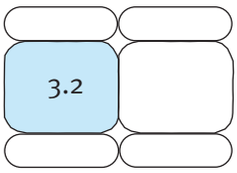
EVOLUZIONE DEI SISTEMI OPERATIVI

IIª Parte: categorie particolari

Maurelio Boari
Paolo Ancilotti

Esaminate nella prima parte le motivazioni che hanno influenzato l'evoluzione del concetto stesso di sistema operativo, sono ora messe in evidenza le motivazioni che hanno portato all'evoluzione di particolari categorie di sistemi operativi: da quelli per personal computer ai sistemi *real-time* ed *embedded*, dai sistemi operativi di rete e distribuiti a quelli per la generazione di macchine virtuali.



1. SISTEMI OPERATIVI PER PERSONAL COMPUTER

Con lo sviluppo dei circuiti integrati su larga scala, negli anni '70 cominciò l'era dei personal computer (inizialmente chiamati microcalcolatori). In termini di architettura non erano molto diversi dai minicalcolatori della classe dei PDP-1, ma erano caratterizzati da un costo allora molto più basso. I primi sistemi realizzati furono chiamati *home computer* ed erano prevalentemente destinati al mondo dei giochi.

Per più di 10 anni lo sviluppo dei personal computer è stato caratterizzato da due linee parallele che solo successivamente sono confluite. La prima, praticamente coincidente con le attività di ricerca svolte presso lo *Xerox PARC* (*Paolo Alto Research Center*), la seconda relativa allo sviluppo dei primi personal computer commerciali.

È dai laboratori della Xerox che nasce il concetto stesso di personal computer inteso come dispositivo sufficientemente potente e facile da usare in grado di soddisfare le necessità individuali di elaborazione, comunicazione e condivisione di informazioni di ogni singolo utente. Alan Kay pubblica il primo articolo sui personal nel 1972 [6].

Nel 1973, presso i laboratori della Xerox Parc fu realizzato il primo personal computer sperimentale, il sistema Alto [15], che ha rappresentato forse il progetto più innovativo nella storia dell'informatica, troppo innovativo per l'epoca, ma destinato ad avere un'influenza notevole negli sviluppi futuri. È stato il primo sistema ad utilizzare un'interfaccia grafica (GUI - *Graphical User Interface*), inventata negli anni 60 presso lo Standford Research Institute, completa di finestre, icone, menù ed un mouse per consentire l'accesso all'utente, al posto delle interfacce a linee di comandi utilizzate nei sistemi operativi dell'epoca. È in quel periodo che presso gli stessi laboratori fu progettata la prima stampante laser, la rete locale Ethernet, il primo linguaggio ad oggetti (*Smalltalk*), il linguaggio Mesa e tutto ciò fu reso disponibile nel calcolatore Alto dotato di un sistema operativo implementato con la tecnologia ad oggetti.

Il sistema Alto, troppo costoso per l'epoca,

non fu mai sviluppato commercialmente ma ne furono realizzati molti esemplari donati a diverse università. Queste macchine, fra l'altro, furono di ispirazione per la realizzazione della workstation *Stanford University Network* (SUN), che venne in seguito commercializzata da un'azienda nata proprio nell'università e chiamata Sun Microsystems.
Con riferimento alla seconda linea di sviluppo, il primo sistema operativo per microprocessore dotato di floppy disc fu il CP/M (*Control Program for Microcomputer*) della Digital Research, progettato da Gary Kildall nel 1973 [8], che poteva essere installato su architetture del tipo Intel 8080 (8 bit), Zilog80 ed altre. CP/M fu il più importante sistema operativo per microcalcolatori negli anni 70. Il primo personal computer sul quale fu installato il CP/M fu l'Altair nel 1975. Il personal computer con CP/M probabilmente più venduto è stato l'home computer Commodore 128.
Molte delle caratteristiche del CP/M furono ereditate dai successivi sistemi operativi, in particolare da quelli adottati per i primi personal computer dell'IBM. Innanzitutto l'interfaccia a linea di comando (derivata dai sistemi operativi della *Digital Equipment* per il PDP-11). In secondo luogo la suddivisione del sistema operativo in due moduli: il primo (CCP) era destinato alla traduzione dei comandi dell'utente, il secondo alla loro esecuzione. Quest'ultimo era composto da una parte BDOS (*Basic Disk Operating System*), che si occupava di gestire le driver di tipo logico per le operazioni richieste dall'utente (per esempio, "apri il file") e le trasformava in una sequenza di comandi inviati al BIOS (*Basic I/O System*) che conteneva il codice dipendente dallo specifico hardware.
I primi prodotti software per il word processing (Word Star) e per la gestione dei database (*dBASE*) furono creati per il CP/M e poi, modificati e potenziati, per i successivi sistemi operativi.
Quando IBM, nei primi anni 80, progettò il PC IBM decise di adottare come sistema operativo il CP/M. Non trovando l'accordo con la Digital Research, adottò come sistema operativo un prodotto della Microsoft denominato DOS (*Disk Operating System*), più tardi denominato MS-DOS (*Microsoft Disk Operating System*) che rapidamente cominciò a dominare il mercato dei PC. La sua diffusione fu facilitata dalla decisione di Bill Gates di vendere MS-DOS alle aziende che producevano calcolatori, perché formasse un unico pacchetto con l'hardware, a differenza della politica allora adottata da *Digital Research* di vendere direttamente CP/M agli utenti finali, un sistema alla volta.
MS-DOS, come CP/M, era un sistema operativo monoutente e monotask, cioè capace di far girare un solo programma alla volta. Inoltre, pensato per il microprocessore Intel 8088 (16 bit) fu progettato con lo scopo prioritario di fornire la massima efficienza, tenuto conto dei limiti del processore. Ciò rese praticamente impossibile una sua trasformazione per sfruttare le maggiori capacità dei nuovi processori che via via si resero disponibili. L'8088, per esempio, non offriva i due stati di funzionamento, user mode e kernel mode e ciò consentiva ai programmi di utente di accedere direttamente alle componenti hardware di base, per esempio utilizzando le operazioni di I/O per scrivere direttamente su video e disco. Inoltre, nella sua semplicità, MS/DOS fu progettato come un sistema monolitico nel quale l'interfaccia ed i livelli di funzionalità non erano ben separati e fu scritto utilizzando il linguaggio assembler del 8088.
Negli anni furono sviluppate diverse versioni di MS/DOS [7] arricchite di nuove funzioni che aumentarono la parte residente in memoria centrale del sistema operativo. Quando, per esempio, IBM introdusse il PC XT dotato di disco fu sviluppata la versione DOS 2.0 che forniva un sistema di gestione dei file di tipo gerarchico (inizialmente tutti i file appartenevano ad un solo livello). Nelle successive versioni predisposte per le nuove serie di PC dotate di microprocessori sempre più potenti (80286, usato nel PC/AT, 80386, usato in PS/2) MS-DOS si arricchì di nuove funzioni, ampliando, per esempio, l'insieme dei comandi a disposizione dell'utente ed introducendo alcune proprietà simili a quelle di Unix, come la ridirezione dell'I/O e la stampa in background (Microsoft aveva realizzato una versione di Unix chiamata Xenix).
Tuttavia, come detto, la struttura del sistema operativo impediva che potessero essere sfruttate appieno le capacità dei nuovi processori. Per rimanere compatibile con le precedenti versioni, il sistema operativo li utilizzava

come un "8086 veloce". Cioè, si continuò ad usare MS-DOS in un ambiente hardware ben superiore alle sue capacità, a maggior ragione dopo l'introduzione del 80486 e del chip Intel Pentium che avevano reso disponibili potenza e caratteristiche che non potevano essere sfruttate da un sistema tanto elementare[1].
Un altro problema era rappresentato dal tipo di interfaccia a comandi usato da MS-DOS. Per competere con il sistema Macintosh della Apple (si veda più avanti), nei primi anni 80 Microsoft cominciò a pensare ad un nuovo sistema dotato di un'interfaccia utente grafica simile appunto a quella del sistema Macintosh. Dopo molti tentativi uscì nel 90 il sistema Windows 3.0. L'esigenza di mantenere la compatibilità con le numerosissime applicazioni che funzionavano su MS-DOS fecero sì, tuttavia, che Windows 3.0 fosse pensato più come un'interfaccia grafica che estendeva le funzionalità di MS-DOS, piuttosto che come un nuovo sistema operativo a se stante.
Parallelamente alla commercializzazione ed allo sviluppo di Windows 3.0 e successive versioni, la Microsoft cominciò a sviluppare un nucleo di sistema operativo multiprogrammato che avesse tutte le caratteristiche dei sistemi operativi allora in auge, come Unix e VMS. Il risultato di tale sforzo fu il rilascio del sistema Windows NT[2], rivolto prevalentemente al mercato professionale ed al mercato dei server. Windows NT [13] offre la multiprogrammazione, la gestione della memoria virtuale, la multiutenza, la protezione delle risorse, il supporto al networking. Inizialmente venne adottata una struttura a microkernel, con un nucleo molto ridotto che forniva le funzionalità elementari a moduli di più alto livello operanti a livello utente. Successivamente, per motivi di efficienza, tutti i moduli del sistema operativo furono portati all'interno del nucleo, compreso il sottosistema di gestione grafica delle finestre.
Una delle caratteristiche di NT è la possibilità di garantire la compatibilità con altri sistemi operativi. È così possibile eseguire applicazioni MS-DOS, OS/2 e POSIX oltre a tutte quelle Windows.
Per un certo periodo, la Microsoft continuò a fornire il supporto a due diverse linee di prodotti: sistemi operativi per l'utenza casalinga, quali Windows 95, Windows 98 e Windows Me; e sistemi operativi per l'utenza professionale quali Windows NT 3.5, NT 4.0 e Windows 2000.
Windows 95, Windows 98 e WindowsMe possono considerarsi delle evoluzioni, non sempre ben riuscite, del sistema Windows 3.0, basate cioè sempre su MS-DOS anche se con un miglioramento in termini prestazionali dovuto ad un parziale adattamento alle architetture a 32 bit. Windows 95 è noto per la sua innovativa e molto efficace interfaccia grafica, mentre fu con Windows 98 che Internet Explorer fu integrato nell'interfaccia grafica sollevando numerose critiche poiché di fatto danneggiava fortemente gli altri sistemi browser. Ne sono scaturite diverse cause legali, in seguito alle quali le autorità antitrust hanno imposto a Microsoft di permettere all'utente la disabilitazione di Internet Explorer.
NT 4.0 e Windows 2000, evoluzioni della prima versione di NT, hanno contribuito alla penetrazione di Microsoft nel settore dei server, fortemente presidiato da Unix e LINUX.
Recentemente la Microsoft ha inteso semplificare il supporto e lo sviluppo dei propri sistemi operativi unificando le due linee di prodotti in un unico sistema operativo, Windows XP [21] che viene commercializzato in due versioni, Home e Professional, con alcune differenze nella tipologia dei servizi supportati. Nel corso del 2008 dovrebbe essere completata la distribuzione della nuova versione Vista.
Da notare che con la comparsa sul mercato del primo PC/AT IBM con processore 80286, che per la prima volta in un personal computer forniva il supporto alla modalità protetta per il sistema operativo, IBM e Microsoft collaborarono per un certo periodo per la realiz-

[1] Per superare i limiti imposti da 8086 furono introdotte funzioni software come i gestori di memoria EMS e XMS, e i cosiddetti DOS extender che gestivano il funzionamento a 32 bit in modalità protetta, che MS-DOS non poteva supportare: ma in ogni caso la gestione del file system e la risposta agli *interrupt* restavano comunque in mano a MS-DOS e soffrivano dei suoi limiti

[2] La sigla **"NT"** non ha un significato ufficiale, tuttavia viene comunemente interpretata come le iniziali di **"N**ew **T**echnology"

zazione di un nuovo sistema operativo che potesse sfruttare la potenza dei nuovi processori. Il tentativo di collaborazione fallì e mentre Microsoft realizzava il sistema NT, IBM intraprese un progetto che doveva portare alla realizzazione del sistema operativo OS/2 [2] con le stesse caratteristiche di multiprogrammazione e multiutenza di NT. OS/2 conobbe negli anni 90 una certa diffusione, ma lentamente venne abbandonato da IBM.
Come si è detto precedentemente, quando Microsoft decise di costruire il successore di MS-DOS fu fortemente influenzata dal successo di un altro personal computer, il Macintosh, che utilizzava un'interfaccia grafica. Questo tipo di interfaccia si diffuse molto rapidamente solo quando Steve Jobs, in seguito ad una visita ai laboratori dello Xerox Parc, decise di adottarla nei suoi calcolatori Apple. Il primo tentativo di Jobs fu Lisa che si rivelò troppo costoso e fallì commercialmente. Il secondo fu Macintosh, il cui progetto iniziò nel 1979 [9], che ebbe invece un grande successo, non solo perché più economico di Lisa, ma grazie alla sua facilità d'uso. La sua interfaccia grafica usava per la prima volta metafore facili da comprendere, quali il *cestino*, la *scrivania*, le *finestre*, gli *appunti* ecc. aprendo l'uso del computer anche a persone che non conoscevano nulla di calcolatori. Per questo motivo il Macintosh è divenuto una pietra miliare nello sviluppo dell'industria del computer. Inizialmente Macintosh utilizzò il microprocessore Motorola 68000, successivamente, a seguito di un accordo con IBM e Motorola, fu adottata una nuova serie di CPU RISC, detta PowerPC. Recentemente è avvenuto il passaggio ai nuovi processori Intel.
Per quanto riguarda il sistema operativo, la prima versione MAC OS non presentava caratteristiche particolari rispetto agli altri semplici sistemi operativi dell'epoca tranne quella di supportare l'interfaccia grafica. Era scritto per la maggior parte in assembler 68000 (una parte anche in Pascal) e fu gradualmente convertito in codice PowerPC.
Analogamente a quanto successo per Microsoft con il passaggio al sistema operativo NT, anche Apple decise di sostituire MAC OS con un sistema operativo completamente nuovo. La soluzione finale fu quella di integrare il sistema operativo OpenStep della NeXT, una società fondata da Steve Jobs dopo la sua uscita da Apple, basato su kernel Mach con codice BSD Unix integrato, con la parte di gestione dell'interfaccia grafica del Mac OS. Il nuovo sistema fu chiamato Mac OS X [19] ed è quello attualmente utilizzato.
Inizialmente il successo del Mac fu frenato dal suo limitato parco software. Nel 1985 la combinazione del Mac con la sua GUI, di Adobe PageMaker e della nuova stampante laser di Apple diedero vita ad una soluzione a basso costo per l'editoria e la grafica pubblicitaria, un'attività che sarebbe diventata famosa con il nome di Desktop Publishing (DTP). L'interesse per il Mac esplose, tanto che costituisce a tutt'oggi un diffuso standard presso le tipografie, gli studi di grafica e le aziende editoriali.
L'altro sistema operativo che contende a Windows e a MAC OS il mondo dei personal è Unix, compresi i suoi derivati, incluso Linux. Quest'ultimo, in particolare, ha acquisito un notevole successo tenendo anche conto che è un sistema open source. Sebbene molti utenti dei sistemi Unix-like siano abituati ad usare un'interfaccia a comandi, tutti i sistemi Unix oggi forniscono anche un'interfaccia grafica chiamata X Windows, realizzata al MIT nel 1984, che permette di gestire finestre tramite mouse. Spesso è poi disponibile Motif, che gira sopra X Windows, e che costituisce una completa GUI simile alle interfacce di Windows e di MAC OS.

Olivetti M20 e M24

Una linea di personal computer fu sviluppata da Olivetti negli anni '80, prima con il sistema M20 e successivamente con il sistema M24.
Il sistema M20 fu sviluppato da Olivetti nell'*Advanced Research Center* di Cupertino e presentato al pubblico nel 1982. Studiato per attirare i potenziali acquirenti del PC IBM risultava incompatibile con quest'ultimo a causa del sistema operativo, il PCOS, interamente sviluppato da Olivetti, e dalla scelta di un microprocessore non troppo diffuso, lo Zilog 2001. La non compatibilità di PCOS con MS-DOS, limitò la diffusione del M20 per l'impossibilità di utilizzare il software che in quantità sempre maggiore e a livello mondiale veniva sviluppato per il sistema operativo della Microsoft. Olivetti sviluppò allora un nuovo sistema, M24, che adottava MS-DOS. L'hardware si basava su un processore Intel 8086 (8 MHz e 16 bit), più potente del 8088 del PC IBM, in grado di contenere oltre al disk-drive, anche un hard-disk Winchester da 5 Mbit e con una grafica migliorata rispetto al predecessore. Il sistema ebbe un notevole successo. Di fondamentale importanza per la sua diffusione si rivelò l'accordo firmato dalla Olivetti con AT&T, il colosso americano delle telecomunicazioni: in seguito a quel accordo le vendite dell'Olivetti M-24 aumentarono in modo significativo, al punto che l'Olivetti divenne, nel 1985, uno dei maggiori produttori di personal computer a livello mondiale.
Le note vicende industriali portarono il gruppo di Ivrea gradualmente all'uscita dal settore dei personal computer e dell'informatica in generale.

2. SISTEMI REAL-TIME, EMBEDDED, MULTIMEDIALI

La possibilità di usare il calcolatore digitale nel risolvere problemi di controllo di apparecchiature o di processi industriali si è dimostrata, fin dalle origini, come una valida alternativa all'uso di calcolatori analogici o comunque di circuiti sviluppati ad hoc per ciascuna applicazione. Ciò ha favorito lo sviluppo di una terza tipologia di sistemi operativi: i *sistemi in tempo reale*, diversi sia dai sistemi batch che dai sistemi time-sharing. In questo tipo di applicazioni il calcolatore è dedicato a controllare il corretto funzionamento di un sistema fisico (*ambiente operativo*) che può essere costituito da un impianto industriale, da una centrale elettrica o di telecomunicazioni, da una centralina per il controllo motore di un'auto, da un robot o comunque da un'apparecchiatura di cui sia necessario mantenere sotto controllo le grandezze fisiche che la caratterizzano. Il sistema di calcolo, mediante opportuni sensori, legge periodicamente il valore delle grandezze fisiche da controllare (pressioni, portate, temperature ecc.). Una volta letto il valore di una grandezza, esegue un programma che verifica se tale valore rientra nell'ambito dei valori corretti oppure se si verifica uno scostamento rispetto a questi. In quest'ultimo caso, il sistema di calcolo retroagisce sull'ambiente operativo mediante opportuni attuatori cercando di riportare la grandezza in questione nel range dei valori corretti.

Ciò che caratterizza il comportamento di un sistema in tempo reale è il fatto che l'esecuzione di ogni programma di controllo deve terminare entro un ben definito intervallo temporale imposto dall'applicazione (*deadline*). L'obiettivo fondamentale di questi sistemi operativi non è, quindi, quello di massimizzare l'efficienza di uso delle risorse, come nei sistemi batch, o quello di minimizzare i tempi di risposta, come nei sistemi time sharing, ma quello di garantire, a priori, che le esecuzioni di tutti i programmi di controllo siano completate entro le rispettive deadline. In altri termini, la validità dei risultati prodotti da un programma applicativo dipende, non solo dalla correttezza del programma, ma anche dall'intervallo temporale entro il quale i risultati sono prodotti. In questo senso la parola **tempo**, entra nel nome di questo tipo di sistemi operativi. L'aggettivo **reale** denota poi il fatto che la risposta del sistema agli eventi che si verificano nell'ambiente operativo deve avvenire durante l'evolversi degli eventi stessi.

Come indicato precedentemente, l'esigenza di avere a disposizione questo tipo di sistemi è nata con i calcolatori stessi, alimentata soprattutto da ricerche in campo militare, e l'evoluzione dei relativi sistemi operativi è avventa parallelamente a quella delle altre tipologie di sistemi.

Il primo calcolatore che, in assoluto, è stato sviluppato per essere utilizzato in applicazioni in tempo reale è il Whirlwind [3], sviluppato da Jay Forrester presso il laboratorio di servomeccanismi del MIT a partire dal 1946, divenuto operativo nel 1949. L'idea di sviluppare quel progetto nasce durante la Seconda Guerra Mondiale quando la U.S. Navy contattò il MIT per esplorare la possibilità di realizzare un simulatore di volo come dispositivo di addestramento dei piloti. Il progetto, noto come *Project Whirlwind*, prevedeva un sistema collegato ad una consolle che aggiornasse in tempo reale i parametri della consolle seguendo un modello fisico dell'aerodinamica del velivolo. L'esigenza di poter modificare il modello fisico da simulare favorì, in seguito, l'idea di progettare un dispositivo programmabile e quindi di sviluppare un calcolatore elettronico per il progetto Whirlwind. In quel periodo storico i calcolatori, come abbiamo visto, funzionavano soltanto in modalità batch, mentre il sistema Whirlwind doveva adattarsi alle manovre del pilota da addestrare rispondendo quindi in tempo reale alle variazioni dei segnali di input. Il tipico comportamento di un sistema batch avrebbe perciò limitato fortemente le funzionalità del simulatore.

Il progetto del calcolatore durò tre anni con un budget di un milione di dollari (dell'epoca) all'anno. Il risultato fu un calcolatore che conteneva 12.500 valvole termoioniche e fu il primo calcolatore a fornire grafici e testi in tempo reale su un display (in pratica un oscilloscopio). Fu il primo ad utilizzare i nuclei magnetici come componenti elementari della memoria centrale (componenti che saranno impiegati in maniera massiccia solo negli anni 60). Fu anche il primo calcolatore ad utilizzare una ROM implementata come una matrice a diodi che influenzò, successivamente, l'introduzione della microprogrammazione da parte di M. Wilkes. Grazie

anche a queste innovazioni, Whirlwind diventò il calcolatore più veloce dell'epoca. Successivamente, iniziò il progetto di un calcolatore ancora più veloce: il Whirlwind II, ma a causa dell'enorme quantità di risorse necessarie, il MIT decise di non andare oltre al progetto.
Nel 1953, l'aeronautica militare americana decise di finanziare l'ambizioso progetto di un sistema di comando e controllo per la difesa aerea degli Stati Uniti: il progetto SAGE (*Semi-Automatic Ground Environment*) [5] col compito di realizzare un sistema di calcolatori destinati a ricevere, mediante linee telefoniche, le informazioni provenienti da una serie di stazioni radar, elaborarle ed automatizzare l'invio di informazioni per comandare i mezzi aerei destinati ad intercettare eventuali attacchi.
Il solo calcolatore che fu ritenuto valido e sufficientemente veloce come punto di partenza per tale progetto fu il Whirlwind e fu proprio dal progetto del Whirlwind II che iniziò, presso i Lincoln Laboratories del MIT (da cui sarebbe successivamente nata come spin-off la Mitre Corporation) lo sviluppo di quello che è unanimemente riconosciuto come il più complesso, costoso e innovativo progetto informatico di tutti i tempi. Il calcolatore, la cui parte hardware fu sviluppata da IBM, fu chiamato con la sigla A/NFSQ-7 ma sostanzialmente coincideva col Whirlwind II. Il sistema fu pienamente funzionante solo dieci anni dopo l'inizio del progetto e rimase in funzione fino ai primi anno 80. Le innovazioni che furono introdotte nel progetto e che ebbero notevoli influenze nello sviluppo di sistemi successivi, furono numerose; contribuì ad introdurre il concetto di sistema on-line, di interattività, di multitasking, l'uso di linee telefoniche per la comunicazione di dati usando modem, l'uso della penna ottica che successivamente avrà influenza nello sviluppo del mouse. Ogni calcolatore A/NFSQ-7 consumava circa 3 MW di potenza, aveva circa 60.000 valvole termoioniche e impiegava più di 100 persone operative. Il programma di controllo era costituito da più di 500.000 line di codice tutte scritte in assembler. L'intero progetto si stima sia costato dagli 8 ai 12 miliardi di dollari del 1964, molto di più dello stesso progetto Manhattan che sviluppò la prima bomba atomica.
In seguito, l'uso di calcolatori per applicazioni real-time, ebbe un notevole impulso sfruttando anche l'introduzione della multiprogrammazione. Sono stati citati precedentemente alcuni esempi come i sistemi operativi RT-11 e RSX-11 della Digital. Ciò che allora caratterizzava un sistema di elaborazione come sistema real-time era legato fondamentalmente alla velocità di esecuzione, e soprattutto alla velocità con cui la CPU rispondeva ai segnali di interruzione, alla possibilità di collegare vari dispositivi di I/O tramite convertitori analogico/digitale (A/D) e digitale/analogico (D/A), al supporto di un real-time clock per la generazione di un riferimento temporale interno e alla schedulazione dei processi applicativi tramite priorità assegnate agli stessi dal progettista dell'applicazione.
Successivamente, anche a causa del fatto che l'affidabilità dei sistemi dedicati ad applicazioni critiche real-time non risultava sempre adeguata, si sviluppò, soprattutto in ambito accademico, una notevole attività di ricerca con l'ottica di comprendere a fondo le peculiarità di questi sistemi. La semplice gestione dei processi applicativi su base prioritaria si dimostrò non ottimale non essendo facile trasformare un insieme di vincoli temporali in un insieme di priorità. Iniziò, quindi, uno studio approfondito sugli algoritmi di schedulazione più adatti a questa categoria di sistemi operativi [10]. Inoltre, fu chiarito che la principale caratteristica di un sistema real-time non doveva essere tanto la velocità di esecuzione quanto la prevedibilità in modo tale che, già in fase di progetto, fosse possibile garantire che ogni processo applicativo potesse rispettare la propria deadline in qualunque condizione di carico del sistema [14]. Da questo punto di vista, furono identificate due categorie di sistemi real-time. Alcuni, noti come sistemi "*hard real-time*", sono quelli dedicati ad applicazioni di controllo critiche, nel senso che la violazione di anche una sola deadline può compromettere il corretto funzionamento del sistema controllato, con conseguenze che possono essere anche catastrofiche per tale sistema. Si pensi ad esempio ai sistemi che controllano l'assetto di volo di un aereo. Altri sistemi, noti col nome di "*soft real-time*", hanno esigenze meno stringenti. In questi casi, la violazione di una deadline non compromette il corretto funzionamento dell'ambiente operativo ma ne riduce le prestazioni.
A partire dai primi anni '80 cominciarono ad uscire i primi sistemi operativi specificamente concepiti per applicazioni real-time. Questi

I Sistemi operativi "Embedded"

Dal punto di vista architetturale i sistemi embedded sono spesso caratterizzati dalla necessità di utilizzare dispositivi hardware dedicati quali sensori, attuatori, convertitori analogico/digitale e digitale/analogico e altri dispositivi necessari per far interagire il sistema con l'ambiente nel quale il sistema stesso è immerso. Dal punto di vista software c'è da notare che l'utente ha raramente accesso alla programmazione del sistema che è, viceversa, compito peculiare del progettista. I programmi applicativi devono soddisfare requisiti di tipo temporale (per esempio le deadline dei singoli processi applicativi), di dimensione (spesso i dispositivi hardware disponibili per questi sistemi hanno limitate funzionalità per cui è richiesta una ristretta occupazione di memoria sia in termini di codice che di dati). Più in generale sono proprietà tipiche dei sistemi embedded la limitazione di consumo (per garantire la massima autonomia nei numerosi casi in cui il sistema è alimentato a batteria, oltre alla necessità di minimizzare la dissipazione di calore tenendo conto che, spesso, tali sistemi sono inseriti in ambienti dove è scarsa la capacità di disperdere il calore prodotto), l'elevata affidabilità (legati alla necessità di garantire in certe applicazioni critiche un ben determinato valore del "MTTF" - *Mean-Time-To-Failure*), le prestazioni (se da un lato migliori prestazioni facilitano il rispetto dei requisiti temporali, dall'altro comportano consumi e costi più elevati) ed il costo (ciò è particolarmente vero in tutti quei casi in cui il sistema è prodotto in larga scala). Come è facilmente intuibile, non sempre è possibile garantire il rispetto di tutti questi requisiti contemporaneamente.

I primi esemplari di sistemi *embedded* sono usciti già verso la fine degli anni '50. Successivamente, il loro sviluppo ha avuto un notevole impulso negli anni '70 con la tecnologia VLSI (*VeryLargeScale Integration*) che ha reso possibile l'integrazione di un'intera CPU su di un singolo chip. Oggi la loro diffusione è talmente ampia da includere praticamente qualunque settore applicativo: dai telefoni cellulari, agli elettrodomestici, dalle centraline per autoveicoli ai sistemi biomedicali, dalle macchine per ufficio alle videocamere, dalle macchine utensili ai sistemi avionici e a numerose altre applicazioni che sarebbe impossibile elencare in maniera esaustiva.

possono essere grossolanamente classificati in due categorie, da un lato sistemi progettati ex novo per questo tipo di applicazioni, dall'altro sistemi che risultano estensioni real-time di sistemi operativi di tipo time-sharing.

Uno dei sistemi più diffusi appartenenti alla prima categoria è VXWorks [25], la cui prima versione fu prodotta dalla società Wind River nel 1980, un sistema Unix-like per applicazioni real-time che è stato successivamente adattato anche per sistemi embedded (vedi di seguito). Citiamo, di seguito e senza voler essere esaustivi, altri sistemi operativi di questa categoria, sviluppati sempre a partire dai primi anni '80:

- VRTX (*Versatile Real-Time Executive*) realizzato da James Ready and Colin Hunter fondatori della Hunter & Ready, oggi distribuito da Mentor Graphics [20] e che ha notevolmente influenzato lo sviluppo dello stesso VXWorks;
- ChorusOs, un sistema con architettura a microkernel realizzato in Francia alla INRIA nel 1980 [23];
- QNX Neutrino, anch'esso con architettura a microkernel, sviluppato nel 1982 e disponibile anche per architetture multielaboratore [4].

Alla seconda categoria appartengono molte estensioni real-time di sistemi operativi commerciali time-sharing. Fra queste, le più note sono le estensioni di Unix, Mach e Linux. Ciascuna di queste rappresenta, in realtà, non un singolo sistema operativo ma bensì un insieme di diverse estensioni dello stesso sistema base. Molte sono per esempio le versioni real-time di Unix a partire da RTU (*Real-Time Unix*), una delle prime implementazioni realizzata dalla società Massachusetts Computer Corporation nei primi anni '80. Analogamente, molte elevato è il numero delle iniziative che hanno come scopo la realizzazione di un sistema real-time a partire dal kernel di Linux.

Una categoria di sistemi operativi real-time che ha avuto un notevole sviluppo, soprattutto negli ultimi tempi a causa della loro ampia diffusione, è costituita dai sistemi operativi *embedded* (vedi riquadro) Con tale termine viene indicato ogni sistema di elaborazione che, a differenza di un sistema *general purpose*, è dedicato a svolgere uno specifico compito. Il termine *embedded* sta ad identificare che il sistema informatico è incorporato all'interno di un sistema più ampio, spesso col compito di svolgere funzioni di controllo. La caratteristica più saliente di questo tipo di sistemi è quindi quella di essere concepiti e realizzati per eseguire una sola applicazione (o un insieme limitato di applicazioni). I programmi applicativi quindi non cambiano durante la vita del sistema anche se, in certi casi, è necessario prevedere una loro evoluzione nel tempo. Nella maggior parte dei casi le applicazioni, come indicato precedentemente, sono relative al controllo di apparecchiature e sistemi di più ampie dimensioni. In questo senso i sistemi embedded appartengono alla più ampia categoria dei sistemi in tempo reale.

I sistemi embedded sono oggi presenti in una quantità enorme di applicazioni e il loro numero è sicuramente superiore di almeno un ordine di grandezza rispetto ai tradizionali personal computer.

Tenendo conto dei vincoli imposti dalle limitate risorse fisiche disponibili per queste tipologie di applicazioni, l'uso di un sistema operativo "*general purpose*", o anche di uno specifico si-

stema real-time, è spesso improponibile sia per le dimensioni di tali sistemi, eccessive rispetto alle risorse fisiche a disposizione, sia per l'assenza di specifiche funzionalità tipiche di molte applicazioni, come per esempio quelle legate alla elaborazione di segnali.
Per questo motivo, e a causa di una sempre maggiore diffusione dei sistemi embedded, negli ultimi tempi si è avuta un'ampia proliferazione di sistemi operativi dedicati a questo tipo di applicazioni che, con riferimento ai criteri seguiti nella loro realizzazione, possono essere classificati in due categorie. Alla prima, nota col termine di "*high-end embedded operating systems*" appartengono tutti quei sistemi che sono derivati da sistemi operativi *general purpose* da cui vengono eliminate tutte quelle componenti che risultano superflue, o comunque non necessarie per le applicazioni embedded (*down-size*), avvalendosi anche di strumenti sviluppati proprio con l'obiettivo di soddisfare i requisiti tipici dei sistemi embedded e destinati a rimpiazzare i più esigenti strumenti standard. Per esempio, tenendo conto dell'ampia diffusione di Linux, sono state realizzate molte versioni di Linux embedded. Analogamente esistono versioni embedded di Windows (Windows CE [17]).
All'altra categoria, nota come "*deeply embedded operating systems*", appartengono quei sistemi operativi, generalmente di dimensioni molto più limitate, realizzati *ad hoc* per particolari tipologie di applicazioni embedded. Come esempi citiamo Palm-OS (un sistema operativo sviluppato nel 1996 per computer palmari e smartphones da PalmSource Inc [22]), Symbian (un sistema operativo sviluppato a partire dal 1988 e concepito, in particolare, per il supporto di applicazioni per dispositivi mobili, tipicamente telefoni cellulari [24]), OSEK/VDX (uno standard per i sistemi software embedded relativi alle applicazioni dedicate alle centraline di controllo per autovetture [18]).
Nella più ampia categoria dei sistemi in tempo reale possiamo far rientrare anche quei sistemi dedicati ad applicazioni multimediali. Queste ultime sono caratterizzate da esigenze particolari: i file multimediali possono avere dimensioni ragguardevoli (un file video MPEG-1 di 100 min occupa circa 1,125 GB di memoria), richiedere una significativa larghezza di banda (dell'ordine delle centinaia di megabyte), il mantenimento di una velocità costante durante la trasmissione (almeno nel caso dello streaming in tempo reale).
Le applicazioni multimediali richiedono livelli di servizio, in genere, diversi rispetto a quelli delle applicazioni tradizionali. Particolarmente importanti sono i requisiti di tempo e di frequenza dato che la riproduzione di dati audio e video impone la trasmissione degli stessi con una cadenza precisa e a una determinata frequenza. Nel caso, ad esempio, di una riproduzione di un video MPEG-1 può essere necessaria una frequenza di 30 fotogrammi al secondo. Ciò significa che la trasmissione del fotogramma F_j successivo a F_i deve avvenire entro 3,34 centesimi di secondo.
I vincoli temporali e di sequenza vengono indicati come vincoli di qualità di servizio (QoS). In questo senso, un sistema operativo per applicazioni multimediali viene spesso caratterizzato come sistema soft real-time in quanto, se non viene rispettata una deadline più che compromettere l'integrità dell'applicazione ne viene degradato il QoS.
A differenza delle applicazioni tradizionali dove il QoS si basa su un servizio del tipo *best-effort* con l'ipotesi che le risorse disponibili siano in grado, anche nel peggior caso possibile, di fare fronte al carico di lavoro, le applicazioni multimediali devono fare riferimento a un livello predefinito di QoS che garantisca cioè i requisiti di frequenza e tempo richiesti.
È per raggiungere questo obiettivo che i sistemi operativi multimediali richiedono un meccanismo di scheduling dei processi di tipo real-time. Lo stesso problema si presenta anche per lo scheduling del disco. La soluzione cui in genere si ricorre si basa sull'utilizzo di due ben noti algoritmi, EDF (*Earliest Deadline First*) e SCAN. Il primo garantisce che vengano servite dapprima le richieste il cui tempo di scadenza è più vicino ed il secondo, raggruppando in modo opportuno le richieste più vincolanti, ottimizza il movimento della testina riducendo quindi il tempo di attesa.

3. VIRTUAL MACHINE MONITOR

Come si è visto nei paragrafi precedenti, a partire dagli anni '70 la diffusione dei moderni sistemi operativi multitasking, e contemporaneamente la riduzione del costo delle compo-

nenti hardware, ha fatto sì che i mainframe lasciassero progressivamente il posto ai minicomputer e ad un paradigma del tipo "un server per ogni applicazione". Questa tendenza ha però portato nei primi anni del 2000 ad una proliferazione di hardware all'interno delle sale server, con tanti server sottoutilizzati, considerato il costante aumento della loro potenza di calcolo, con grossi problemi legati agli spazi fisici, al condizionamento degli ambienti, agli elevati costi di alimentazione e di gestione. Si è posto quindi il problema della riorganizzazione delle sale server per rendere più efficiente l'uso delle risorse, semplificarne la gestione e aumentare la sicurezza (*consolidamento hardware ed applicativo*).
La soluzione che si sta sempre più diffondendo, si basa su una tecnologia detta delle *macchine virtuali*, introdotta per la prima volta da IBM negli anni 60 e consistente nel virtualizzare i server, facendoli operare cioè, ciascuno con la sua applicazione e con il suo sistema operativo (macchina virtuale) su un unico calcolatore fisico (host) ed affidando ad un nuovo sistema operativo, *Virtual Machine Monitor* (VMM) o *hypervisor*, il compito di partizionare le risorse hardware della macchina host tra le varie macchine virtuali e di assicurare la loro indipendenza. Il VMM si pone come mediatore unico nelle interazioni tra le macchine virtuali e le sottostanti componenti hardware, garantendo un forte isolamento tra loro e la stabilità complessiva del sistema.
Il sistema IBM più noto, realizzato con queste tecnologie fu il VM/370 che supportava macchine virtuali dotate di diversi sistemi operativi (IBM OS/360, DOS/360). La diffusione in quegli anni di questo tipo di sistemi portò anche alla progettazione di architetture hardware appositamente pensate per una loro efficiente realizzazione. Questo tipo di sistema fu poi abbandonato in favore, come si è detto, dell'utilizzo di server a minor costo dedicati a specifiche applicazioni al punto che le nuove architetture non hanno più fornito l'hardware necessario per una loro realizzazione efficiente.
La ripresa della tecnologia delle macchine virtuali è stata resa possibile dall'utilizzo come sistemi host di moderne architetture con proprietà di scalabilità, in grado cioè di consentire il dimensionamento dinamico della potenza di elaborazione e dello spazio disco al crescere della necessità. Inoltre sia Intel che AMD hanno recentemente introdotto modelli, Intel VT e AMD-V, dotati di supporto nativo alla virtualizzazione.
Le caratteristiche dei diversi tipi di VMM sono state descritte in un lavoro recentemente pubblicato su questa rivista [1], a cui si rimanda per ulteriori approfondimenti Si vuole qui segnalare, per concludere, che tra i vantaggi caratteristici di questi sistemi, oltre alla già citata possibilità di concentrare più server (con diversi sistemi operativi) su un'unica architettura HW per un utilizzo efficiente delle componenti hardware, va sottolineata la semplificazione delle operazioni quotidiane di presidio e gestione ordinaria poiché l'intera *Server Farm* è diventata controllabile da un'unica interfaccia – *Virtual Center* – attraverso cui è possibile accedere alle console dei server, monitorarne l'uso delle risorse, cambiarne a caldo la connettività di rete, attivare allarmi al superamento di soglie critiche, comandarne la migrazione a caldo nel caso occorra compiere manutenzioni sul sottostante hardware. È inoltre diventato semplice aggiornare le componenti hardware virtuali di una VM: si può incrementare la RAM o il numero di processori, aggiungere spazio disco o ulteriori schede di rete nello spazio di un riavvio.
Un altro vantaggio fondamentale dell'ambiente virtuale è legato alla creazione di un nuovo server: si possono infatti confezionare dei *template* di installazioni tipiche ed eseguire installazioni di nuovi server virtuali a partire da essi, con un notevole risparmio di tempo ed energie.

4. SISTEMI OPERATIVI PER ARCHITETTURE DISTRIBUITE

L'uso delle tecnologie di telecomunicazione dedicate all'interconnessione di calcolatori, iniziato negli anni '60, aprì una nuova linea di ricerca nell'ambito della disciplina informatica, quella delle reti di calcolatori, ricerca destinata ad avere una profonda influenza sull'evoluzione dei sistemi operativi.
Mentre lo sviluppo delle architetture dei calcolatori ha sempre avuto come obiettivo primario quello di migliorare le prestazioni di un singolo calcolatore (si pensi per esempio allo sviluppo delle architetture multieaboratore),

nel caso delle reti l'enfasi viene spostata soprattutto sulla comunicazione e sulla condivisione delle risorse di calcolo con l'obiettivo di realizzare applicazioni distribuite sui singoli nodi di una rete, sia essa una rete locale o una rete geograficamente distribuita.
Le motivazioni che sono state alla base dello sviluppo di ambienti atti a fornire il supporto per l'esecuzione di applicazioni distribuite sono molteplici:

- molte applicazioni sono naturalmente distribuite (si pensi per esempio a tutto il settore delle applicazioni bancarie);
- la possibilità di distribuire il carico computazionale di un'applicazione fra i vari nodi della rete;
- una maggiore affidabilità legata al fatto che un guasto su un singolo nodo della rete non compromette, se opportunamente gestito, l'integrità dell'intero sistema;
- un'implicita scalabilità del sistema complessivo;
- la possibilità di garantire a utenti diversi di un ambiente distribuito di condividere dati e risorse di elaborazione.

L'impatto che lo sviluppo dell'informatica distribuita ha avuto sui sistemi operativi è stato duplice: da un lato è progressivamente emersa la necessità di dotare i sistemi operativi esistenti di ulteriori componenti dedicati alla gestione delle interfacce di rete e all'implementazione dei protocolli di comunicazione con l'obiettivo di garantire le interazioni fra sistemi operativi, non necessariamente dello stesso tipo, residenti su diverse nodi di una rete; dall'altro è iniziata una nuova linea di ricerca tendente a progettare e realizzare una nuova tipologia di sistemi operativi specificamente concepiti per essere utilizzati sui calcolatori di una rete, in particolare una rete locale, con l'obiettivo di far vedere ai singoli utenti di ciascun nodo della rete un unico e coerente sistema di elaborazione.
I sistemi operativi che sono stati sviluppati secondo la prima delle due precedenti linee sono noti come *Network Operating Systems* (NOS) [16]. In questo caso, ogni utente opera su nodo della rete dotato di un proprio sistema operativo autosufficiente ma capace di comunicare con gli altri sistemi, anche se eterogenei, della rete. Ciò significa, per esempio, che da un nodo è possibile effettuare il login remoto su un diverso nodo della rete e, successivamente, eseguire comandi su un'altra macchina. Oppure ancora, copiare file da una macchina all'altra. Ciò che caratterizza questi sistemi è però il fatto che un utente ha sempre una chiara visione della rete e la conoscenza di dove si trovino le risorse di cui ha bisogno.
Un passo ulteriore verso una più spinta integrazione è rappresentato dalla realizzazione di file system distribuiti. In questi casi il file system viene gestito da uno o più nodi della rete (*File Server*) che accettano richieste provenienti da programmi residenti su altri nodi (*Client*). Un utente residente su una macchina client può accedere a file remoti come se fossero locali una volta che lo stesso abbia integrato la parte di file system remoto di suo interesse all'interno della gerarchia delle directory relativa al proprio sistema. Il *Network File System* (NFS) della Sun rappresenta un esempio di questa tipologia di sistemi [12].
I sistemi operativi sviluppati secondo l'altra linea sono noti come *Distributed Operating Systems* (DOS) [16]. In questo caso, su tutti i nodi della rete viene montata la stessa copia del sistema operativo che, come detto prima, ha il compito di presentare ad ogni utente un unico sistema di elaborazione, nascondendogli il fatto che alcune risorse sono locali al nodo su cui risiede, altre remote. Ogni utente accede, cioè, alle risorse del sistema complessivo nello stesso modo, indipendentemente dalla locazione fisica delle stesse. Amoeba rappresenta uno dei primi esempi di questo tipo di sistemi [11].

5. CONCLUSIONI

Lo scopo di questo articolo è stato quello di ripercorrere l'evoluzione storica dei sistemi operativi, analizzando le motivazioni che sono state alla base della loro introduzione e del loro sviluppo. È stata esaminata l'evoluzione dei sistemi a partire dai primi sistemi batch fino alle attuali realizzazioni nei diversi ambiti applicativi, cercando di mettere in evidenza anche come l'inserimento di nuove funzioni e le loro scelte di progetto abbiano fortemente risentito dei progressi di altre branche dell'intera disciplina informatica e al tempo stesso ne abbiano a loro volta influenzato l'evoluzione.
Nella prima parte del lavoro si sono ripercorse le tappe fondamentali che, partendo dai primi sistemi negli anni 50, hanno portato al-

la realizzazione, negli anni ottanta/novanta, degli attuali sistemi operativi multithreading che trovano in Unix, Linux e NT le più note e più utilizzate realizzazioni.
Nella seconda parte si è analizzato il mondo dei sistemi operativi sviluppati a partire dagli anni '80 per i personal computer, per i sistemi real-time e per le applicazioni embedded, chiudendo con un breve accenno ai sistemi operativi per macchine virtuali e ai sistemi operativi distribuiti.
Si può accennare, per concludere, come recentemente, nell'ambito della nuova tecnologia web 2.0 [26], si siano diffuse applicazioni che consentono di accedere in remoto dal proprio browser ad ambienti che simulano le funzionalità presenti sul desktop di un normale PC e consentono di condividere e contribuire ad incrementare informazioni che possono essere accedute da una comunità di utenti. Per rappresentare queste applicazioni viene talvolta usato il termine di sistemi operativi web (*webOS*), anche se la denominazione appare impropria e sostituibile con quella più corretta di webdesktop. Come esempi di applicazioni di questo tipo, oltre a quelle realizzate nel mondo da Google, si possono ricordare Desktoptwo [27], EyeOS [28], YouOS [29].

Bibliografia

[1] Boari M., Balboni S.: Tecniche di Virtualizzazione: Teoria e Pratica. *Mondo Digitale*, n. 1, Marzo 2007.

[2] Deitel H.M., Kogan M.S.: *The Design of OS/2*. Addison Wesley, 1992.

[3] Everett R.R., Swain, F.E: *Whirlwind I Computer Block Diagrams*. Report R-127, MIT Servomechanisms Laboratory, 1947. 1

[4] Hildebrand D.: *An Architectural Overview of QNX*. Proceedings of the Workshop on Microkernels and Other Kernel Architectures, 1992.

[5] Jacobs J.F.: *The SAGE Air Defense System: A Personal History*. MITRE Corporation, 1986.

[6] Kay A.: *A Personal Computer for Children of All Ages*. Proceedings of the ACM National Conference, August 1972, Boston.

[7] Ken W. C., Barry A. F., Shon O. S.: *MS-DOS 4.0*. Tecniche Nuove, Milano, 1989.

[8] Kildall G.: CP/M: *A Family of 8-and 16-bit Operating Systems*. Byte, June 1981.

[9] Lammers, ed.: *Programmers at Work*. Redmond, WA: Microsoft Press, 1986.

[10] Liu C.L., Leyland J.W.: Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment. *Journal of ACM*, Vol. 26,n. 1, 1973.

[11] Mullender S.J., Van Rossum G., Tanenbaum A.S, Van Renesse R., Van Staveren H.: A*meba: a Distributed Operatine System for the 1990s*. IEEE Computer, Vol. 23, n. 5, MAy 1990.

[12] Sandberg R.: *The Sun Network File System: Design, Implementatiklm and Experience*. Sun Microsystems, Mountain View, CA, 1987.

[13] Solomon D.: *The Windows NT kernel Architecture*. IEEE Computer, October 1998.

[14] Stankovic J.A.: *Misconceptions About Real-Time Computing: A serious Problem for Next-Generation Systems*. IEEE Computer, October 1988.

[15] Thacker C.P., McCreight E.M., Lampson B.W., Sproull R.F., Boggs D.R.: Alto: *A Personal Computer*. Tch. Rep. CSL-79-11, Palo Alto CA: Xerox Palo Alto Research Center, August 1979.

[16] Tanenbaum A.S.: *Distributed systems: principles and paradigms*. Prentice-Hall, 2002.

[17] http://msdn2.microsoft.com/en-us/embedded/aa731407.aspx

[18] http://portal.osek-vdx.org

[19] http://www.kernelthread.com/mac/osx/

[20] http://www.mentor.com//

[21] http://www.microsoft.com/windows/products/windowsxp/default.mspx.

[22] http://www.palm.com/us/

[23] http://www.sun.com/chorusos/

[24] http://www.symbian.com/

[25] http://www.windriver.com/

[26] http:// www.oreillynet.com

[27] https://desktoptwo.com

[28] http://eyeos.com

[29] http://www.youos.com

Paolo Ancilotti è professore ordinario di Sistemi Operativi presso la Scuola Superiore Sant'Anna di Pisa di cui è stato Direttore. È autore di numerosi articoli scientifici ed alcuni libri. Ha interessi di ricerca nel settore dei sistemi operativi, dei sistemi real-time e della programmazione concorrente.
E-mail: p.ancilotti@sssup.it

Maurelio Boari è professore ordinario di calcolatori elettronici presso la Facoltà di Ingegneria dell'Università di Bologna. È autore di numerosi articoli scientifici ed alcuni libri. Ha interessi di ricerca nel settore dei sistemi distribuiti, linguaggi di programmazione e sistemi operativi. È attualmente delegato del Rettore per l'informatica e le reti di Ateneo.
E-mail: maurelio.boari@unibo.it