Microsoft Basic 7.0

di Raffaello De Masi

Il Basic, fin dalla sua comparsa, fu sinonimo di prestazione di terz'ordine, lentezza, groviglio senza fine con spaghetti-program inestricabili. Vero o falso? Non tocca a me dirlo: o forse tocca a me dire qualche parola a proposito dell'atteggiamento che ho avuto nei confronti di questo sondaggio.

Chi ha letto altre prove, su questo idioma, da me eseguite sa quanto apprezzi questo linguaggio e come abbia sempre cercato di dimostrare come i suoi detrattori avessero torto. Nel caso della prova comparata del Turbo-Quick Basic evidenziai come programmazione strutturata e chiarezza di codice erano possibile e molto più semplici con i Basic oggi a disposizione, col vantaggio dell'interattività che altri non hanno; cercai di far vedere che chiarezza e linearità del codice sono frutto della «pulizia» del programmatore e non prerogativa di questo o quell'idioma.

Oggi, a costo di scatenare procelle sul mio capo mi sento di affermare che con questo Basic in mano un programmatore esperto può dare del filo da torcere a tutti i linguaggi di questo mondo, il tutto con la facilità proverbiale del Basic. Non ci credete? Se chi mi legge non ha preconcetti verso questo idioma, non ha che da seguirmi in questa prova o magari passare da un rivenditore per un esame sul campo; si ritroverà a maneggiare un attrezzo capace di fare cose apparentemente impensabili e non solo per un Basic. Come si dice: «provare per credere»!

Il pacchetto

Altro che pacchetto, il pacco è il caso di dire! Tutto il materiale del Basic 7 è racchiuso in una grossa scatola che contiene due ciclopici manuali legati in raccoglitori ad anelli e da un volume rilegato; completa il tutto un manualetto di «getting started» e i dischetti del software, ben dodici, raccolti in due buste chiuse con la solita clausola «Se apri accetti...».

Ma cosa è o meglio che cosa rappresenta; alla Ferrini possiamo dire che la stessa definizione lo dice: il nome completo del pacchetto è infatti «Microsoft® Basic 7.0 Professional Development System»; il suo compito-fine è quello di permettere di creare e perfezionare am-

pi e sofisticati programmi capaci di girare sotto MS-DOS e OS/2. Il pacchetto è composto dei seguenti moduli:
- il completo ambiente di sviluppo Microsoft QuickBasic Extended (QBX)
- il compilatore (BC)
- una serie di utility e tool come:
— il Microsoft Segmented-Excecutable linker (LINK)
— il Microsoft Library Manager (LIB)
— il Microsoft Program-Maintenance Utility (NMAKE)
— il Runtime Module Build Utility (BUILDRTM)
— il Microsoft HelpFile Creation Utility (HELPMAKE).
- Una serie di librerie precostituite come il Data-Time, il Financial e il Format.
- Una serie di ToolBox precostituiti, come MatrixMath, Presentation Graphics e User Interface.
- Il Microsoft CodeView debugger e una serie di tool di sviluppo rappresentati da routine in diversi linguaggi.
- Una documentazione che, prendendo spunto dal supporto stampato, fornisce una documentazione online completa.

Parliamo per un momento della documentazione cartacea; la suddivisione in quattro blocchi non è peregrina; si parte dal Getting Started, che mette l'utente iniziale in condizione di lanciare il linguaggio e di entrare in ambiente. Il manuale prevede che l'utente abbia almeno alla lontana utilizzato anche solo sporadicamente il QuickBasic; a tal uopo il primo paragrafo di questo manualetto è dedicato a mostrare come e in che modo questo nuovo linguaggio sia migliorato rispetto alle precedenti versioni. Il secondo paragrafo (che per la verità dovrebbe essere il primo ad essere letto prima di eseguire l'installazione, consente di eseguire il «setup», l'installazione su disco rigido, con tutte le clausole destinate ad ottimizzare la grandezza e la velocità del programma. Il capitolo 3, più specialistico mostra come ottenere il meglio del sistema attraverso l'uso di memoria estesa o espansa. L'ultimo fornisce una breve introduzione al nuovo sistema di Help on-line.

Il primo manuale vero e proprio è la Programmer's Guide, vero e proprio tutorial del linguaggio (anche se, giustamente, certa informazione di base è saltata a pié pari). La prima parte mette a fuoco le caratteristiche del Basic, incluso alcune caratteristiche specialistiche come strutture di controllo di flusso, manipolazione di numeri e stringhe, gestione e recupero delle situazioni d'errore, ecc. La seconda parte dello stesso manuale è invece dedicata alle utility esterne di sviluppo fornite assieme al linguaggio, con una accurata disamina della sintassi e delle opzioni di comando per ogni utility, con una serie di rapidi ma efficaci esempi d'uso delle stesse. Una serie di appendici, in fondo al manuale, richiamano le definizioni di base del linguaggio Basic.

L'altro grosso manuale è il Basic Language Reference, contenente a sua volta tre parti principali; la prima, una vera e propria guida di riferimento del linguaggio, descrive le funzioni del Microsoft Basic, le istruzioni, i comandi e i metacomandi. La seconda «Add-On Library Reference» descrive le funzioni comprese nelle librerie aggiunte, la terza «Basic Toolbox Reference» infine le procedure contenute nei toolbox.

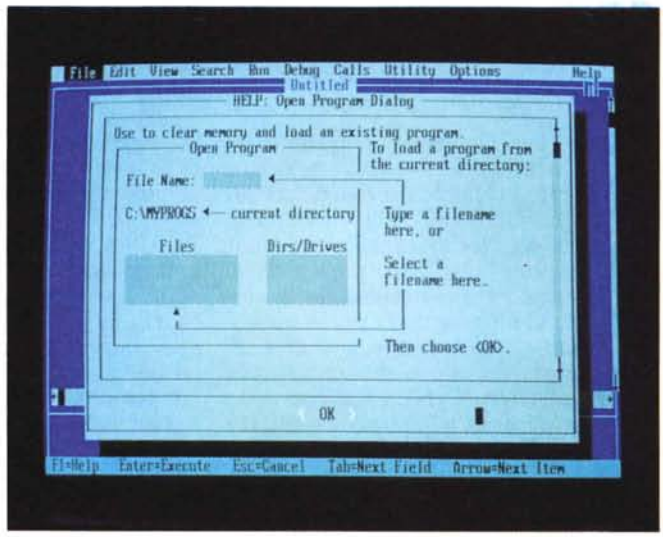
Oltre a questa documentazione cartacea, come già abbiamo accennato prima, MS Basic 7 mette a disposizione del programmatore una completa base di dati contenente la maggior parte delle istruzioni presenti nei manuali; il tutto è gestito attraverso il Microsoft Advisor on-line Help System; si tratta di un Help cosiddetto context-sensitive che spazia

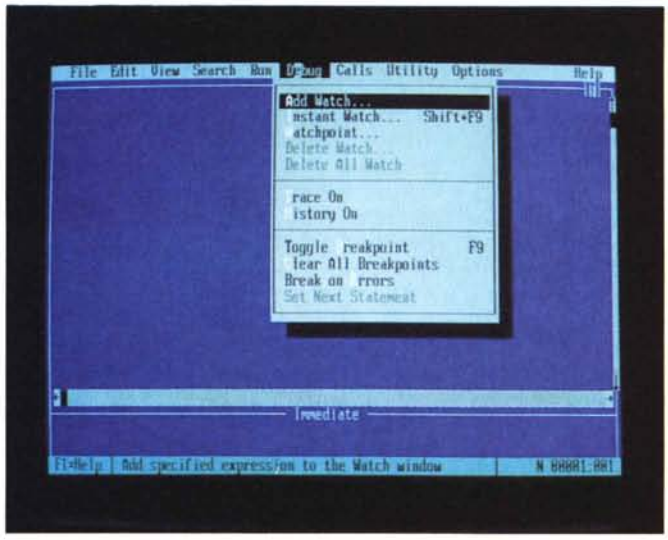
Microsoft® Basic 7.0

for IBM Personal Computer and Compatibles
Microsoft Corporation

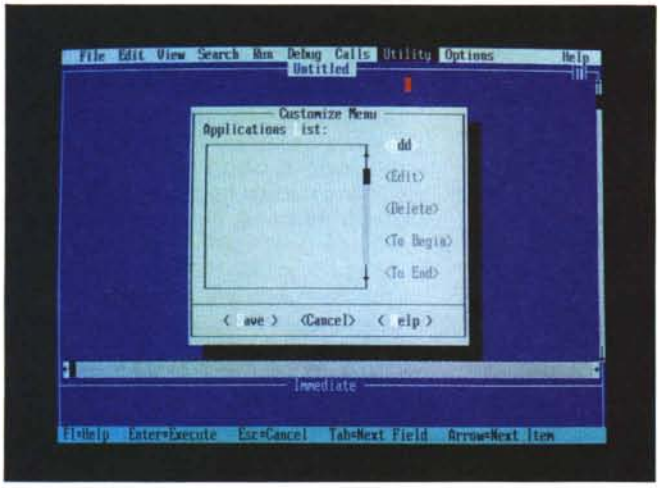
Costruttore:
Compaq, USA.
Distributore:
Microsoft Italiana
Via Cassanese, 224 Pal. Tiepolo
20090 Segrate (MI)
Prezzo ***(IVA esclusa):*** ***L. 850.000***

L'Helpcontext sensitive, in linea, con i path grafici esplicanti l'azione in corso.





Il menu di debug, con le possibilità di inserire breakpoint e trap di «on error».



La window principale di customizzazione del menu.

dalle caratteristiche generali del linguaggio Basic al QBX alle opzioni del compilatore, ai messaggi d'errore, alla notazione simbolica possibile nel linguaggio.

Le caratteristiche specifiche del linguaggio

Per la verità, parlare di linguaggio è davvero molto restrittivo. In questo caso di fronte a un vero e proprio tool di sviluppo che, prendendo le mosse da un ambiente QuickBasic like si sviluppa verso caratteristiche, capacità, efficienza e prestazioni mai così riunite in un linguaggio di programmazione. Si tratta, per questo, di un package che offre il meglio solo in mano a programmatori professionisti, ma che mantiene, comunque, la facilità e la disponibilità d'uso di idiomi meno avanzati, come Turbo, Quick o Better Basic, tanto per citare i più recenti presenti sul mercato.

Nonostante questi straordinari miglioramenti Basic 7 non è caduto nella trappola della specializzazione; esso rimane un linguaggio di uso molto generale e passa senza problemi (come lo evidenziano, tra l'altro, anche i numerosi esempi forniti) dall'area business a quella scientifica. Le migliorie presenti sono elencabili in una serie di categorie così individuabili:

— subpackage ISAM. Si tratta del nuovo Indexed Sequential Access Method (da cui l'acronimo), che lavora solo in DOS (non in OS/2), ed è in breve (ne parliamo altrove) un veloce e semplice metodo per la ricerca di record in ampi e complessi database, con operazioni complesse e congiunte di selezione, ordinamento, manipolazione e trasferimento di informazioni.

Il package contiene una serie di utility molto interessanti, come ISAMCVT.EXE, che consente di leggere e convertire file dBase in formato ISAM, ISAMREPR.EXE che ripara in maniera molto rapida ed efficace i file ISAM guasti, ISAMPACK.EXE che compatta i file fino al 30%, ISAMIO.EXE che converte file ASCII in ISAM e viceversa. Parleremo di tutto ciò fra poco.

— Operazioni specifiche del DOS: il linguaggio contiene una serie di comandi DOS-like che consentono di acquisire operatività proprie del sistema operativo senza uscire dall'ambiente; esempi sono DIR$, CURDIR$, CHDRIVE, dalle funzioni analoghe a quelle dei corrispondenti comandi di sistema operativo.

— Opzione CDT (Currency Data Type) che permette di conservare la precisione della notazione in virgola fluttuante, con il vantaggio della velocità della notazione intera. Il vantaggio, derivante dalla rappresentazione interna come interi anche delle cifre decimali si apprezza soprattuto nelle operazioni di addizione e sottrazione.

— Completa e articolata gestione strutturata delle condizioni di errore; si tratta forse del miglioramento più marcato e avanzato. Rispetto alle precedenti versioni, è stato completamente modificato il principio di manipolazione dell'evento; nelle precedenti versioni, infatti le routine di error-handling erano gestite a livello di modulo; quando una handle di errore era attivata, esso restava attivo per tutte le procedure all'interno del modulo.

Con Microsoft Basic è invece possibile la gestione dell'errore a livello di modulo o a livello di procedura. Lo stesso evento può invocare diverse routine di manipolazione d'errore, a seconda di quale procedura è in corso.

Ad esempio, è possibile invocare diverse routine di manipolazione dell'errore per il codice ERR 54 [Bad file name], in quanto questo evento ha differenti significati a seconda delle diverse operazioni su file.

— Nelle precedenti versioni del linguaggio, era possibile creare strutture di dati contenenti tipi di dati numerici e di stringa statici usando l'istruzione TYPE ... END TYPE. In questa versione le cose migliorano in quanto è possibile adottare array statiche in addizione ai tipi già nominati, cosa che consente di costruire strutture di dati più flessibili.

— Nelle versioni precedenti (e, a quanto mi risulta, in tutti i Basic tuttora circolanti) in caso di errore di timeout di una periferica non era possibile sapere quale periferica lo aveva generato o da quale linea proveniva l'errore. Con questa nuova versione informazioni più accurate circa questo tipo d'errore possono essere ottenute attraverso le funzioni ERDEV$ e ERDEV. La prima indica la porta su cui si è verificato l'errore, la seconda indica il tipo di errore occorso, che può essere più particolarizzato attraverso le opzioni di analisi dell'handshaking e del timing (CDx, CSx, DSx, OPx) tutte controllabili nel più generale uso della più nota istruzione OPEN COM.

— Nuove librerie; tra queste offrono particolari vantaggi quelle di tempo-data, quelle finanziarie, e quelle di formato. Si tratta di utility che mettono a disposizione dell'utente tool già disponibili su pacchetti specializzati e su spreadsheet, come la trasformazione di valori di data in numerici, il calcolo di tutta una serie di valori finanziari, come deprezzamento valore presente e futuro, calcolo degli interessi, ecc. C'è da precisare, però, che queste funzioni hanno un range differente da quello accettato da MS Excel.

— Nuovo toolbox dedicato al calcolo matriciale; esso contiene una serie di codici sorgenti, già immediatamente utilizzabili, per diverse operazioni matematiche su matrici. Tra queste ricordiamo le quattro operazioni, l'inversione, il calcolo del determinate, e, udite udite, la soluzione delle equazioni simultanee usando la regola di Gauss.

— Infine, ma non ultimo, il linguaggio dispone di un completo toolbox di presentazione, capace di trasformare in chart e grafici statistici i dati ricavati dai programmi; è possibile produrre grafici a torta, a barra, a linea e accatastati, oltre a combinare tra loro gli stessi. Lo stesso toolbox include, inoltre, un set di font grafiche di particolare accuratezza.

— Possibilità di adozione di una interfaccia utente personalizzata; si tratta di un notevole miglioramento di tutto quanto già visto sul mercato; oggi è possibile costruire applicazioni dotate di

menu, box di dialogo, finestre multiple, gestite magari dal mouse.

Le migliorie alle caratteristiche già esistenti
Miglioramenti delle prestazioni

Oltre a queste novità, gli implementatori di questo eccezionale linguaggio hanno rivisto praticamente tutto il linguaggio; ben poche cose sono state lasciate prive di migliorie; ne citiamo solo alcune, tra le più utili.

Nelle precedenti versioni del linguaggio (e non solo nel caso del Basic) esisteva una barriera insormontabile che era quella dell'area di stoccaggio delle variabili. Tutte le stringhe a lunghezza variabile erano conservate nella cosiddetta «near memory» [memoria prossima], una relativamente piccola porzione di memoria (64 kb) che doveva anche conservare le altre variabili semplici (intere, in virgola fluttuante, e in stringa fissa). Anche in caso di moderato uso delle variabili e di accurato lavorio di garbage collection era facile superare, in programmi appena appena più complessi, questi limiti.

Questa versione del Basic supporta le cosiddette «far string», le stringhe lontane, esiliate; con questo sistema le stringhe a lunghezza variabile sono depositate fuori dalla memoria prossima, in un'area di memoria che, pittorescamente viene definita memoria remota; la cosa è davvero raffinata se si considera che questa è organizzata e gestita in maniera abbastanza simile a quella prossima; come esiste un modulo principale di 64K, è possibile costruire automaticamente diversi addizionali blocchi da 64K, a seconda dell'effettiva necessità che, mano a mano si verifica nel programma.

Altra raffinatezza, anch'essa già presente ma qui resa ancora più sofisticata, è la possibilità di creare programmi in overlay; con questa tecnica è possibile costruire codici di lunghezza ben superiore a quella effettivamente residente in memoria (ad esempio, anche con i più miseri 640K è, in teoria, possibile maneggiare programmi lunghi fino a 16 mega. Con una versione di programma di tipo overlay, parti di codice sono residenti fuori dal programma principale, e sono caricate solo se ce n'è bisogno. Questa possibilità, unita alla migliorata «granularità» (possibilità di spezzettare il codice in parti individualmente accessibili) delle routine di runtime consente di scrivere codici e applicazioni davvero piccole e agili, evitando di caricare routine e librerie inutili. Come se non bastasse il compilatore-linker è dotato di un sistema di gestione ottimizzata del processo, che permette di costruire codici finali di minuscole dimensioni. Inoltre, utilizzando una apposita tavola software di settaggio delle opzioni di compilazione è possibile ottimizzare il codice per macchine dotate di processore 80286 (e successivi), che trarrà vantaggio dalle più avanzate istruzioni proprie di queste macchine. Inoltre un pacchetto matematico integrato, destinato a chi non possiede un coprocessore matematico, può portare a notevoli miglioramenti della velocità complessiva del codice (fino al 40% in caso di calcoli ricorsivi del tipo spreadsheet). Accanto a questo package esiste un più convenzionale emulatore di coprocesso-

La possibilità di lanciare, da ambiente, comandi DOS; una possibilità che aumenta di molto la potenza del package.

Che cosa è ISAM

Microsoft Basic 7 mette a disposizione la potenza e la flessibilità del Metodo di Accesso Sequenziale Indicizzato (ISAM) attraverso una serie di comandi diretti, inseriti già nel linguaggio di base; come già detto nel testo, attraverso ISAM si ha a disposizione un metodo semplice ed efficiente per accedere a specifici record in ampi e complessi database.

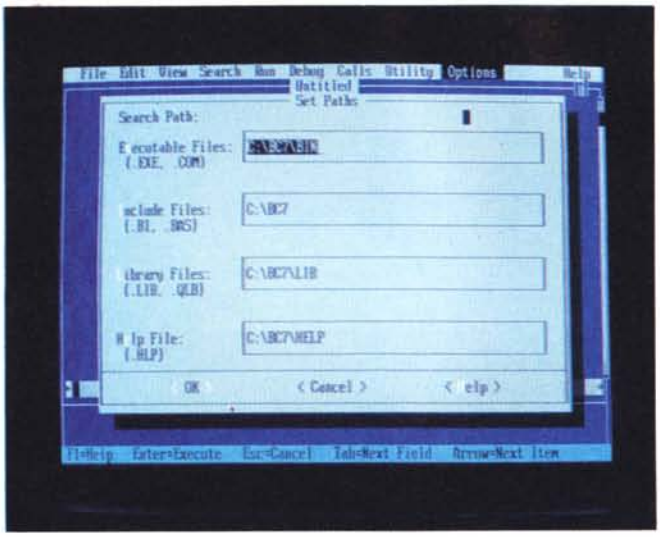
Quando un programma usa o modifica record inseriti in un file, molto spesso è chiamato a raggruppare, riordinare, revisionare le registrazioni in vario modo. Quando il file è complesso e articolato è ben nota la difficoltà di costruire un codice efficiente per la manipolazione di tali esigenze. Molto spesso è richiesto un codice di programma piuttosto lungo, con relativi tempi di redazione estesi e sovente estremamente noiosi.

ISAM rappresenta un approccio differente al problema in quanto consente la creazione e l'aggiornamento in uno speciale data file nel quale i record non abbisognano di un continuo riordino per mantenere, mi si perdoni l'assurdo, una struttura ordinata.

Oltre a ciò, un file ISAM contiene informazioni che descrivono e facilitano l'accesso a ogni record. Molte di queste informazioni sono contenute in tabelle e «indici» (da cui il nome); le tabelle servono a diversi usi, tra cui l'accesso a ognuno dei valori di uno specifico record di dati, gli indici invece sono le vie principali per la ricerca e l'ordinamento dei record presenti nelle tavole.

Le funzioni e le istruzioni del pacchetto ISAM manipolano, organizzano e modificano dati presenti in file particolari, i file ISAM appunto. Gli algoritmi tipici di questo ambiente consentono tutte le operazioni di sort e di ricerca attraverso algoritmi estremamente efficienti, per cui l'uso di questo pacchetto si risolve alla fine in un risparmio di tempo nella fase di programmazione e in una maggiore efficienza del programma. Per applicazioni di tipo database (l'ambiente di utilizzo ideale) i file ISAM sono più convenienti ed efficienti dei file ad accesso causale in quanto è possibile accedere ai record stessi anche se essi sono ordinati in maniera differente.

La sezione dedicata all'ISAM, nella «Programmer's Guide» è molto ben strutturata e particolareggiata; costruire un programma database-like, o leggere uno degli esempi presenti è cosa facile e agevole; un esempio, breve, di programma, convenzionale e ISAM-type mostra che il codice prodotto è nel primo caso lungo quasi il triplo rispetto all'altro.



La possibilità di definire path di ricerca, inserire librerie e file di help; una volta assemblato il pacchetto da caricare, esso può essere sempre rieditato per le opportune correzioni o modifiche.

re in matematica IEEE, che addirittura migliora le prestazioni di macchine già in possesso di 8087.

Miglioramenti dell'ambiente

L'ambiente specifico di programmazione, il QBX, offre caratteristiche altamente professionali, come il supposto di memoria espansa. Con tale sistema QBX supera in maniera del tutto trasparente la barriera dei 640K, usando la memoria espansa, se presente, per qualsiasi parte del programma più piccola di 16K di grandezza. Una serie di comandi specifici, raggruppati nelle subroutine di View, permette di determinare quante e quali parti di programma (codici di modulo o procedure) vanno a finire nella memoria espansa.

Lasciando per adesso da parte l'uso del CodeView lo stesso Basic dispone di un editor avanzatissimo con (finalmente) una serie di comandi UNDO-REDO bufferizzati; è possibile eseguire ordini UNDO per ritornare fino a 20 passi indietro nella scrittura del codice.

Non manca un menu di utility customizzabile in cui assegnare comandi di DOS, shortcut, intere linee di comandi o altro. Inoltre esiste una vera e propria sezione finalizzata alla customizzazione dei tasti funzione. Ad esempio, se si preferisce usare un set di comandi di editing cui si è già abituati (molti ad esempio usano un word processor per redigere i programmi) invece di quelli specifici di QBX (che sarebbero preferibili, comunque, essendo senz'altro più potenti e finalizzati) è possibile caricare uno dei quattro keyfile precostruiti che possono essere comunque customizzabili; se proprio uno è davvero incontentabile può crearsi tutto a suo piacimento con l'utility MKKEY. Ancora è possibile eseguire output su stampanti multiple, controllare completamente una serie estesa di opzioni del compilatore, e gestire un on-line Help di caratteristiche estremamente raffinate.

Utilizzabilità del sistema e montaggio del linguaggio

Microsoft Basic 7 richiede la seguente configurazione minima:

- un calcolatore della classe XT (minimo) con DOS 3.0 o successivo o OS/2 versione 1.1. o successiva,
- un Hard Disk,
- almeno 60K di memoria RAM.

Sono completamente supportati Microsoft Mouse e ogni altra periferica del genere (trackball o altro) compatibile con le specifiche di Microsoft Mouse.

Tutto l'ambiente di sviluppo, come avevamo accennato precedentemente, è racchiuso in dodici dischetti da 5"1/4; non è possibile semplicemente copiare i programmi sull'hard disk in quanto molti dei file presenti sono in forma compattata. Occorre a tal uopo usare il programma SETUP.EXE che scompatta i file (usando il programma UNPACK.EXE), costruisce le librerie e esegue una serie diversa di operazioni per rendere utilizzabile l'ambiente di sviluppo. Il file SETUP.EXE è compreso nel disco 1 e può essere utilizzato in uno dei seguenti modi:

- installare il package completo, incluso l'ambiente QBX, i tool di sviluppo di linguaggio misto, le utility e le librerie;
- eseguire una installazione mirata, ad esempio escludendo dalla installazione librerie o tool di cui non si prevede un utilizzo o un bisogno immediato.

La seconda opzione, indubbiamente, consente di risparmiare un bel pezzo di memoria, ed è preferibile per chi desidera accedere alle caratteristiche del linguaggio in maniera graduale e sistematica; infatti Setup è sempre rilanciabile per aggiornare l'ambiente mediante l'installazione selettiva di altre caratteristiche non incluse nella prima installazione.

Il lancio è del tutto autodocumentante: esso contiene tutte le informazioni e gli HelpScreen necessari al completo processo di installazione self-tailored. È possibile, attraverso di esso, specificare path e directory, specificare librerie e moduli di runtime, selezionare i file da installare (nel caso di installazione selettiva o di file specifici o di parti di MS Basic). L'installazione avviene in circa un'ora per macchine basate sull'8086 per ridursi a mezz'ora su macchine della classe 386. La velocità di installazione è condizionata molto dalla presenza in memoria di programmi residenti (come, ad esempio, GRAPHICS), che è opportuno deinstallare dalla memoria prima del lancio del SETUP.

La quantità di disco rigido occupata è ovviamente funzione del materiale che si desidera installare. Ovviamente la prima volta che si installa il linguaggio, può sorgere qualche dubbio sul materiale da caricare; in questo caso è prevista una installazione di default, pilotata attraverso un menu, che consente anche di eseguire modifiche a precedenti installazioni. Ancora, durante una installazione, vengono posti nelle varie directory una serie di file di tipo .EXE, .LIBE e .OBJ, come esempi o documentazione che è possibile cancellare per far posto sull'Hard Disk.

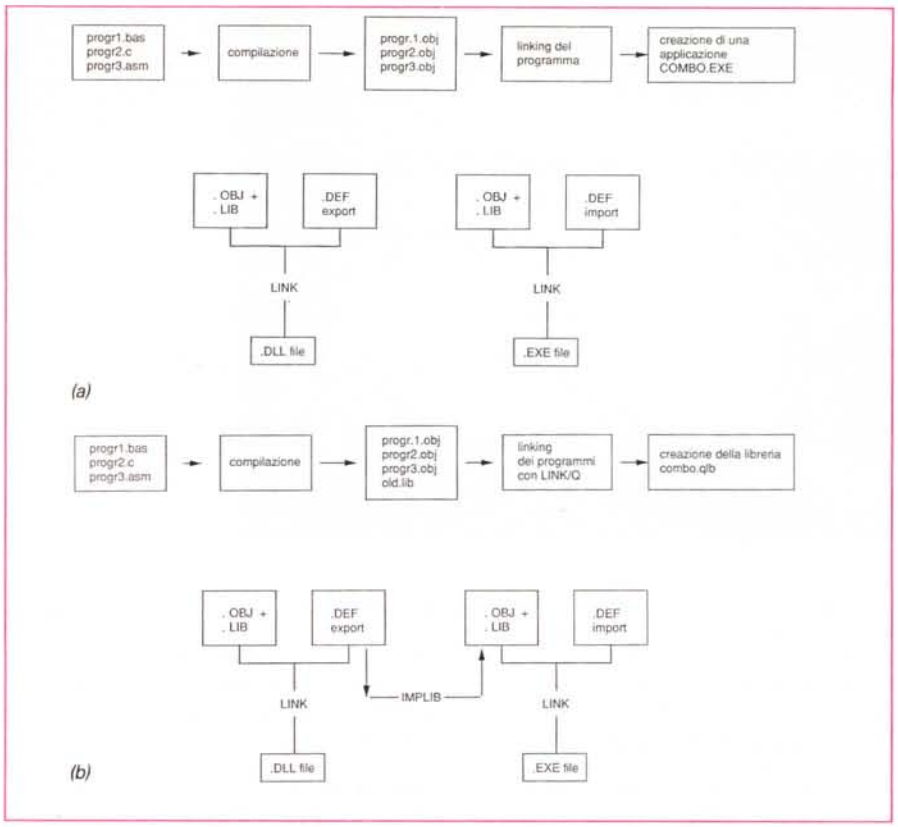
La gestione della memoria e l'uso di applicazioni complesse

Con un ambiente di tali dimensioni è ovvio che il problema più complesso da affrontare è quello della corretta gestione della memoria. La manipolazione di questa da parte di QBX può avvenire in maniera estremamente complessa, appena appena si desideri scrivere un programma un po' più grande di qualche pagina. L'ambiente QBX e il blocco ISAM, insieme, richiedono 450K di ROM (il solo QBX ne abbisogna di 300); con una macchina di 640K di memoria convenzionale possono essere supportati programmi di grandezza solo moderata; per «pièces» di maggiore estensione può essere necessario l'uso della

memoria estesa, specie quando si usa il blocco ISAM.

Il problema di disporre della massima memoria possibile può essere affrontato in vario modo; premessa la rimozione di tutti i programmi residenti in memoria, occorre affidarsi a memoria aggiuntiva installata attraverso l'aggiunta di moduli o schede. QBX sposta blocchi di circa 64K del suo codice fuori della memoria convenzionale (quando espansioni di memoria sono disponibili) in maniera del tutto trasparente. Inoltre è possibile utilizzare la memoria espansa come buffer ampio fino a 1.2 mega, come supporto per i programmi utilizzanti l'ISAM. L'utilizzo della memoria espansa avviene a mezzo di driver appositi (uno di questi [HIMEM.SYS] è fornito col package, ma è possibile utilizzarne altri, preferiti dal programmatore), ma, come prevedibile, solo un driver alla volta può essere presente; inoltre l'uso di emulatori di memoria espansa interferisce con i driver presenti per cui occorre scegliere tra l'uno o l'altro. È possibile inoltre utilizzare una opzione [/nofrills, abbreviata /nof] per recuperare memoria addizionale al programma.

L'uso di memoria estesa può avvenire, comunque, solo con macchine dotate di processore 286 o successivi; la memoria estesa viene gestita inoltre attraverso le specifiche di XMS [eXtended Memory Specification]. L'uso di memoria espansa, invece, è possibile con tutta la famiglia 8086; fino a 64K di memoria indirizzabile vengono utilizzati attraverso un meccanismo di paginazione della memoria. Il tutto viene gestito attraverso le specifiche EMS [Lotus-Intel-Microsoft Expanded Memory Specification], sia a mezzo dei driver forniti nel package [RAMDRIVE.SYS e SMARTDRV.SYS] sia attraverso i driver forniti direttamente dai produttori dell'hardware installato. I driver appena nominati consentono, attraverso una serie di comandi con una sintassi piuttosto articolata di creare anche una RamDisk memory e una area di disk-cache.



Uso del linker per operazioni di creazione di file .EXE con (a) o senza l'uso di libreria (b).

Alcune caratteristiche tipiche di Basic 7

La vera particolarità di questo Basic è quella di essere modulare; procedimento già adottato in altri linguaggi e portato a regola di vita nel «C» e nel Forth, permette di caricare in memoria solo una parte di tutte le istruzioni disponibili, attraverso l'acquisizione di moduli specifici. Vediamo alcune caratteristiche interessanti dei diversi blocchi componenti.

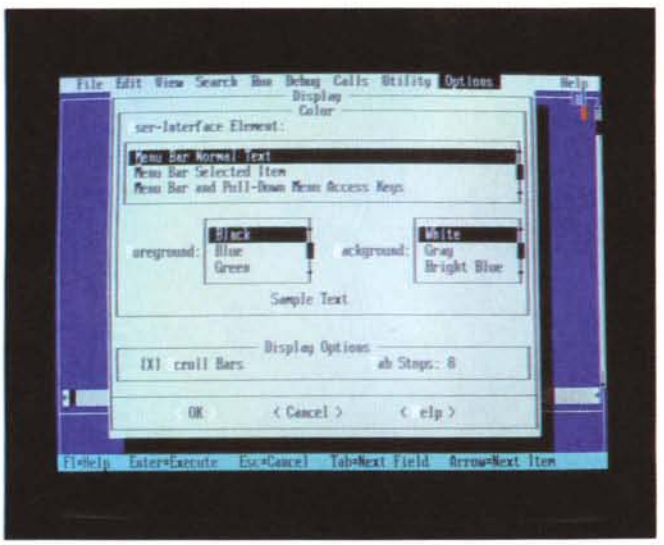
Il modulo principale contiene il vero e proprio Basic Standard; si tratta della versione più evoluta, paragonabile in gran parte con quanto già presente in QuickBasic ultima versione; esistono i più completi tool di controllo di flusso [FOR...NEXT, DO...LOOP, WHILE ...WEND, SELECT...CASE], di definizione di funzioni e procedure [FUNCTION, CALL, SUB, DECLARE, COMMON], di operazioni di I/O (che soprattutto per quanto attiene alla gestione dei file sono sofisticate e molto funzionali, con la disponibilità di comandi DOS-oriented come quelli precedentemente descritti), di manipolazione di stringhe (tra cui alcuni inediti puntatori). La grafica è purtroppo non molto di più di quella vista in altre versioni e risente in maniera pesante della gestione pixel-oriented. Dove invece sono stati fatti passi da gigante è nella gestione dell'event ed error trapping; abbiamo qui a disposizione una ventina di operatori che vanno dal più semplice [ON ERROR...GOTO] alla più sofisticata gestione delle diverse modalità d'errore (anche di questo abbiamo parlato già).

Il salto di qualità e di prestazioni avviene quando si accede alle add-on-library (genericamente raggruppate nella cateogria xxx.LIB o xxx.QLB), articolate in tre gruppi: DATE-TIME, FINANCIAL e FORMAT. Tutte ricordano (e non poteva essere che così) quanto già visto in Excel e consentono, le prime una gestione numerica delle operazioni su data, le seconde una serie molto potente e articolata di funzioni finanziarie (es. NPV, PV, FV, RATE#, DDL, e numerose altre), le ultime, infine, una completa operazione di formattazione sui tipi di dati.

Il ToolBox, anzi per essere precisi i file di ToolBox sono dedicati a operazioni specialistiche di notevole livello; essi sono raggruppati per categorie di utilizzo e sono:

- procedure di matematica su matrici
- procedure di presentation graphic attraverso SUB e FUNCTION
- gestione di font
- gestione della interfaccia utente con procedure finalizzate alla creazione e utilizzo dei menu, finestre, mouse e altro.

Addentrarsi in questo mondo di utilizzo è per lo meno azzardato, con lo spazio che abbiamo a disposizione. Ricorderemo, a braccio, tutta la serie di operazioni su matrici e determinanti, la



Anche il look esige la sua parte; l'user interface element, per il setup delle opzioni d'ambiente.

possibilità di creare grafici a torta e a barra, con colori definibili dall'utente, la costruzione di window estremamente personalizzate e correlate in maniera articolata, la possibilità di creare comandi personalizzati attraverso l'uso di bottoni e di campi di input e di editing, l'uso di window mouse-sensitive. Ma qui le cose divengono troppo lunghe da descrivere e, probabilmente, il doppio delle pagine che Marinacci mi ha messo a disposizione non basterebbe.

Prima di chiudere qualcosa sul compilatore [BC]. Si tratta della edizione migliorata di quanto già visto precedentemente; piuttosto veloce (sebbene non abbia trovato indicazioni sui manuali credo che il traguardo delle 15000 linee sia abbondantemente superato) accoglie una serie di migliorie rispetto alla versione 6; tra queste ricordiamo l'opzione /Fs (per l'immagazzinamento nella far-string memory), /G2 (per gestire istruzioni specifiche del processore 80286; mediamente un codice così compilato è almeno il 30% più compatto di quello partorito da un 8086), /Ot (che ottimizza la chiamata alle procedure), /Ix (destinato all'ISAM) /Fp e /Fpi (per la gestione ottimizzata della matematica in virgola mobile). La parte di manuale destinata al compilatore è molto estesa e ben commentata (come quasi tutto, d'altro canto) e permette operazioni di compilazione che forniscono codice generalmente più compatto di quello ottenibile con Quick Basic. Anche le operazioni di linking sono piuttosto semplici e agevoli soprattutto in presenza di librerie redatte in altri linguaggi (tipicamente «C» e Assembler; una possibilità da non scartare è quella di «agganciare» librerie di altri linguaggi, come ad esempio il Graphix Turbo Pascal o similia. Anche qui la documentazione è molto ampia e articolata (circa 120 pagine) e guida in maniera facile (nello spirito del linguaggio) attraverso la non semplice selva del procedimento.

Prima di concludere una occhiata al Microsoft Code View; non è una novità del linguaggio, certo è, ma è opportuno spendere qualche parola su questo tool talora insostituibile, specie in programmi lunghi e complessi.

Code View (per la precisione «Microsoft® Code View debugger and development utilities») è un potente debugger simbolico, utilizzabile con codici prodotti con compilatori Microsoft Basic, C, Fortran, e Pascal, oltre che con Microsoft Macro Assembler.

Con Code View è possibile tenere traccia in maniera ordinata e non molto complessa, attraverso una agevole interfaccia window-oriented, degli errori logici presenti in un programma. Il suo compito, ridotto all'essenza, è quello di mostrare il codice sorgente o assembly, indicando quale linea viene eseguita, quali variabili vengono modificate e quali valori ad esse assegnate, quali schermate vengono coinvolte nell'I/O, e molte altre cose di tal genere. In tutto questo l'uso di CV può essere di grande aiuto, anche perché, per la verità, il suo uso è davvero semplice e intuitivo. Accanto a queste possibilità per così dire immediate esiste la caratteristica di poter intervenire sulle librerie e di poter modificare il file oggetto in modo da essere manutenibile più facilmente; e infine, attraverso alcune utility proprio di questo tool è possibile modificare addirittura direttamente l'ambiente DOS.

Dicevamo che Code View non è certo né nuovo né specifico di QBX. Esso, comunque, in questa versione ha incamerato una serie di nuove caratteristiche che così possiamo riassumere:

- possibilità di controllo e di cross-reference di array, puntatori e strutture
- valutazione automatica e trasparente di espressioni multilinguaggio
- supporto del 386
- supporto della memoria espansa
- emulatore di 8087, con acquisizione di tutte le istruzioni e comandi della serie 7
- completa compatibilità con programmi che usano la tecnica di overlay
- una serie di nuovi comandi, destinati anche alla redirection di I/O e a comandi di startup.

Conclusioni

Massimo 35.000 battute, aveva raccomandato la Molinari. Come racchiudere in così poco, così tanto? Per far capire le dimensioni del leviatano che abbiamo di fronte possiamo solo far cenno alla monumentale documentazione e ai 10 e più mega di software installabile. La conclusione è che qui non si è più di fronte a un linguaggio ma a un vero e proprio ambiente di programmazione.

Certo Basic 7 va in mano al programmatore professionista; ma questo non per gli stessi motivi che fanno dire la stessa cosa di altri linguaggi; qui specifica utilizzabilità professionale significa anche difficoltà e estrema specializzazione dell'utente; in QBX anche il principiante o chi ha giocherellato col GWBasic si ritroverà in un ambiente amichevole, facile da usare, ma che gli può consentire di accedere alle vette più alte e ai risultati più avanzati della programmazione, il tutto, lo ripetiamo, confortati sempre da un supporto (documentazione, help, facilità di approccio) del tutto impensabile in altri tipi di linguaggio.

Per concludere vorrei solo dire che adottare questa versione del linguaggio impone molta umiltà; chi sposa questa versione dovrà sapere che gli saranno richieste ore, giorni o settimane di lavoro e studio intenso per acquisire tutte le potenzialità del linguaggio; Basic 7 non è fatto per il programmatore occasionale solo perché si ritroverebbe con qualcosa che utilizzerebbe solo in parte; meglio in questo caso adottare il Quick-Basic (di cui parleremo prossimamente), meno costoso e più userfriend. Ma chi si avvicinerà col piede giusto a Basic 7 non avrà certo nulla da temere da utenti di altri linguaggi, dal più vecchio Fortran al più avanzato «C». MC

PC MASTER

80286/12

- motherboard Suntac
- Landmark speed 16 MHz
- espandibile a 1/2/4 Mb
- gestione memoria EMS
- Award bios con setup

configurazione base con cabinet da tavolo, 512 Kb RAM, drive 1.2 Mb e tastiera

con hard disk 20 Mb	
sk. dual	L. 1.630.000
sk. VGA 8 bit	L. 1.740.000
con hard disk 40 Mb	
sk. dual	L. 1.840.000
sk. VGA 8 bit	L. 1.950.000
espansione 1 Mb	+ L. 130.000
espansione 2 Mb	+ L. 220.000
VGA 16 bit	+ L. 100.000

80386/SX

- Landmark speed 21 MHz
- espandibile 8 Mb
- NEAT Chip&Tech chipset
- gestione memoria EMS
- Phoenix bios con shadow

configurazione base con cabinet da tavolo, 1 Mb RAM, drive 1.2 Mb e tastiera

con hard disk 20 Mb	
sk. dual	L. 2.180.000
sk. VGA 8 bit	L. 2.290.000
con hard disk 40 Mb	
sk. dual	L. 2.390.000
sk. VGA 8 bit	L. 2.500.000
espansione 2 Mb	+ L. 90.000
espansione 4 Mb	+ L. 430.000
VGA 16 bit	+ L. 100.000

80386/20

- CPU 80386 32 bit
- Landmark speed 28 MHz
- espandibile a 8/16 Mb
- Phoenix bios con shadow

configurazione base con cabinet da tavolo, 2 Mb RAM, drive 1.2 Mb e tastiera

con hard disk 40 Mb	
sk. dual	L. 3.090.000
sk. VGA 16 bit	L. 3.300.000
con hard disk 80 Mb	
sk. dual	L. 3.590.000
sk. VGA 16 bit	L. 3.800.000
con hard disk 150 Mb ESDI	
sk. dual	L. 4.220.000
sk. VGA 16 bit	L. 4.430.000
espansione 4 Mb	+ L. 370.000

80386/25

- Landmark speed 37 MHz
- espandibile 16 Mb
- cache controller Austek
- AMI bios + cache bios
- 32 Kb 25 ns cache

configurazione base con cabinet tower, 4 Mb RAM, drive 1.2 Mb e tastiera

con hard disk 40 Mb	
sk. dual	L. 4.220.000
sk. VGA 16 bit	L. 4.430.000
con hard disk 80 Mb	
sk. dual	L. 4.720.000
sk. VGA 16 bit	L. 4.930.000
con hard disk 150 Mb ESDI	
sk. dual	L. 5.350.000
sk. VGA 16 bit	L. 5.560.000

80386/33

- Landmark speed 53 MHz
- espandibile 16 Mb
- cache controller Austek
- AMI bios + cache bios
- 64 Kb 15 ns cache

configurazione base con cabinet tower, 4 Mb RAM, drive 1.2 Mb e tastiera

con hard disk 80 Mb	
sk. dual	L. 5.020.000
sk. VGA 16 bit	L. 5.230.000
con hard disk 150 Mb ESDI	
sk. dual	L. 5.650.000
sk. VGA 16 bit	L. 5.860.000

disponibili anche 33 MHz Hauppauge!

configurazioni e prezzi di 20/25/33 MHz possono variare

LAPTOP

286/386 da L. 3.900.000!

- super VGA display
- batterie + aliment. 220 V
- hard 40 Mb 28 ms
- slot 8 bit standard
- keypad+frame
- due seriali + pa

NUOVI PREZZI SU TUTTE LE CONFIGURAZIONI!

MONITOR NON INCLUSI!
in pronta consegna
VGA/Msync colori da L. 690.000
VGA/Dual monocr. da L. 199.000

PERIFERICHE & ACCESSORI
vasto assortimento telefonare!

FCH

FANTASOFT COMPUTER HOUSE
Via O.T.Tozzetti 7/b - 57126 LIVORNO
Tel:0586/805.200 - Fax:0586/803.094

Vendita all'ingrosso e per corrispondenza - Prezzi IVA esclusa - Sconti a rivenditori e per quantità