alcuni molto utili. Alcuni dei comandi preesistenti, inoltre, sono stati migliorati per adattarsi alle nuove caratteristiche del sistema. Ad esempio è possibile formattare un HD, parte di questo (per l'uso in congiunzione col Sidecar dotato di tale memoria di massa), e dischi da 5.25 se disponiamo del drive esterno 1020.

Proprio per tali dispositivi, sono stati aggiunti comandi come MOUNT e BINDDRIVERS atti a istallare, generalmente nella startup-sequence di un dischetto, nuovi dispositivi, interfacce comprese.

Ancora per i drive da 5.25 è presente l'istruzione DISKCHANGE tramite la quale informiamo il sistema operativo di aver cambiato il disco contenuto nell'unità. Come noto i drive da 5.25 non «sentono» tale operazione, mentre il sistema *deve* sempre essere conscio di ciò che l'operatore combina. Provando infatti a cambiare disco senza farcene accorgere dal sistema, operazione come detto possibile con i drive da 5.25 e simulabile con quelli da 3.5 andando a «smucinare» coi ponticelli interni, abbiamo effetti assai strani, come finestre marchiate con un nome di un disco e contenuto di un altro, e cose simili.

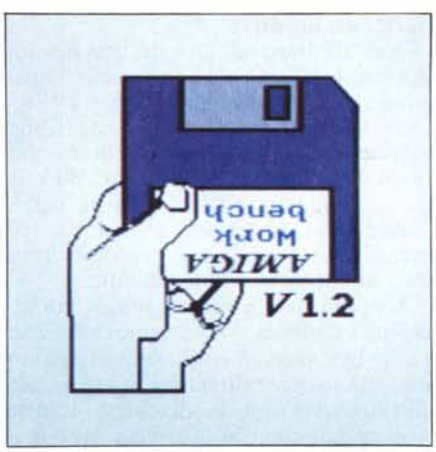
Un'altra possibilità offerta dalla release 1.2 è quella di settare la configurazione della tastiera. Possiamo optare per una tastiera tedesca, spagnola, francese, inglese, italiana, islandese, svedese, danese, norvegese, canadese e standard USA. Ciò indipendentemente dalla tastiera di cui disponiamo. Se ad esempio con la macchina ci è stata fornita la tastiera USA, ma noi siamo troppo bravi con quella italiana (tanto bravi da digitare senza guardarla) non dobbiamo far altro che digitare:

```
SETMAP I
```

per essere accontentati. Analogamente per rendere STANDARD una tastiera maledettamente italiana (io non sarei in grado di usarla) come, da un po' di tempo viene fornita con l'Amiga. In questo caso digiteremo:

```
SETMAP USA
```

Si noti come il settaggio della tastiera non è locale ad un task ma globale per tutto l'Amiga. In altre parole se dopo aver scelto una configurazione apriamo un nuovo CLI anche in questo avremo la tastiera settata precedentemente.



Il vero Kickstart 1.2 si riconosce sin dalla richiesta del workbench.

Path e NEWCLI FROM

Tra le caratteristiche più interessanti di questa release 1.2 non possiamo non annoverare la possibilità di indicare al sistema operativo le directory da esplorare (e in che ordine) per ricercare un comando da eseguire. Dalla versione 1.1 sappiamo infatti che quando digitiamo un comando questo verrà prima cercato nella directory corrente, poi in quella assegnata al device C: (Ci-duepunti) e solo se neanche lì è trovato viene dato un messaggio di errore su video.

Con la versione 1.2 e col comando PATH è possibile modificare il percorso da compiere per trovare un comando.

Indicando PATH seguito dal nome di una directory non facciamo altro che aggiungere al precedente percorso un nuovo luogo.

Ad esempio, se disponiamo di una directory separata NuoviComandi e desideriamo la ricerca anche in questa digiteremo:

```
PATH NuoviComandi
```

Per togliere una o più directory dal path esiste l'opzione RESET che usata

Due momenti di una riparazione dischetto a seguito del comando DISKDOCTOR.

«liscia» resetta il path a C:, se si specifica una directory questa viene semplicemente tolta.

Esempi:

```
PATH RESET
PATH NuoviComandi RESET
```

Infine, per conoscere il path corrente digiteremo semplicemente:

```
PATH
```

Sempre nelle pagine del manualetto leggiamo di una nuova forma del comando NEWCLI, per così dire, programmata.

È possibile aprire un nuovo CLI il quale riceva comandi non da tastiera ma da un file comandi precedentemente preparato.

Non capiamo bene a cosa possa servire ciò, dal momento che col comando EXECUTE, preceduto da un RUN, si ha lo stesso effetto e in più possiamo utilizzare i costrutti come IF, LABEL e SKIP.

L'unica differenza tangibile è che nel caso del NEWCLI viene creata una nuova finestra allo scopo mentre nel caso del RUN EXECUTE la finestra utilizzata è quella dalla quale è partito il comando. Oltre a ciò se l'ultima istruzione del command file non è un ENDCLI, la finestra aperta resta attiva per ulteriori comandi manuali.

Ad ogni modo la sintassi è la seguente: posto che il nostro file di comandi si chiami, guardacaso, Pippo, per eseguirlo in una finestra separata digiteremo:

```
NEWCLI FROM Pippo
```

Attenzione, Pippo deve contenere solo comandi digitabili da tastiera!

Buffer e dottori

Col comando ADDBUFFERS è possibile implementare un buffer RAM per ogni drive. In questo modo l'accesso a disco, se riferiamo spesso ad un insieme ristretto di blocchi, diventa molto più veloce. Informaticamente parlando tale porzione di memoria è detta *cache*, deposito, e la sua funzione è proprio quella di mantenere in memoria principale le parti di memoria secondaria più riferite nell'ultimo intervallo di tempo. L'unità di incremento buffer assomma a circa 500 byte quindi digitando, come consigliato sul manuale per ottenere i primi benefici senza sprecare troppa ram, il comando:

```
ADDBUFFERS DF0: 30
```

utilizzeremo un buffer di circa 15 k per l'unità a dischi DF0:. Allo stato delle attuali conoscenze, non ci risulta o, meglio, sul manualetto non c'è scritto, come fare per diminuire o togliere buffer ad un drive.

Nel titoletto di questo paragrafo, dottore è riferito ad una nuova istruzione, non a caso denominata DISKDOCTOR, con la quale è possibile salvare il salvabile di un dischetto non funzionante (read-write error, disk is unreadable, not validated disk, ecc.).

Abbiamo compiuto alcuni esperimenti su dischi guasti e i risultati sono stati abbastanza soddisfacenti.

L'operazione è un po' lunga, ma alla fine i risultati si ottengono, anche se i vari file «salvati» non appartengono più alla relativa directory ma sono posti tutti nella root del dischetto. Inutile aggiungere che più il disco è incasinato meno sono le probabilità di ripescare tutti i file.

Ad ogni modo, la sintassi è quanto mai semplice, basta indicare il drive nel quale è contenuto il disco difettoso. Ad esempio:

```
DISKDOCTOR DF1:
```

Priorità

Apriamo una breve parentesi. Dedicheremo un po' di righe di articolo al multitasking di Amiga. L'argomento, in generale (a quei tempi di Amiga non ne avevo ancora sentito parlare), è stato già trattato in Appunti di Informatica, ma, voglio essere buono, non vi rimando alla lettura di quegli articoli.

Una delle carte vincenti di questo amatissimo computer, l'ho già detto molte volte e non mi stancherò di ripeterlo, è la possibilità di lanciare più processi parallelamente. I vari comandi RUN, NEWCLI e lo stesso Workbench testimoniano sufficientemente. Ad esempio da Workbench posso caricare un'applicazione e, memoria permettendo, un'altra, poi un'altra ancora e così via. Tanto per raccontarvene una, con l'Amiga 2000 arrivato in redazione lo scorso mese, essendo questo dotato di 1 megabyte di ram abbiamo lanciato contemporaneamente Scribble2, Analyze e MiAmigaFile2 e disponevamo ancora di oltre 200 k di memoria libera. Come dire: «cominciamo a ragionare!».

Tutti sanno però che all'interno di Amiga vive un solo processore, il motorola 68000, il quale, pur essendo attorniato da una nutrita equipe di altri processor dedicati (Agnus, Paula e Denise, loro sottoparti comprese) per sua natura è in grado di eseguire un solo processo per volta. Solo con un

sistema operativo Tanto-di-Cappello come quello di Amiga (anche se ancora non funziona perfettamente) e qualche altra porzione di hardware accessorio si riesce a simulare parallelismo a livello di processi.

In altre parole, in ogni istante un solo processo è attivo (trad.: è eseguito dalla CPU) e resta in tale stato fino a quando non si verifica uno di questi due eventi:

1) Richiesta I/O
2) Quanto di tempo scaduto

Se un processo, avendo richiesto un dato da una periferica, ad esempio l'unità a dischi, deve attendere il suo arrivo, non tiene occupata inutilmente la CPU fino al completamento dell'operazione, ma la rilascia e si accoda (tutte queste operazioni, per essere più precisi, non vengono effettuate dal processo ma dal 68000) nella lista dei processi sospesi, un'apposita struttura dati presente in ogni computer multitask (serio). A questo punto, il processore, libero, preleva un processo dalla lista dei processi pronti (altra struttura, simile alla precedente) e continua l'esecuzione di quest'altro. Grazie alle sospensioni da operazioni di I/O si ha un primo tipo di *ricambio* tra i processi. Oltre a questo, come annunciato, un apposito orologio interno ad intervalli di tempo regolari manda un'interruzione al processore per accelerare il ricambio, ma soprattutto per non contare troppo sulle operazioni di I/O di un dato processo. In altre parole se entro T microsecondi il processo J si sospende a causa di un I/O bene, altrimenti allo scadere del tempo si ha un *ricambio* forzato del processo in esecuzione. Naturalmente in questo caso il processo «segato» non è accodato nella lista dei processi sospesi ma direttamente in quella dei processi pronti: non aspetta nulla dall'esterno per ripartire se non il suo turno di CPU. Manca un solo anello per chiudere la catena: quando il dato in arrivo dal disco è effettivamente arrivato, una nuova interruzione provoca che il processo in attesa (passiva) del dato viene posto nella lista dei processi pronti e quindi anche questo è pronto a ripartire quando sarà il suo turno.

Tutto questo dire per passare al successivo comando CLI 1.2. Si tratta del comando CHANGETASKPRI che, come dice il suo nome, permette di cambiare la priorità di un processo, rispetto agli altri. Come ciò sia realizzato all'interno di Amiga non siamo ancora in grado di dirlo. Generalmente ciò può avvenire in almeno due modi distinti: diversificazione del quanto di tempo oppure lista processi pronti con funzionamento a priorità. Nel primo caso a seconda della priorità del processo viene assegnato un quanto di tempo più o meno grande quando questo viene eseguito dalla CPU. Nel secondo caso, un inserimento nella lista dei processi pronti non avviene in coda a tutti quelli già presenti, ma si mantiene l'ordine della lista inserendo nel punto opportuno: prima del processo con priorità più bassa del nostro e dopo quello con priorità più alta. Si noti che quest'ultimo metodo è troppo penalizzante per i processi a bassa priorità sino al punto di rischiare la ben nota (agli informatici) starvation o attesa infinita (che rende meglio il senso).

Tornando al comando AmigaDOS, esso è locale al CLI in cui lo impostiamo e viene ereditato dai processi creati dal CLI in questione. Ovvero se dispongo di tre CLI, 1,2 e 3, e da CLI 2 digito:

```
CHANGETASKPRI 3
```

(di default la priorità è 0 e può assumere valori compresi tra -5 e 5) il CLI 2 avrà priorità 3 ed avranno priorità 3 tutti i processi creati da questo, col comando RUN o NEWCLI.

Comando SETDATE

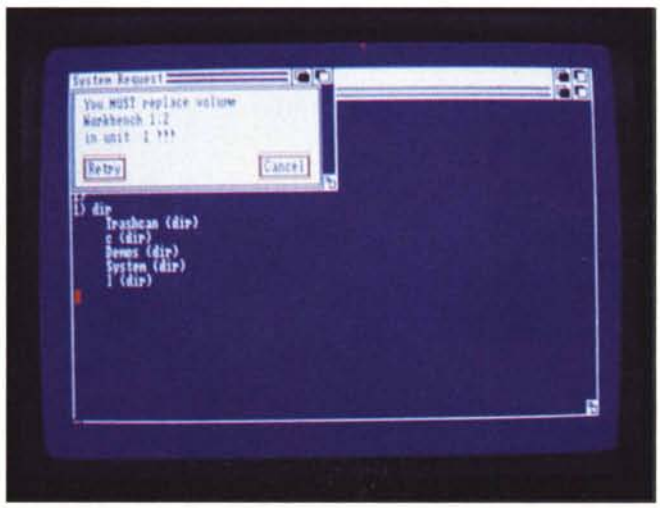
Secondo quanto indicato nel manualetto, questo comando dovrebbe servire per cambiare la data associata ad un file o ad una directory. Il verbo è al condizionale per il fatto che sembra proprio non funzionare. Naturalmente ciò è riferito al dischetto di cui in questo momento disponiamo, può darsi che qualche lettore ottenga risultati differenti.

In ogni caso la sintassi di tale comando è la seguente:

```
SETDATE NomeFile NuovaData
```

dove NuovaData va scritta secondo la sintassi del comando DATE. A noi succede che una volta utilizzato tale comando, il file riferito sparisce dalla lista dei comandi pur essendo vivo e vegeto. Tanto più che digitanto DIR lo vediamo nella directory insieme agli altri file, come se nulla fosse successo. Col comando LIST, non appare più. E non c'è stato verso di renderlo nuovamente visibile se non copiare il file fantasma con un nuovo nome, cancellare quello vecchio e rinominare il file nuovo col nome vecchio.

Amighevoli stranezze... MC



A sinistra il comando PATH. Inseriamo anche la directory SYSTEM nel percorso. A destra, la più simpatica System Request di Amiga. Con la release 1.2 abbiamo provato a togliere un dischetto dal drive mentre era in corso la lettura della directory. Con l'1.1 avrebbe dato un bel GURU MEDITATION.