a cura di Corrado Giustozzi

C'era una volta il DOS

di Sergio Polini

Questo è un saluto. Non è un addio, dal momento che a nessuno è dato prevedere cosa riserva il futuro. È certo, comunque, che la rubrica Turbo Pascal è giunta al termine. Impegni personali dell'autore, ma anche l'intensa evoluzione intervenuta nel rapporto tra il programmatore e il suo PC, impongono di porre termine ad una rubrica nata nel lontano gennaio 1988: un'era che, vista con gli occhi di oggi, appare popolata da dinosauri. Mia figlia, allora, aveva appena sei mesi; oggi è una bellissima signorina alle prese con le asperità della seconda elementare. A quei tempi, poteva muoversi solo portata in braccio; poche settimane fa si è potuta permettere di vincere la sua prima gara di sci. Il settore dell'informatica personale vanta progressi altrettanto vistosi.

Ordinai la prima copia del Turbo Pascal nel 1984: mi attirava la possibilità di disporre, con poca spesa, di un linguaggio particolarmente adatto alla programmazione strutturata. Ne rimasi entusiasta. Affascinato dalle opere del suo ideatore, Nicklaus Wirth, in particolare dal classico *«Algorithms + Data Structures = Programs»*, feci qualche esperimento anche con il Modula-2. E scrissi a MC: perché non parlate di questi linguaggi?

La mia lettera non si riduceva alla sola domanda, ma era un tantino argomentata. Corrado il Grande ne rimase incuriosito, tanto che mi contattò. Ne nacque una pluriennale amicizia, condita da una comune passione per il mondo del DOS ma anche di Unix, per il C ma anche per altri linguaggi (Fortran per lui, Pascal per me), per il mondo dei *real programmers*.

Qualche anno dopo, la domanda cruciale: *«Sergio, ricordi quando scrivesti a MC proponendo una rubrica di Pascal?»*, *«Certo!»*, *«Bene. Perché non la fai tu?»*. Potete indovinare la mia risposta; allora mi sembrava più che altro un bel gioco, solo col tempo mi accorsi che si trattava di un impegno non trascurabile. Ma quante soddisfazioni! Dal lettore che, ad uno SMAU di qualche anno fa, mi disse che comprava MC proprio per la mia rubrica, a quello studente di non ricordo più quale università che, tramite MC-link, mi ha chiesto più recentemente aiuto per governare un robot tramite un programma residente pilotato da Windows. Più di ogni altra cosa, comunque, mi ha ripagato il sapere che il mio lavoro sui programmi residenti è stato utilizzato per approntare software teso a facilitare l'accesso al computer da parte di portatori di handicap.

È proprio dura smettere. Ma è anche necessario. Con gli anni sono aumentati, insieme ai capelli bianchi (non troppi, per la verità!), anche gli impegni del mio lavoro «normale» in un grande gruppo bancario alle prese con la rapida evoluzione del mondo finanziario. Soprattutto, sono cambiati radicalmente sia il rapporto tra la «gente» e i computer sia l'attività di programmazione, al punto che risulta impossibile proseguire secondo la consolidata impostazione della rubrica.

Alto e basso livello

Una volta era il DOS: una piatta interfaccia a carattere, ostica per l'utente non esperto; un ambiente difficile per il programmatore, almeno nella misura in cui volesse realizzare applicazioni facili da usare e, insieme, sicure.

Programmare comporta necessariamente la capacità di gestire strutture di dati adeguate alla complessità della realtà di cui il programma si pone come modello; coerentemente con tale esigenza, la rubrica propose quasi fin dall'inizio esempi di manipolazione di liste sequenziali e concatenate, semplici e doppie, di liste di liste.

Contemporaneamente, però, si avvertiva l'esigenza di mettere l'utente al riparo da situazioni tanto banali quanto potenzialmente in grado di mandare a pallino qualsiasi applicazione: una stampante spenta, un drive senza dischetto, la pressione contemporanea dei tasti «Ctrl» e «C». Di qui gli articoli sui cosiddetti errori critici, con tanto di ricorso all'assembler.

Appena un anno prima (febbraio e marzo 1987), nella neonata area Pascal di MC-link si discuteva di scrittura diretta sulla memoria video. Forse i lettori di oggi neppure sanno di cosa si tratti: un intervento diretto sull'hardware, con le tecniche tipiche della programmazione in assembler, al fine di accelerare l'output su video.

L'anno successivo (1989), si trattò di procedure e strutture di dati ricorsive, di analisi lessicale dell'input, ma anche di un problema suscitato dall'abitudine di vedere un'opzione «DOS» nel menu di un'applicazione. La procedura Exec del Turbo Pascal lasciava allora poca memoria a disposizione per una sessione DOS e, quindi, mi parve opportuno chiedere a Kim Kokkonen (presidente della TurboPower Software) l'autorizzazione a illustrare la procedura più efficiente da lui realizzata: ancora una volta l'assembler e la trattazione di aspetti di basso livello, quali il «contesto» di un programma e la gestione della memoria espansa.

Nel 1990, per venire incontro ad un'esigenza chiaramente manifestata dagli utenti di MC-link, proposi una serie di articoli sui programmi residenti, ovvero sull'unico modo di tenere attivi più processi sotto DOS; seguirono routine per la gestione delle eccezioni (quali una divisione per zero o la radice quadrata di un numero negativo), che intervenivano direttamente sui meccanismi adottati dal compilatore per segnalare i famigerati run-time error.

In quegli anni, per programmare «bene» era necessario sia conoscere ed applicare le regole della programmazioe strutturata, manipolando le strutture di dati proposte dai sacri testi dell'informatica, sia intervenire a basso livello sugli interrupt generati dal DOS, sulla sua gestione della memoria e dell'I/O, perfino sull'hardware.

Gli oggetti

Due eventi, che possiamo ricondurre al 1990, incisero profondamente sullo scenario appena delineato.

Sia la Microsoft che la Borland rilasciarono compilatori Pascal in cui la sintassi del linguaggio veniva arricchita da estensioni «orientate all'oggetto». La Microsoft produsse una versione di Windows finalmente in grado di sfruttare in qualche modo le potenzialità dei processori 80386.

Torneremo tra breve a Windows. La programmazione orientata all'oggetto ebbe infatti per prima un impatto sensibile sulla rubrica. Alcuni articoli introduttivi apparsi sul finire del 1990 prepararono il terreno per un'ampia trattazione che, per buona parte del 1991, tentò di svelare ai lettori tutti i segreti della OOP. Si discusse dei concetti della OOP e delle conseguenze che ne derivavano sulla sintassi dei linguaggi di programmazione, si approfondì la differenza tra procedure e metodi, tra metodi statici e metodi virtuali, si dedicò ampio spazio alle strutture di dati della OOP.

Ragionare in termini di oggetti comporta il superamento della tradizionale distinzione tra algoritmi e strutture di dati: un oggetto, infatti, potrebbe essere definito come un dato strutturato dotato di un proprio «comportamento», incarnato nei metodi che costituiscono parte essenziale della sua definizione.

Ne risulta la possibilità di attuare concretamente i principi di *information hiding*, di dare nuova e più pregnante sostanza ai canoni della programmazione per moduli. Brad Cox, autore più volte citato sulle pagine della rubrica, giunse a preconizzare uno stile di programmazione basato su quelli che definì *software IC*, componenti integrati software: come un congegno elettronico, anche un PC, può essere costruito assemblando un opportuno insieme di componenti già pronti, così un programma può essere sviluppato in tempo relativamente breve assemblando «componenti software», cioè oggetti che siano istanze di classi già disponibili o facilmente derivabili da classi già pronte.

La disponibilità di una libreria di classi utilizzabili in molteplici programmi, con una flessibilità sconosciuta alle tradizionali librerie di funzioni e procedure, avrebbe potuto eliminare la necessità del costante ricorso a tecniche di programmazione di basso livello: una classe va usata per quello che fa, prescindendo completamente da come lo fa; i dettagli della

sua implementazione, una volta messi a punto e collaudati, possono essere semplicemente ignorati, dimenticati.

È mutato radicalmente anche il tradizionale concetto di «struttura di dati»: ad alberi, liste e simili, alle sofisticate tecniche necessarie per la loro efficiente manipolazione, si sostituiscono generici *«contenitori»*, distinguibili per i criteri di gestione dei dati elementari invece che per la loro rappresentazione in memoria: un *bag* può contenere più copie di uno stesso oggetto, un *set* no; un *dizionario* contiene coppie di elementi invece che elementi singoli, e così via. Al programmatore non interessa se un particolare contenitore è realizzato, nelle nascoste pieghe della sua implementazione, mediante un array, un albero o una lista; l'implementazione può anche cambiare, ma l'uso della classe contenitore rimane invariato.

La panacea di tutti i mali? Non esattamente.

L'era dei sistemi

Windows 3.0, dicevamo, riuscì a porre finalmente a disposizione dell'utente normale un ambiente capace di sfruttare in qualche modo le potenzialità del processore 80386 (Corrado ed io, come altri, giocavamo con Unix, ma Unix non è mai stato un sistema operativo «di massa»).

Programmare sotto Windows non è difficile, una volta fatta la mano, ma è tutt'altra cosa che programmare sotto DOS: non più questo o quell'interrupt, non più le idiosincrasie dell'hardware, ma una API complessa, la necessità di ragionare per messaggi invece che per chiamate di procedura, strumenti di sviluppo tanto ostici per il programmatore quanto gradevole poteva risultarne il prodotto per l'utente finale.

La OOP poteva venire in aiuto e, in effetti, così è stato: una libreria di classi come ObjectWindows consente di nascondere molti dei dettagli.

Si posero però subito altri problemi.

In primo luogo, il successo di Windows 3.0 sancì l'affermazione dello standard SAA-CUA per le interfacce utente, al punto che, anche sotto DOS, le infinite varianti, tanto diverse tra loro, di un'interfaccia genericamente popolata da finestre e menu a tendina si adeguarono rapidamente. A ciò contribuì indubbiamente anche la Borland, proponendo il suo Turbo Vision per applicazioni DOS: diventava possibile, almeno in teoria, realizzare applicazioni per DOS e per Windows dotate della stessa interfaccia utente; soprattutto, quella possibilità si configurava anche come necessità, in quanto Windows non spazzò via quel DOS su cui, bene o male, era comunque costruito.

Sarebbe stato auspicabile, per il programmatore, poter prescindere dalla scelta tra questo o quell'ambiente per le proprie applicazioni; sarebbe stato estremamente utile poter lavorare soprattutto sulla «sostanza» di un programma, disponendo di facili strumenti per dotarlo dell'una o dell'altra interfaccia (sotto DOS o sotto Windows) secondo le scelte, i gusti, le specifiche necessità dei suoi utenti.

Concettualmente appariva tutto semplice: una gerarchia di classi avrebbe permesso di nascondere i dettagli della implementazione degli stessi elementi d'interfaccia (una finestra, un menu, una dialog box) in ambienti operativi diversi.

Purtroppo, però, Turbo Vision e ObjectWindows differivano profondamente non solo nei dettagli dell'implementazione, ma anche nella struttura della gerarchia di classi e nell'interfaccia di classi solo apparentemente simili: la **TWindow** dell'uno, ad esempio, ha ben poco a che vedere con la **TWindow** dell'altro.

La rubrica Turbo Pascal cercò di offrire il proprio contributo: si partì con alcuni elementi dell'interfaccia, quali il menu e la riga di stato, per proporre classi che agevolassero il porting di applicazioni da DOS a Windows; con lo stesso intento, si approntarono classi che rendessero più spedita la realizzazione di applicazioni MDI sotto Windows (tutte le applicazioni Turbo Vision sono tendenzialmente MDI), si illustrarono unit che consentivano di gestire le stampe da Turbo Vision in modo analogo a quanto avviene sotto Windows, si evidenziò la possibilità di gestire in modo analogo, nei due ambienti, la comunicazione attraverso le porte seriali. Si esplorò anche quel territorio di confine rappresentato dalla programmazione sotto DOS in modo protetto, approfittando della possibilità, offerta dal Borland Pascal, di realizzare programmi DOS in grado di utilizzare qualsiasi DLL, comprese quelle di Windows.

La programmazione visuale e le DLL

Non v'è dubbio, tuttavia, che sono risultati interessanti più i singoli argomenti trattati che il tentativo di gettare un ponte tra DOS e Windows. Nonostante, infatti, esistano librerie di classi utilizzabili per applicazioni destinate a girare sotto diversi ambienti, la guerra di religione tra sostenitori di questo o quel sistema sembra prevalere su qualsiasi istanza di coesistenza pacifica: i fan di Windows e di OS/2 si azzuffano anche più di quanto sapevano fare, all'epoca dei dinosauri, pascaliani, c-isti e fortranari.

Come se non bastasse, le potenzialità della OOP sembrano vanificate, al momento, sia dalla difficoltà di assimilare uno stile di programmazione fortemente innovativo, sia (parere puramente personale) dalla mancanza di sistemi di gestione dei dati orientati all'oggetto di grande diffusione.

Al tempo stesso, l'attività di programmazione sembra trarre beneficio, più che da incapsulazione, ereditarietà e polimorfismo, dai compilatori «visual» e da un diverso tipo di *software IC*, le *DLL*.

Quest'ultimo è forse l'aspetto più interessante. In anni ormai lontani, scrivevo per conto mio il codice necessario per dotare un'applicazione di un'interfaccia tipo foglio elettronico; oggi acquisterei una DLL. Lo scorso anno mi si pose il problema di controlli di EDIT in grado di gestire testo formattato; altri impegni mi hanno impedito di andare oltre un abbozzo di analisi, ma sicuramente avrei acquistato una DLL dotata di controlli di EDIT per testi in formato RTF. Nello stesso modo affronterei il problema della comunicazione seriale.

Il mondo della programmazione è radicalmente mutato; la scelta di un linguaggio appare ormai meno critica della scelta del sistema operativo; i tempi di sviluppo dipendono strettamente dalla disponibilità di una DLL, più che dallo stile di programmazione; un buona familiarità con algoritmi e strutture di dati è probabilmente meno importante della padronanza di una API.

Si imporrebbe, quindi, quanto meno un ripensamento non frettoloso sul senso di una rubrica come questa. Altri impegni, come già detto, mi costringono comunque a chiudere qui il ciclo iniziato in quel lontano 1988, nella speranza che Delphi (che non ho avuto ancora il tempo di esaminare) risolva i problemi oggetto degli ultimi appuntamenti. Quanto al futuro, si vedrà.

Una cosa è certa: non dimenticherò mai la simpatia con cui tanti lettori hanno manifestato interesse per i miei sforzi; non dimenticherò mai, lasciatemelo dire, la soddisfazione che tante volte mi ha procurato lo scoprire che, seduto al mio PC e magari collegato col mio modem, potevo contribuire a risolvere qualche piccolo problema altrui. MC

Sergio Polini è raggiungibile tramite MC-link alla casella MC1166 e tramite Internet all'indirizzo mc1166@mclink.it.

Arrivederci

Come avrete almeno intuito, se non l'avete già letto nel testo di Sergio, con questa puntata si chiude la rubrica di Turbo Pascal. Nuove responsabilità lavorative, e l'obiettiva constatazione che il mondo e la filosofia della programmazione sono radicalmente cambiati, hanno infatti portato Sergio a considerare conclusa questa fase della sua oramai storica rubrica.

Già, storica: perché quando è nata, sul numero 70 di MC, il mondo era assai diverso da com'è ora. Ed anche perché in tanti anni di «onorato servizio» la rubrica non ha praticamente perso mai un colpo, presentandosi sempre come punto di riferimento della programmazione in Pascal prima sotto DOS e poi sotto Windows.

È con un certo dispiacere, dunque, che saluto da queste pagine Sergio, ringraziandolo a nome non solo della redazione ma di tutti voi lettori per la sua costante e rassicurante compagnia in questo nostro lungo viaggio nell'informatica personale; viaggio che abbiamo intrapreso tutti assieme tanti anni fa e che ci ha portato, ed ancora ci sta portando, in luoghi che mai avremmo immaginato. La OOP è uno di questi luoghi (e Sergio, per modestia, ha tralasciato nel suo saluto di dire che i primi articoli sulla OOP in Italia sono stati i suoi sull'Object Pascal ed i miei sul C++, scritti di concerto; per non parlare del libro realizzato a quattro mani), la programmazione visuale è un altro, e chissà quanti ve ne sono ancora da visitare in futuro.

A Sergio, che oltre ad essere un valido collaboratore è da tanti anni soprattutto un grande amico, auguro innanzitutto di riuscire al meglio nel suo nuovo incarico (e, conoscendolo, non ho alcun dubbio che lo farà!), e poi di ritrovarci ancora, in futuro, assieme su queste pagine. Per parlare magari di qualcosa di nuovo e stimolante, qualcosa di diverso e rivoluzionario... com'era il Turbo Pascal quando, ai tempi dei tempi, decidemmo di dedicargli una rubrica fissa.

Corrado Giustozzi