Turbo Pascal 6.0

di Sergio Polini

Quando la Borland Italia ci ha presentato il Turbo Pascal 6.0, Corrado Giustozzi ed io abbiamo potuto conoscere anche il nuovo product manager americano del Turbo Pascal. Zack Urlocker era stato assunto da Philippe Kahn solo da una ventina di giorni, ma era già un personaggio noto: proveniva dal Whitewater Group, la società che ci ha dato ottimi prodotti per Windows e OS/2 quali il Resource Toolkit, WinTrieve e soprattutto Actor, un sistema completo per lo sviluppo di applicazioni object-oriented in quegli ambienti; aveva scritto diversi articoli sulla OOP per varie riviste, tra cui Computer Language e il Journal of Object-Oriented Programming. Rimasi però piuttosto perplesso: va bene che anche il Turbo Pascal è ormai orientato all'oggetto, ma cosa c'entrano Windows e OS/2? Analoghe perplessità quando, dopo poche parole introduttive, abbiamo visto il prodotto. Una interfaccia utente molto migliorata per l'ambiente integrato, simile a quello del Turbo C++, una libreria di funzioni che consente di realizzare facilmente programmi con quella stessa identica interfaccia (Turbo Vision), ritocchi solo marginali al resto. In particolare, contrariamente a quanto era spesso avvenuto in passato, solo ritocchi marginali alla sintassi del linguaggio.

Ma Urlocker non era venuto da solo. Era come al solito presente David Intersimone, «Director of Developer Relations» della Borland e, in pratica, suo «ambasciatore». Un anno prima, quando era venuto a presentare il Paradox Engine, aveva spiegato a Corrado che la Borland intendeva concentrare le proprie attenzioni sugli ambienti MS-DOS e OS/2, escludendo in particolare Windows, ritenuto troppo poco significativo. Ma il mondo dell'informatica è in continua evoluzione... Ricorderete forse che il numero di MC di giugno scorso riportava l'annuncio della versione 2.0 del Paradox Engine, comprendente tra l'altro il supporto a Windows

3.0, che era stato contemporaneamente presentato dalla Microsoft; ricorderete forse che già allora la Borland aveva annunciato futuri sviluppi di altri suoi pacchetti per il nuovo ambiente grafico. Lo stesso Intersimone confermava l'interesse per Windows 3, per via del successo di questo: ovvia conseguenza di notevoli e sostanziali miglioramenti rispetto alle versioni precedenti (chi non abbia ancora potuto constatare di persona la validità di Windows 3, potrebbe magari rileggersi l'ottima «anteprima» curata da Corrado per quello stesso numero di MC).

Tra le tante innovazioni di Windows 3, vi è una interfaccia utente molto simile a quella del Presentation Manager di OS/2, con un'adesione quasi totale alle specifiche SAA-CUA (Systems Application Architecture — Common User Access) come aggiornate dalla IBM nel 1989. Il numero 1 dell'informatica aveva fin dal 1987 proposto le linee per lo sviluppo di interfacce utente coerenti da sistema a sistema, dal personal al mainframe, dai tradizionali terminali «non programmabili» alle workstation grafiche, ma i ritardi di OS/2 e la scarsa attrattiva esercitata dalle precedenti versioni di Windows non ne avevano certo incoraggiato la diffusione. Abbiamo continuato a vedere programmi dotati ognuno della «sua» interfaccia, abbiamo continuato a dover imparare daccapo per ogni programma con quali comandi aprire un file, chiedere aiuto, immettere dati, stampare risultati, terminare le operazioni. La veloce diffusione di Windows 3 sta contribuendo a cambiare rapidamente questo scenario, in quanto rappresenta, tra l'altro, la veloce diffusione di un ambiente che fa proprie le specifiche SAA-CUA. È vero che non tutti si sono convertiti all'interfaccia grafica, ma sono comunque ovvi i benefici che possono derivare dall'effettiva diffusione di uno standard, anche nel ben più diffuso «modo testo» del buon vecchio MS-DOS. Potremmo aggiungere che analoghi sono i benefici, sia per gli autori che per gli utenti di un programma, della diffusione di quella programmazione «per eventi» che si associa indissolubilmente ad una interfaccia come quella di Windows 3 o del Presentation Manager. Potremmo infine rilevare che la programmazione sotto Windows o Presentation Manager è tutt'altro che banale, al punto che la IBM sembra intenzionata ad includere nella SAA anche lo Smalltalk V/PM della Digitalk, come strumento «raccomandato» per lo sviluppo di applicazioni sotto OS/2, proprio per la maggiore facilità di programmare per finestre ed eventi in modo object-oriented.

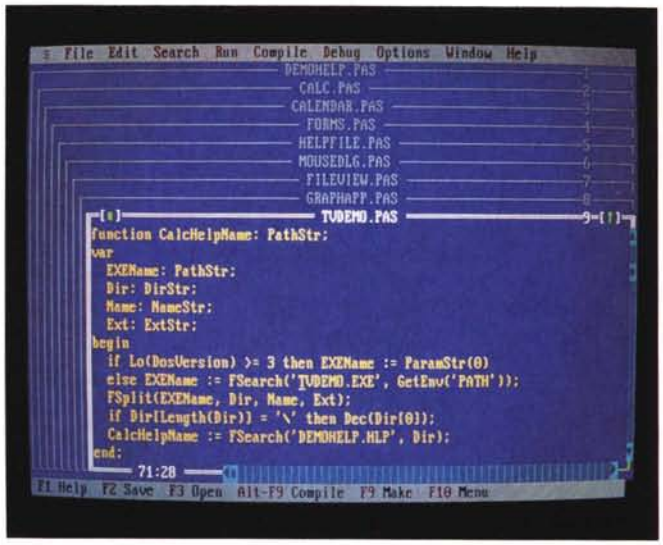
Se questa è la tendenza per i prossimi anni, e se si rileva che il Turbo Pascal era già object-oriented e che il Turbo Vision non aiuta solo a programmare per finestre ma anche per eventi, si può cominciare a pensare che la Borland non ci propone «solo» una interfaccia utente in modo testo simile a quella di ambienti grafici, ma ci propone piuttosto uno strumento che, grazie alla sua capacità di «girare» perfettamente anche sotto il diffusissimo MS-DOS e su macchine meno «ricche» di quelle che i nuovi ambienti richiedono, può rappresentare il modo migliore di prepararsi all'imminente futuro dell'informatica individuale.

Turbo Pascal 6.0

Produttore:
Borland International, Inc.
1800 Green Hills Road
P.O.Box 660001
Scotts Valley, CA 95067-0001
Distributore:
Borland Italia S.r.l.
Via Cavalcanti, 5 - 20127 Milano
Telefono: 02-2610102

Prezzi *(IVA esclusa):*

Turbo Pascal 6.0	*L. 299.000*
Turbo Pascal 6.0 Professional	*L. 498.000*
Upgrade al Pascal 6.0 Pro:	
- da qualsiasi linguaggio Turbo non Porfessional	*L. 299.000*
- da qualsiasi Turbo Pascal Pro	*L. 199.000*



Il nuovo ambiente integrato consente di lavorare contemporaneamente su più file in finestre mobili, ridimensionabili e sovrapponibili.

L'interattività con l'utente avviene mediante dialog box corredate di check box e di radio button.

La dialog box mediante la quale si può cambiare la directory corrente.

Installazione

Il Turbo Pascal 6.0 richiede almeno 256K di RAM per il compilatore separato, almeno 512K per l'ambiente integrato. L'ambiente integrato è in grado di utilizzare l'eventuale memoria espansa (EMS) disponibile per parcheggiarvi sorgenti o codice in overlay, mentre una nuova opzione da dare sulla riga comando («/T») consente di evitare il caricamento automatico del file TURBO.TPL per risparmiare memoria (ricordiamo che TURBO.TPL contiene le unit standard *System, Overlay, Crt, Dos* e *Printer,* oltre a quelle che ripropongono la compatibilità con il Turbo Pascal 3.0, *Turbo3* e *Graph3;* se non si carica TURBO.TPL, tali unit vanno estratte mediante il programma TPUMOVER.EXE per poter essere trovate sul disco dal compilatore).

Viene fornita la solita elegante e facile procedura di installazione automatica, che guida l'utente sia nel caso disponga di un sistema a due floppy che di un disco rigido, quest'ultimo da considerare, come e più che per il passato, ampiamente raccomandabile. Dopo un'installazione completa, infatti, ci si ritrova con un'occupazione su disco complessiva di circa 3,5 Mega; è vero che lo spazio si riduce se si sceglie di non decomprimere gli esempi, forniti in formato ZIP, ma ciò non è consigliabile in considerazione della necessità di averli a portata di mano se ci si vuole familiarizzare con il Turbo Vision. Si deve aggiungere che, per la prima volta, non è possibile installare tutto su un sistema con soli floppy da 360K: il file per l'help in linea occupa infatti quasi 700K.

Accanto ai tradizionali compilatori integrato e separato (TURBO.EXE e TPC.EXE), la versione Professional comprende ora anche il «Turbo Drive», ovvero una versione per memoria estesa del compilatore separato (TPCX.EXE), utile nel caso non si riesca a compilare in 640K ma si disponga di una macchina con almeno un 80286 e almeno un mega di memoria estesa. Chi acquisti la versione Professional riceve anche il Turbo Debugger (che richiede 384K di RAM e 1,4 Mega di spazio su disco, esempi compresi), il Turbo Assembler (rispettivamente 256K e 360K), e il Turbo Profiler (384K e 512K). Il Turbo Debugger, aggiornamento di quello fornito con il Turbo C++ 1.0, comprende le versioni per macchine con processori 80x86 (TD286.EXE e TD386.EXE), che richiedono almeno 512K di memoria estesa. Non aggiungerò altro su tali prodotti, in quanto Turbo Debugger e Turbo Assembler sono già stati esaminati nel numero di settembre ed è imminente la prova del Turbo Profiler.

Rimangono per il resto i requisiti della piena compatibilità con i personal IBM e della disponibilità di un DOS almeno 2.0, mentre si aggiunge la possibilità di usare un mouse, purché compatibile con la versione 6.x o successiva del driver Microsoft.

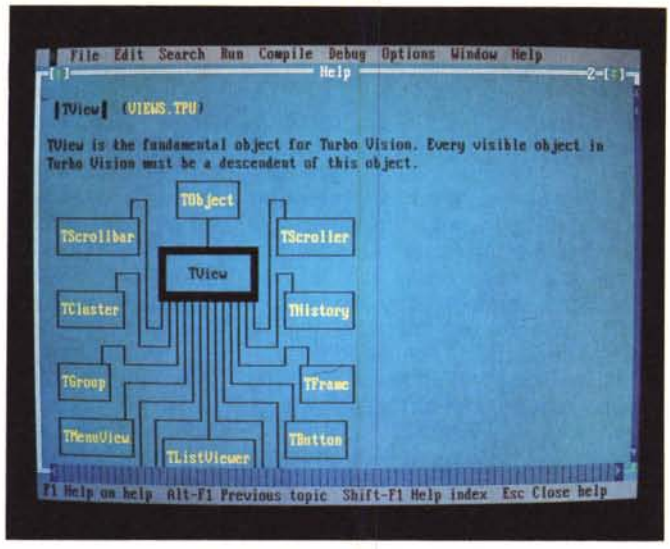
Documentazione

È stata completamente riscritta. Abbiamo ora quattro manuali e cinque file su disco. Il manuale *Guida all'uso* illustra il nuovo ambiente di sviluppo, i fondamenti della programmazione in Turbo Pascal, unit e OOP compresi, l'uso del debugger integrato, le opzioni dei compilatori integrato e separato. Il manuale *Programmazione* approfondisce tutti gli aspetti della sintassi del linguaggio, illustra il contenuto delle unit standard, i meccanismi di gestione della memoria, sia statica che dinamica, il formato interno dei dati (oggetti compresi), le convenzioni di chiamata di una funzione o procedura e la gestione degli interrupt, l'accesso a file di testo o alle porte, le ottimizzazioni del codice, le direttive di compilazione, l'interfacciamento con il linguaggio assembly e i messaggi d'errore. Le procedure e funzioni standard sono illustrate in un apposito volume di riferimento intitolato *Libreria,* mentre al nuovo Turbo Vision è dedicato un volume omonimo, che comprende sia un'efficacissima esposizione introduttiva che una completa guida di riferimento.

L'esposizione è fedele allo standard a cui la Borland ci ha da tempo abituato. Nonostante infatti la struttura della documentazione cartacea sia stata completamente rivista, i contenuti sono stati quasi integralmente mantenuti e aggiornati. Fa ovviamente eccezione il manuale dedicato al Turbo Vision, che merita ogni elogio. La parte introduttiva, devo ripeterlo, è ottima: mette in grado il lettore di entrare pian piano in un modo di programmazione sicuramente più complesso di quanto possa sembrare dalla scorrevolezza dell'esposizione e dalla gradualità degli esempi (disponibili anche su disco); si rimane con il dubbio se programmare per finestre e per eventi appaia semplice grazie alla potenza della programmazione object-oriented oppure grazie all'efficacia dell'esposizione. La guida di riferimento è inoltre strutturata in modo molto intelligente: una prima sezione descrive il contenuto delle diverse unit, una seconda espone in ordine alfabetico tutti i tipi **object,** commentandone scopo e funzionamento, con annotazioni sulla opportunità di ridefinire i metodi e con diagrammi che illustrano graficamente la collocazione di ogni oggetto nella gerarchia; la terza e ultima sezione descrive infine tutti gli elementi (costanti, tipi, variabili, procedure e funzioni) che non fanno parte della gerarchia di oggetti del Turbo Vision. Si sa sempre dove trovare l'informazione che serve.

Si deve tuttavia rilevare che in nessun manuale si trova la tradizionale appendice dedicata alle differenze con le versioni precedenti del Turbo Pascal; se è vero che le differenze nella implementazione del linguaggio non sono molte, va anche detto che si può incappare in un problema tando fastidioso quanto banale come quello del «punto e virgola», di cui vi riferirò tra breve.

Si può anche osservare con qualche preoccupazione il trasferimento su disco di argomenti fin qui ospitati dai manuali cartacei, come la descrizione dei tradizionali programmi di utilità (TPUMOVER, MAKE, TOUCH, GREP, BINOBJ, THELP), nonché del nuovo TEMC (compilatore di macro da aggiungere all'editor dell'ambiente integrato), mentre non ho trovato documentazione alcuna né di EMSTEST.EXE (che esegue un test completo della memoria espansa) né



L'help in linea ripropone sullo schermo buona parte della documentazione cartacea. Qui vediamo cosa si vede se si preme F1 mentre il cursore è su TView. Notate la possibilità di «zoomare» anche la finestra di Help.

di CPUTEST.EXE (di cui posso solo dirvi che sulla mia macchina mi riferisce che «This CPU tests ok»). Mi sarebbe anche paciuto avere su carta la seconda parte del file TVISION.DOC, che comprende preziosi suggerimenti su come organizzarsi per l'overlay delle unit del Turbo Vision e sui principi da seguire nella dichiarazione di nuovi tipi **object** derivati da quelli compresi nella gerarchia. Accetto invece volentieri le usuali integrazioni al manuale contenute nella prima parte di quel file. A questo riguardo, però, devo pure rilevare che, mentre è stato fedelmente tradotto il file HELPME!.DOC (si chiama AIUTO!.DOC e contiene utili indicazioni sull'installazione e su possibili «sviste» nell'uso dell'ambiente integrato o delle nuove caratteristiche del linguaggio), non altrettanto è stato fatto con il file FIXES.DOC, che conteneva correzioni e integrazioni da apportare ai manuali originali. Non aveva ovviamente senso tradurlo così com'era, visto che i manuali sono in italiano (a meno di non richiedere espressamente l'edizione originale), ma sarebbe certo stato utile un file con la segnalazione di quegli errori che inevitabilmente sfuggono nella prima edizione di una traduzione tanto impegnativa, e nella sostanza molto curata; qua e là capitano infatti un «devised» tradotto con «mascherato» o un «right» tradotto con «sinistro», che possono occasionalmente disorientare il lettore.

Non vorrei comunque essere andato oltre le mie intenzioni: la documentazione merita nel complesso un giudizio ampiamente positivo.

Ambiente di sviluppo

Abbiamo qui una delle due principali novità. L'ambiente integrato è stato integralmente rifatto, e ripropone ora quella interfaccia utente «praticamente perfetta» che avevamo già apprezzato nel Turbo C++ 1.0 (MC dello scorso settembre).

Possiamo ora editare i nostri sorgenti in finestre multiple, mobili, ridimensionabili e sovrapponibili, e può trattarsi di file anche molto ingombranti (il limite è di un mega, grazie all'uso di un meccanismo di swap su disco o della memoria espansa); possiamo muovere o copiare blocchi di testo da una finestra all'altra mediante un *Clipboard* e le operazioni di *cut/copy and paste;* l'interattività è facilitata da un completo sistema di menu e di dialog box, queste ultime dotate di radio button e check box (rispettivamente per famiglie di opzioni tra loro alternative e non), nonché di righe di input con associata history list (che consente di ripescare le stringhe precedentemente immesse) o list box (che permette di scegliere tra più stringhe rappresentate in «finestrelle» dotate di scroll bar per una più rapida e comoda consultazione). Il tutto, come già detto, usabile velocemente non solo da tastiera ma anche mediante il mouse.

Risulta anche arricchito e potenziato l'help on-line ipertestuale, in cui vengono accentuate caratteristiche già note, quali il posizionamento automatico sulla schermata relativa alla parola su cui è il cursore (se la parola non è compresa nell'indice dell'help si seleziona quella più simile), o il trasferimento degli esempi in una finestra di editing (esiste un apposito comando nel menu *Edit* che consente di effettuare l'operazione con grande rapidità, passando per il *Clipboard).* Piacciono le possibilità di muovere e soprattutto di ingrandire la finestra di help («zoommabile» come qualsiasi altra), nonché la presenza della documentazione completa anche del Turbo Vision. Va segnalato al riguardo che l'interfaccia utente del Pascal 6.0 è stata realizzata proprio con il Turbo Vision, di cui costituisce quindi un ottimo «demo».

Il menu *Options* è ora più ricco, in quanto permette anche di cambiare i colori dell'interfaccia e di salvare su un apposito file su disco sia le opzioni di configurazione che l'aspetto corrente dell'interfaccia (ad esempio: tre finestre aperte su altrettanti file), colori compresi. Ne segue che non troviamo più il programma TINST.EXE, la cui funzione è quasi integralmente assolta dall'ambiente integrato; fa eccezione la possibilità di modificare i comandi dell'editor, che viene ora assicurata da TEMC.EXE, un compilatore di macro con le quali si può in altro modo adattare l'editor ai propri gusti (TEMC.EXE produce un file di configurazione che viene letto automaticamente da TURBO.EXE).

Rimane invece sostanzialmente immutato il debugger integrato, nonostante una migliore organizzazione dei relativi menu, in quanto manca quella opzione *Inspect,* tradizionale esclusiva del Turbo Debugger di cui avevamo apprezzato la disponibilità nell'ambiente integrato del Turbo C++.

Il paragone con il Turbo C++ viene spontaneo, forse anche troppo. Si riscontra che vi sono notevoli somiglianze tra le interfacce utente dei due prodotti, tanto che stupiscono le differenze: non c'è la *Programmer's Platform,* cioè la possibilità di chiamare altri programmi (tipicamente il Debugger, l'Assembler o utility come GREP o TOUCH) senza uscire dall'ambiente integrato, non ci sono i *Project,* non viene usato il VROOMM. In realtà, se magari una *Platform* non avrebbe guastato, la compilazione separata in Turbo Pascal è molto più semplice che non in C o in C++ e i *Project* sono praticamente superflui; in Turbo C++, al contrario, non è possibile compilare un programma comprendente più di un file di sorgente se non ricorrendo ai *Project.* Non troppo diverso il discorso per quanto riguarda il VROOMM: un C++ è comunque «affamato» di memoria, al punto che i traduttori Comeau, Glockenspiel o Guidelines richiedono *almeno* un mega o un mega e mezzo di RAM; la Borland è riuscita a contenere le esigenze del Turbo C++ in 640K (il C++ della Zortech si accontenta di 512K solo perché non propone un vero e proprio ambiente integrato), ma il VROOMM era praticamente necessario tanto quanto lo era per Quattro Pro. Il Turbo Pascal si può invece permettere il lusso di mantenere in memoria, oltre ad editor, compilatore e debugger, anche tutte le unit comprese nel file TURBO.TPL; se la RAM dovesse scarseggiare, si può evitare il caricamento di TURBO.TPL (come detto sopra), o si può ricorrere al compilatore separato, eventualmente a quello per

memoria estesa se si è acquistata la versione Professional.

In realtà l'ambiente integrato del Turbo Pascal non deriva da quello del Turbo C++, ma entrambi derivano dall'adesione a specifiche che si stanno sempre più imponendo come standard.

Implementazione del linguaggio

Come ho già accennato in apertura, sono poche le innovazioni apportate alla sintassi del linguaggio; non troviamo ad esempio né *operator overloading* né ereditarietà multipla. Si apprezza molto, comunque, la possibilità di scrivere in un programma istruzioni in linguaggio assembly direttamente in forma simbolica, senza dover più ricorrere alla trascrizione della loro codifica in esadecimale; si dispone infatti di una parola riservata **asm,** che segnala al compilatore l'inizio di una sezione scritta in assembly, e di una direttiva **assembler** che, se applicata ad una funzione o procedura, permette alcune ottimizzazioni (anche i parametri valore sono passati per indirizzo, vengono usati i registri del processore o del coprocessore per i risultati delle funzioni, non viene generato l'usuale codice di entrata e di uscita se non vi sono parametri né variabili locali).

Ancora più importante l'introduzione della direttiva **private,** che consente di dichiarare «privati» alcuni campi-dati o metodi di un tipo **object.** Il manuale dedicato all'OOP del Pascal 5.5 non poteva che limitarsi a *raccomandare* di accedere ai campi-dati di un oggetto solo attraverso appositi metodi del tipo *SetX* e *GetX,* ma ora è possibile regolare effettivamente l'accesso mediante la combinazione della protezione assicurata dal meccanismo delle unit e della nuova direttiva: un campo dichiarato **private** in una unità di compilazione, unit o program, può essere letto o scritto liberamente — e un metodo liberamente chiamato — da tutto il codice della stessa unità, ma rimane inaccessibile per il codice di altre (vi rimando alla rubrica Turbo Pascal per una discussione di tale aspetto e relativi esempi).

Troviamo anche una piccola variazione nella sintassi della dichiarazione di un tipo **object:** è ora necessario terminare con un punto e virgola la dichiarazione di ogni campo, compreso l'ultimo; per dirla in altro modo, a differenza di

SAA-CUA, OS/2, Windows e Vision

Parlando delle finestre, il manuale del Turbo Vision propone alcuni consigli supponendo, a titolo di esempio, che si voglia realizzare un editor: raccomanda di porre nelle sole «viste interne» (i «pannelli» di una finestra) la funzionalità dell'editor, lasciando la finestra come tale quanto più possibile così come viene fornita dalle unit precompilate del Turbo Vision: un «gruppo» di «viste», il cui compito è essenzialmente quello di presiedere, dietro le quinte, alla visualizzazione di tali viste. «Le viste possono essere facilmente riutilizzabili se sono state progettate con cura, e spostare l'editor di testi in ambienti diversi potrebbe rivelarsi un'operazione non facile se le sue funzionalità fossero divise fra un gruppo ed una vista». A quali «ambienti diversi» pensava l'estensore del manuale?

Cambiamo per un attimo argomento. Alcuni utenti del Turbo Pascal 5.5 hanno comprato ed usato *Object Professional* 1.0, una vasta e ben fatta libreria di funzioni della TurboPower Software, comprendente tra l'altro anche una ricca gerarchia di classi per la realizzazione di interfacce utente sofisticate. Ad essi si pone ora un bel dilemma: restare fedeli al Pascal 5.5 con *Object Professional,* o passare al Pascal 6.0 con Turbo Vision? C'è anche da dire che, volendo usare l'*Object Professional* con il Pascal 6.0, si dovrebbe chiedere l'upgrade alla TurboPower Software, in quanto la versione 1.0 del loro prodotto non è compatibile con la nuova gestione della memoria dinamica nel Pascal 6.0. A mio parere, la rinuncia al Turbo Vision sarebbe giustificata solo se si fosse già usata la libreria della TurboPower in una gran quantità di codice: l'*Object Professional* infatti, per quanto molto flessibile e potente, è anche molto complesso e richiede lunghi tempi di apprendimento; basti pensare che viene fornita una utility apposita per la preparazione del sistema di menu di un programma, mentre con il Turbo Vision bastano poche semplici istruzioni. Ma c'è dell'altro. Proviamo a dare un'occhiata a cosa troviamo nel Turbo Vision (ho messo su MC-Link un file VISION.ZIP che ne contiene un efficace demo).

a) Viste

Ogni cosa che può apparire sul video discende da *TView,* classe astratta di porzioni rettangolari dello schermo. Da questa deriva anche *TGroup,* classe di tutte le viste che possono avere «sottoviste», comprese anche le classi *TProgram* e *TApplication,* in quanto ogni programma comunica con l'utente attraverso viste. Tra queste abbiamo istanze di *TWindow* e *TDialog,* che possono comprendere sottoviste «terminali» (non derivate cioè da *TGroup)* quali *TFrame, TButton, TCheckBox, TRadioButton, TMenuBar, TMenuBox, TInputLine, TListBox, TScrollBar,* ecc., in generale tutto l'armamentario di una interfaccia utente quale quella del Pascal 6.0. Ma non solo di questo. Nel dicembre del 1987 la IBM pubblicò un volume dal titolo *Systems Application Architecture, Common User Access: Panel Design and User Interaction;* vi erano delineati i principi del disegno di interfacce utente quanto più possibile coerenti da sistema a sistema, dal personal al mainframe. Nel 1989 quel volume è stato aggiornato e sdoppiato: *SAA-CUA: Basic Interface Design Guide,* dedicato ai sistemi con terminali non programmabili, dall'OS/400 in su, e SAA-CUA: Advanced Interface Design Guide, dedicato all'interfaccia grafica di OS/2 e i cui principi sono quasi integralmente rispettati anche da Windows 3.0. Nell'interfaccia grafica CUA troviamo finestre mobili e dimensionabili a piacere, dialog box, menu a barra con associati sottomenu pull-down, scroll bar, check box e radio button, list box e combo box, cioè una combinazione di una input line e di una list box (l'IBM propone — anzi impone — traduzioni in varie lingue di tutti i termini della CUA, ma purtroppo non mi riesce proprio di scrivere «sviluppo azione» invece di «pull-down»...). Tutti elementi che ritroviamo nell'interfaccia del Pascal 6.0 e che possiamo incorporare nei nostri programmi grazie al Turbo Vision. Vi sono certo differenze, dovute non solo al fatto che la Borland — per ora — si limita ad interfacce in modo testo, ma soprattutto al multitasking: in OS/2 (e anche in Windows, sia pure con restrizioni) l'utente può eseguire più applicazioni contemporaneamente, ognuna delle quali agisce dentro una finestra; il Turbo Vision è però fatto per l'MS-DOS, e quindi ogni programma occupa l'intero schermo, dentro una «finestra» che non può corrispondere a quelle di OS/2 o Windows (non se ne possono cambiare le dimensioni, ecc.). Rimane il fatto, tuttavia, che i principali componenti dell'interfaccia SAA-CUA si ritrovano fedelmente emulati in Turbo Vision.

b) Elementi non visibili

I *TStream* consentono l'I/O polimorfico. Il Turbo Pascal non consente di aprire «file di oggetti», e lo stesso C++ non offre alcuno strumento per salvare su disco gli oggetti creati durante l'esecuzione (non mancano librerie che lo consentono, prima tra tutte la NIH di Keith Gorlen, che è però di uso tutt'altro che semplice). Si parla a questo riguardo del problema della «persistenza» degli oggetti (della loro capacità, cioè, di

*La direttiva **assembler** e l'istruzione **asm** consentono di inserire in un programma istruzioni in linguaggio assembly direttamente in forma simbolica.*

```
procedure Cap(var Strg: string); assembler;
(* Converte in maiuscolo il parametro Strg *)
begin
  asm
      LES   DI, Strg
      MOV   CL, ES:[DI]
      INC   CL
  @1: DEC   CL
      JZ    @2
      INC   DI
      CMP   ES:BYTE PTR [DI], 'a'
      JB    @1
      CMP   ES:BYTE PTR [DI], 'z'
      JA    @1
      SUB   ES:BYTE PTR [DI], 20H
      JMP   @1
  @2:
end;
```

quanto avveniva nella versione 5.5 l'**end** finale deve essere sempre preceduto da un punto e virgola. Curiosa innovazione che sarebbe stato bene mettere in buona evidenza, in quanto può capitare di non riuscire a capire immediatamente i motivi per cui non si riesce a ricompilare sorgenti scritti per la versione precedente.

Vi è comunque una ben più importante fonte di possibili incompatibilità, questa volta ben documentata. Le unit compilate con il Pascal 5.5 vanno ricompilate e questo, punti e virgola a parte, in genere non comporta alcun problema. A meno che non si tratti di unit in cui si usano le variabili *FreeMin* o *FreePtr*, quelle che consentivano il controllo dell'heap. La gestione della memoria dinamica è stata infatti cambiata. In passato i blocchi allocati e poi rilasciati venivano inseriti in una lista *(FreeList)* implementata come un array di 8191 coppie di puntatori (uno per l'inizio e l'altro per la fine di ogni blocco): ne potevano derivare problemi nel caso di deallocazione di blocchi in situazioni di lista piena; eventualità altamente improbabile, ma pur sempre possibile, soprattutto a causa di una «granularità» dell'heap ri-

«persistere» tra una esecuzione ed un'altra di un programma), che il Turbo Vision risolve appunto mediante i *TStream*. Le *TCollection* consentono la creazione e gestione di insiemi di oggetti più o meno come un array o una lista, ma gli oggetti possono essere di tipo diverso. Vi sono altre classi derivate da queste, e tutte supportano la programmazione orientata all'oggetto non solo nel disegno della interfaccia, ma anche di quello che vi agisce dietro. Vanno tra queste segnalate le classi *TResourceFile* e *TResourceCollection*, che permettono la creazione di file di «risorse», cioè di oggetti usati da un programma che, invece di essere creati e inizializzati, possono essere tratti da un file che può essere modificato senza necessità di intervenire sul programma che lo usa. È questo il metodo più rapido per creare programmi con sistemi di menu alternativi, con messaggi in diverse lingue, con diversi colori, ecc. Ed è un metodo che ritroviamo, anche se realizzato in modo un po' diverso, nella programmazione sotto Windows o OS/2.

c) Programmazione per eventi

La SAA distingue tra programmazione tradizionale o «procedurale» e programmazione *event-driven*, praticamente necessaria per sistemi a finestre e quindi per OS/2 e Windows. Più che di una necessità, tuttavia, si tratta di un approccio molto efficace allo sviluppo di applicazioni, basato sulla netta separazione tra la ricezione dell'input dell'utente e la risposta a tale input, quindi tra il disegno dell'interfaccia e il funzionamento del programma. A differenza che nell'*Object Professional*, nel Turbo Vision ritroviamo anche una piena adesione alla programmazione per eventi; possiamo ad esempio creare una riga di stato ridefinendo il metodo *InitStatusLine* di *TApplication:*

```
procedure TMyApp.InitStatusLine;
var R: TRect;
begin
  GetExtent(R);
  R.A.Y := R.B.Y - 1;
  StatusLine := New(PStatusLine, Init(R,
    NewStatusDef(0, $FFFF,
      NewStatusKey('', kbF10, cmMenu,
      NewStatusKey('~Alt-X~ Exit', kbAltX, cmQuit,
      NewStatusKey('~F4~ New', kbF4, cmNewWin,
      NewStatusKey('~Alt-F3~ Close', kbAltF3, cmClose,
      nil))))),
    nil)
  ));
end;
```

Possiamo implementare le azioni da eseguire in risposta ai comandi ridefinendo il metodo HandleEvent (alle costanti cmQuit e cmClose pensa già il metodo di TApplication, che quindi è sufficiente chiamare):

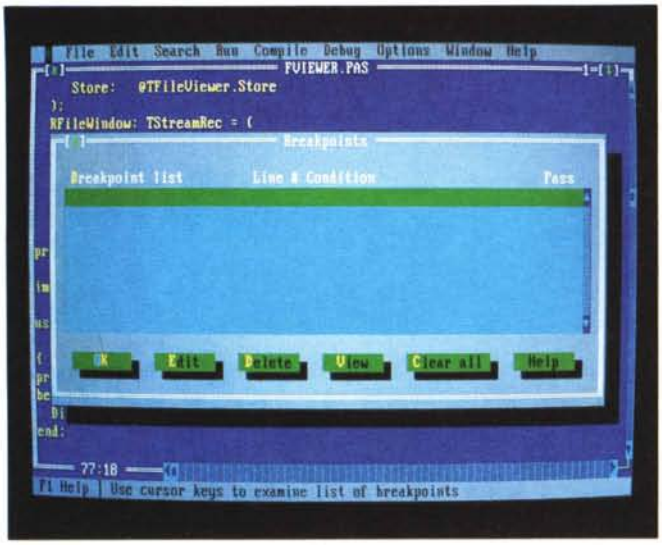
```
procedure TMyApp.HandleEvent(var Event: TEvent);
begin
  TApplication.HandleEvent(Event);
  if Event.What = evCommand then
  begin
    case Event.Command of
      cmNewWin: NewWindow;
    else
      Exit;
    end;
    ClearEvent(Event);
  end;
end;
```

Non è possibile entrare ora nei dettagli della programmazione per eventi, ma basta notare che, una volta definiti il metodo *NewWindow* e il case che lo chiama, possiamo togliere quell'«F4» dalla riga di stato, possiamo definire un menu che contenga l'opzione «Nuova Finestra», possiamo pure decidere che invece si apra una nuova finestra quando si preme il bottone destro del mouse mentre si è sulla scroll bar di una finestra già aperta, ma tutto ciò, in quanto verrebbe tradotto in un «evento», non comporterebbe alcuna modifica né di *HandleEvent* né di *NewWindow*.

d) Comunicazioni tra viste

In Windows e in Presentation Manager le diverse applicazioni possono comunicare tra di loro anche mediante il *Clipboard*, una sorta di buffer condiviso da tutte. Analogamente, il Turbo Vision incoraggia (e il compilatore integrato implementa) lo stesso tipo di meccanismo per la comunicazione di dati tra le diverse «viste» di una stessa applicazione.

Ne possiamo concludere che il Turbo Vision offre una ricca gerarchia di classi per il disegno di interfacce utente flessibili e sofisticate, permette di tenere indipendenti il funzionamento dell'applicazione e la sua interfaccia, il tutto in modo reso ancora più semplice ed efficace a un tempo grazie alla programmazione per oggetti; ma va un po' oltre: aderendo in buona misura alle specifiche SAA-CUA, consente di realizzare applicazioni di aspetto coerente con uno stile che si sta sempre più affermando come standard, e, quello che forse più conta, potenzialmente trasportabili senza troppo sforzo sotto altri ambienti. Qualcuno a questo punto potrebbe osservare, ad esempio, che non si programma ancora in Pascal sotto Windows 3 (l'SDK di Windows 3 non supporta più il Pascal Microsoft). È vero, ma recentemente la Borland ha annunciato in America un Turbo Pascal per Windows, che potrebbe rendere possibile portare almeno sotto tale ambiente i programmi realizzati con il Turbo Vision per l'MS-DOS. Che fosse questo il «diverso ambiente» cui pensava l'estensore del manuale?



La dialog box dedicata ai breakpoint del debugger integrato.

dotta ad un solo byte; ciò comportava che veniva sempre allocato il numero effettivo di byte richiesti, con la conseguenza che ad esempio, allocati e poi rilasciati 50 byte, quel blocco poteva essere riutilizzato per una successiva richiesta di 49, creando così un blocco libero di un solo byte (e potenzialmente tanti blocchi «piccoli») difficilmente riutilizzabile. Poteva anche accadere che la lista, pur non piena, non riuscisse ad accogliere nuovi blocchi in quanto tra questi non vi erano i blocchi agli indirizzi più alti dello heap, quelli cioè verso i quali cresceva la lista (che cresceva verso il basso come uno stack), al punto che era stato predisposto un «cuscinetto» tra l'indirizzo più alto dell'heap *(HeapPtr)* e quello più basso della lista *(FreePtr)*, la cui dimensione era conservata nella variabile *FreeMin*. Ora la *FreeList* viene implementata come una lista vera e propria, con blocchi di almeno otto byte contenenti nei primi quattro l'indirizzo del blocco successivo e negli altri la dimensione, e la granularità del l'heap è stata elevata a otto byte. Da qui l'incompatibilità con codice che intervenisse su *FreePtr* (i riferimenti a *FreeMin* possono essere semplicemente eliminati), come l'*Object Professional* 1.0 della TurboPower Software; ma tutti quei problemi sono spariti. Credo che l'attuale migliore implementazione della gestione della memoria dinamica giustifichi ampiamente l'onere di eventuali adattamenti.

Conclusione

Ci sarebbe naturalmente da descrivere almeno sommariamente la seconda principale novità del Pascal 6.0, il Turbo Vision; magari in un paragrafo intitolato «Librerie». Così facendo, tuttavia, mi sembrerebbe di ridurlo appunto a mera libreria, mi sembrerebbe di cadere nell'errore di considerare il Turbo Vision come l'ennesima libreria di funzioni per la realizzazione di interfacce utente a finestre, per di più in un ormai quasi antiquato «modo testo». Ho preferito quindi dedicare un riquadro ad una breve discussione del rapporto tra il Turbo Vision e l'interfaccia utente «standard», quella descritta dalla IBM nelle sue pubblicazioni relative al SAA-CUA ed implementata nel Presentation Manager di OS/2 e in Windows. Sono infatti convinto che solo in questo modo si può comprendere correttamente la collocazione del nuovo prodotto della Borland.

Esaminando il compilatore abbiamo riscontrato in primo luogo che la Borland ha ormai aderito a quello standard, sia dotando l'ambiente integrato di un'interfaccia utente con esso coerente, sia mettendo a disposizione di tutti la gerachia di classi con cui quella interfaccia è stata realizzata. Abbiamo riscontrato anche una qualche minore «potenza» rispetto al Turbo C++, sia nell'ambiente integrato che nelle caratteristiche object-oriented del linguaggio. A chi servono allora il Turbo Pascal 6.0 e il Turbo Vision?

Per rispondere basta considerare che, chi non abbia già una qualche dimestichezza con la OOP, deve mettere in conto un arduo cammino se vuol cominciare dal C++; molto potente ma anche molto complesso, il C++ è tra l'altro ancora privo di una soluzione più o meno standard al fondamentale problema della «persistenza» degli oggetti. Chi voglia apprendere la programmazione per eventi in una interfaccia utente a finestre, si trova forse anche peggio se vuol cominciare dal Software Development Kit che la Microsoft propone per Windows 3: strumento tanto affascinante quanto ostico. Persino la IBM riconosce poi le notevoli difficoltà della programmazione sotto Presentation Manager. Chi volesse infine il meglio dei tre mondi, potrebbe trovare in Smalltalk un efficace «incapsulamento» delle asperità del trattamento di finestre ed eventi in un coerente e potente ambiente di sviluppo integralmente orientato all'oggetto, ma è universale il giudizio che i tempi di apprendimento non sono brevi. Eppure non si può ignorare che finestre, eventi ed oggetti rappresentano l'immediato futuro della programmazione. Ecco quindi la risposta: Pascal 6.0 e Turbo Vision consentono un approccio indolore a tutti gli ingredienti di questo futuro, non richiedendo né lunghi tempi di apprendimento né ambienti bisognosi di processori muscolosi e di RAM in quantità. Si propongono quindi a chi voglia muovere senza affanno i primi passi nel mondo di domani, nella migliore tradizione del Pascal, linguaggio «didattico» per eccellenza. Ma non solo: ho già notizia di qualche software house che li ha scelti per realizzare fin da ora programmi dotati di una interfaccia utente moderna sotto MS-DOS, e facilmente portabili sotto Windows non appena sarà disponibile il Turbo Pascal per quell'ambiente. Si propongono infatti anche a chi voglia accorciare i tempi di sviluppo di software professionale il cui ciclo di vita non venga compromesso dalla prevedibile evoluzione del mercato.

A che prezzo tutto ciò? Sarebbe facile dire «agli stessi prezzi del Turbo Pascal 5.5». In realtà va considerato che si dispone ora di una possibilità di upgrade «ad ampio spettro»: può accedere convenientemente alla versione Professional chiunque abbia già un qualsiasi linguaggio Borland. Di questo deve tenere conto in particolare chi credesse che, volendo acquistare le versioni Professional sia del Turbo Pascal che del Turbo C++ (magari perché il Turbo Drive si trova solo nel 6.0 Pro), si troverebbe a dover pagare due volte Debugger, Assembler e Profiler. Ne ho discusso con la Borland e, esaminate le diverse combinazioni possibili, ne abbiamo concluso che situazioni del genere molto difficilmente potrebbero verificarsi; hanno comunque assicurato ampia disponibilità nei confronti di chi li contattasse per problemi di tale natura.

MC

the new IMPACT

SERIES II ™ A500-HD+

a futura generazione nelle periferiche Amiga 500
n nuovi standards di bellezza e qualità

MPACT
eries II

IMPACT SERIES II trasforma il tuo A500 in un computer vero e più divertente!

Il nuovo SERIES II A500-HD della GVP é il massimo per quanto riguarda Hard Disk, memoria ed espandibilità per il vostro Amiga 500. Le più importanti caratteristiche sono:

Bordo di attacco
stesse caratteristiche di FAAASTROOM ™ e VLSI su misura ad alta tecnologia dei prodotti del nuovo SERIES II A2000 SCSI-RAM della GVP.

Possibilità future
Soltanto il nuovo "Mini-Slot" evidenzia tutti i segnali della barra di espansione A500, tenendo conto delle future interessanti possibilità di espansione - la sola alternativa intelligente al rischioso sistema "Pass-Through".

Affidabilità
Una ventola interna garantisce il raffreddamento dell'alimentazione ed evita che si verifichi un sovraccarico nell'alimentazione del vostro A500. GVP non verrà meno per quanto concerne affidabilità e qualità !

Espansione di memoria
Espandione RAM interna fino a 8MB usando i moduli di memoria SIMM, facili da installare.

Design
Il design personalizzato, eseguito mediante stampaggio ad iniezione, si accorda perfettamente con il vostro A500 per bellezza ed eleganza ineguagliabili, stabilendo un nuovo standard per le unità periferiche A500.

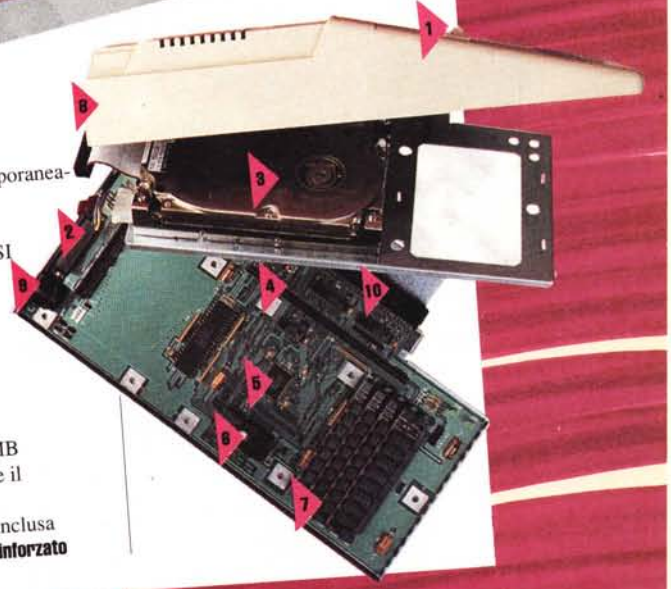
Livello tecnico-scientifico
Nuovo Hard-Disk Drive interno da 1"; disponibile da 40MB fino a 100MB.

Prestazioni
Fornisce prestazioni di Hard-Disk senza compromessi che finora non sono ancora state viste in A500.

Provare per credere
Prendetene uno dal vostro più vicino fornitore GVP : non crederete ai vostri occhi

Date uno sguardo all'interno

1. **Interruttore gioco**: abilita la Ram abilitando contemporaneamente la compatibilità gioco
2. **Entrata esterna** SCSI: permette di attaccare fino a 7 periferiche SCSI
3. **Hard Disk Drive di serie da 1"- 20MB fino a 100MB**
4. **"Mini-Slot"** per future possibilità di espansioni
5. **Chip VLSI specifico GVP**
6. **Driver SCSI FAAASTROM della GVP**
7. **Espansione Ram interna**: fino a 8MB
8. **Ventola interna**: raffredda durante il funzionamento
9. **Entrata alimentazione universale**: inclusa
10. **Connettore bordo scheda 86-PIN rinforzato**



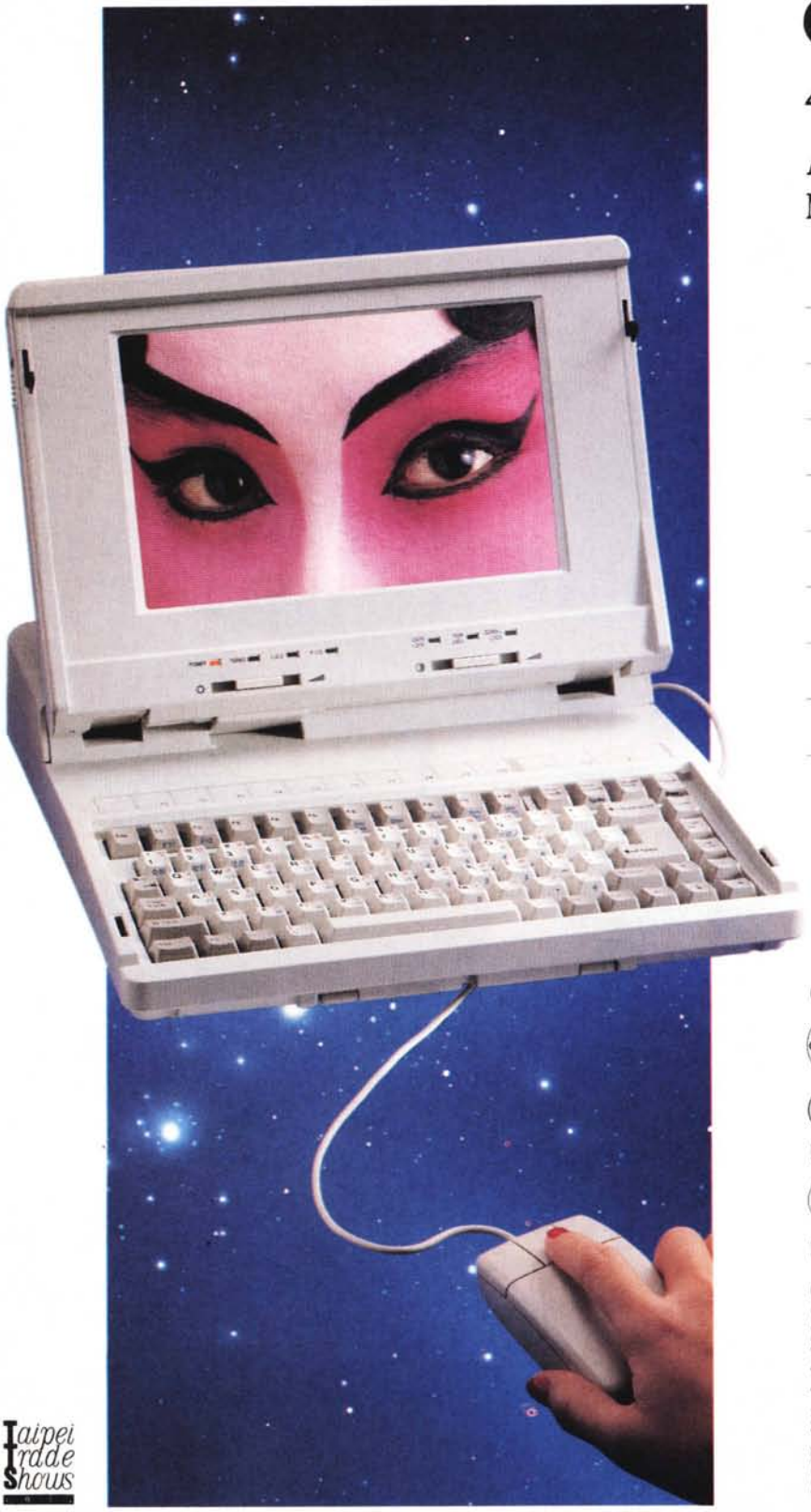
GVP

NON STOP ELECTRONICS DIVISION SPA
VIA BUOZZI, 11 - 40057 CADRIANO (BO)
TEL. 051/765299 - FAX. 051/765252

NON STOP
electronics division

台 北

TAIPEI



COMPUTEX'91

4-10 Giugno 1991

Accesso Istantaneo al Mondo dei Computer

In Programma

- Computer
- Periferici
- Software
- Sistemi di Automazione per Uffici
- Comunicazione Dati
- Applicazioni
- Memorie di Massa
- Componenti

Organizzatori:

CHINA EXTERNAL TRADE DEVELOPMENT COUNCIL

TAIPEI COMPUTER ASSOCIATION

Patrocinatore

TAIPEI WORLD TRADE CENTER

Locazione: TWTC EXHIBITION HALL
CETRA EXHIBITION HALL

Contatto: TWTC EXHIBITION HALL
5 Hsinyi Road, Section 5, Taipei, Taiwan
Republic of China
Tel: (02)725-1111 Fax: 886-2-725-1314
Telex: 28094 TPEWTC
TAIPEI COMPUTER ASSOCIATION
3Fl., No. 2 Pa Teh Rd., Sec. 3, Taipei, Taiwan
Tel: (02)7764249 Fax: (02)7764410

Rappresentante:
Milano-Centro Commerciale Per L'estremo Oriente
Tel: (02)29404196, 29403319 Fax: 39-2-2047077

Taipei Trade Shows