A. Silberschatz - P. B. Galvin - G. Gagne

SISTEMI OPERATIVI

concetti ed esempi

ottava edizione

PEARSON

Abraham Silberschatz
Peter Baer Galvin
Greg Gagne

SISTEMI OPERATIVI

Concetti ed esempi

Ottava edizione

Edizione italiana a cura di Vincenzo Marra
Università degli Studi di Milano

PEARSON

Indice

Seconda parte Gestione dei processi

Capitolo 3 Processi

Capitolo 4 Thread

Capitolo 5 Scheduling della CPU

Capitolo 6 Sincronizzazione dei processi

Parte quarta Gestione della memoria secondaria

Capitolo 10 Interfaccia del file system

Capitolo 11 Realizzazione del file system

Capitolo 12 Memoria secondaria e terziaria

Parte sesta Sistemi distribuiti

Capitolo 16 Strutture dei sistemi distribuiti

Capitolo 17 File system distribuiti

Capitolo 18 Coordinazione distribuita

Parte settima Sistemi a orientamento specifico

Capitolo 19 Sistemi real-time

Prefazione all'ottava edizione italiana

Il testo di Silberschatz, Galvin e Gagne sui moderni sistemi operativi è uno dei manuali universitari di riferimento nell'insegnamento della materia. L'autorevolezza di Silberschatz si coniuga con le esperienze accademiche e professionali di Galvin e Gagne per dare vita a una trattazione aggiornata e approfondita dell'argomento.

L'ottava edizione mantiene inalterato l'impianto originale del testo, rilevatosi nel corso degli anni di grande efficacia didattica. D'altro canto, quasi tutti i capitoli sono stati rivisti e aggiornati per includere le principali novità emerse nel mondo dei sistemi operativi da quattro anni a questa parte. Il materiale sui sistemi operativi open-source, per esempio, è un'importante caratteristica della nuova edizione, come pure il maggiore spazio dedicato alle macchine virtuali. Gli Autori, inoltre, dimostrano ancora una volta particolare attenzione per la platea dei professionisti dell'informatica: i progetti pratici che corredano il testo offrono una concreta possibilità d'applicazione dei concetti teorici trattati.

Insomma, anche questa nuova fatica di Silberschatz e dei suoi coautori riesce nella difficile impresa di illuminare gli aspetti concettuali della materia senza metterne in ombra quelli tecnologici.

Vincenzo Marra
Dipartimento di Informatica e Comunicazione
Università degli Studi di Milano

Prefazione

Così come i sistemi operativi sono una parte essenziale dei sistemi di calcolo, un corso sui sistemi operativi è una parte essenziale di un percorso di studio d'informatica. Con i calcolatori ormai presenti praticamente in ogni campo, dai giochi ai più complessi e raffinati strumenti di pianificazione impiegati dagli enti governativi e dalle grandi multinazionali, la loro evoluzione ha ormai assunto un ritmo vertiginoso. D'altra parte, i concetti fondamentali restano abbastanza chiari; su di loro si fonda la trattazione svolta in questo libro.

Il testo è stato concepito e scritto per un corso introduttivo sui sistemi operativi, di cui fornisce una chiara descrizione dei concetti di base. Gli Autori si augurano che anche i professionisti giudichino il libro un'utile guida. Prerequisiti essenziali per la comprensione del testo da parte del lettore sono la familiarità con l'organizzazione di un generico calcolatore, la conoscenza di un linguaggio di programmazione ad alto livello, come il C o Java e delle principali strutture dati. Nel Capitolo 1 s'introducono le nozioni riguardanti l'architettura dei calcolatori necessarie alla comprensione dei sistemi operativi. Benché siano scritti prevalentemente in C e talvolta in Java, gli algoritmi analizzati nel testo sono facilmente comprensibili anche senza una conoscenza approfondita di questi linguaggi di programmazione.

I concetti sono esposti attraverso descrizioni intuitive, che evidenziano i risultati importanti sul piano teorico senza, tuttavia, ricorrere a dimostrazioni formali. Le note bibliografiche rinviano il lettore agli articoli di ricerca in cui sono stati presentati e dimostrati, per la prima volta, tali risultati, e contengono anche indicazioni utili a reperire materiale di approfondimento. In luogo di prove formali, abbiamo utilizzato – a corredo dei risultati – grafici ed esempi che ne illustrano la correttezza.

I concetti fondamentali e gli algoritmi trattati in questo testo spesso si basano su quelli impiegati nei sistemi operativi disponibili sul mercato. Si è in ogni modo cercato di presentare questi concetti e algoritmi in una forma generale, non legata a un particolare sistema operativo. Nel libro sono presenti molti esempi che riguardano i sistemi operativi più diffusi, oltre che i più innovativi, tra cui Solaris di Sun Microsystems, Linux, Microsoft Windows Vista, Windows 2000 e Windows XP, Apple Mac OS X.

Quando nel corso del libro si usa Windows XP come esempio di sistema operativo, si intendono Windows Vista, Windows XP e Windows 2000. Se ci si riferisce a una caratteristica propria di una specifica versione del sistema, la distinzione viene resa esplicita.

Organizzazione del libro

La struttura di questo testo rispecchia la lunga esperienza degli Autori, in qualità di docenti, nei rispettivi corsi di sistemi operativi; nel redigerlo si è tenuto conto delle valutazioni dei revisori del testo, nonché dei commenti inviati dai lettori delle precedenti edizioni. Inoltre, il testo recepisce nei contenuti quanto suggerito sull'insegnamento dei sistemi operativi da *Computing Curricula 2005*, edito dall'Unità Operativa Comune della IEEE Computing Society e dall'Association for Computing Machinery (ACM).

Come consiglio generale per il lettore, raccomandiamo di seguire i capitoli nell'ordine in cui sono presentati, poiché questo metodo garantisce la massima accuratezza nello studio dei sistemi operativi. È tuttavia possibile selezionare i capitoli (o i paragrafi) secondo una progressione diversa.

processi o utenti. Tale meccanismo deve fornire un metodo per definire i controlli e i vincoli ai quali gli utenti vanno sottoposti, e i mezzi per realizzarli. La sicurezza, invece, consiste nel proteggere sia le informazioni memorizzate all'interno del sistema (dati e codice) sia le risorse fisiche del sistema di calcolo da accessi non autorizzati, tentativi di alterazione o distruzione e dal verificarsi di incongruenze nel funzionamento.

- **Sistemi distribuiti**. I Capitoli dal 16 al 18 trattano i sistemi distribuiti. Con tale locuzione di solito ci si riferisce a un insieme di unità d'elaborazione che operano senza condivisione di memoria o di clock. Un sistema di questo tipo è in grado di mettere a disposizione dei propri utenti le varie risorse da esso controllate. La possibilità d'accesso a risorse condivise consente un incremento delle prestazioni del sistema, oltre che una maggiore affidabilità e disponibilità di dati e programmi. Attraverso l'uso di file system distribuiti, utenti, server e unità di memorizzazione si possono dislocare tra i vari siti del sistema distribuito. Tale tipo di sistema deve quindi provvedere alla realizzazione di diversi meccanismi per la sincronizzazione e la comunicazione capaci di gestire particolari questioni legate alle situazioni di stallo e alle situazioni critiche che non si incontrano nei sistemi centralizzati.
- **Sistemi a orientamento specifico**. I Capitoli 19 e 20 si occupano dei sistemi mirati a obiettivi specifici, compresi i sistemi real-time e multimediali. Tali sistemi, per la specificità dei propri requisiti, si differenziano da quelli a carattere generale, ovvero dall'argomento che permea il resto dell'opera. Per i sistemi real-time è importante non solo la "correttezza" dei risultati dell'elaborazione, ma anche il rispetto di scadenze temporali determinate a priori. I sistemi multimediali postulano le cosiddette garanzie di qualità-del-servizio, affinché i dati multimediali siano recapitati ai client entro tempi prestabiliti.
- **Casi di studio**. Nei Capitoli dal 21 al 23 del libro e nelle Appendici dalla A alla C (disponibili sul sito) si integrano i concetti esposti descrivendo alcuni sistemi operativi reali, tra i quali i sistemi Linux, Windows XP, FreeBSD, Mach e Windows 2000. I sistemi Linux e FreeBSD sono stati scelti poiché UNIX è un sistema operativo sufficientemente ridotto da poter essere studiato e compreso nei dettagli, pur non essendo un sistema operativo giocattolo. La maggior parte dei suoi algoritmi interni è stata scelta sulla base della semplicità e non per le prestazioni o la raffinatezza. Sia Linux sia FreeBSD sono tra l'altro presenti in tutti i dipartimenti d'informatica delle principali università, perciò un gran numero di studenti può accedervi liberamente. I sistemi Windows XP e Windows 2000 sono stati scelti perché offrono l'opportunità di studiare un sistema operativo moderno progettato e realizzato in modo radicalmente diverso da UNIX. Nel Capitolo 23 sono descritti brevemente alcuni tra i sistemi operativi che hanno maggiormente influenzato l'evoluzione di questo settore.

Ambienti di programmazione

Per delineare i concetti fondamentali della materia, in questo libro sono esaminati molti sistemi operativi realmente esistenti. Tuttavia si è dedicata particolare attenzione alla famiglia di sistemi operativi di Microsoft (Windows Vista, Windows 2000 e Windows XP) e alle varie versioni di UNIX (tra cui Solaris, BSD e Mac OS X). Si è scelto, altresì, di riservare ampio spazio al sistema operativo Linux, in particolare all'ultima versione del kernel (la 2.6 al momento della stesura del testo).

Contenuti del libro

Il testo è suddiviso in otto parti principali.

- **Generalità.** Nei Capitoli 1 e 2 si spiega che cosa sono i sistemi operativi, che cosa fanno e come sono *progettati* e *realizzati*. Questa spiegazione è offerta attraverso un'analisi dell'evoluzione dei concetti dei sistemi operativi, delle comuni caratteristiche e di quei servizi che un sistema operativo deve fornire agli utenti e agli amministratori del sistema. Essendo privi di riferimenti al funzionamento interno, questi capitoli sono consigliabili a chiunque voglia sapere che cos'è un sistema operativo, senza soffermarsi sui dettagli degli algoritmi che ne controllano il funzionamento.
- **Gestione dei processi.** Nei Capitoli dal 3 al 7 si descrivono i concetti di processo e concorrenza che costituiscono il fondamento dei moderni sistemi operativi. Per processo s'intende l'unità di lavoro di un sistema, che sarà caratterizzato da un insieme di processi eseguiti in modo concorrente, alcuni dei quali sono parte del sistema operativo (quelli incaricati di eseguire il codice di sistema), mentre i rimanenti sono i processi utenti (il cui scopo è, appunto, l'esecuzione del codice utente). In questi capitoli si affrontano i differenti metodi impiegati per lo scheduling e la sincronizzazione dei processi, la comunicazione tra processi e la gestione delle situazioni di stallo. Tra questi argomenti è compresa un'analisi dei thread e un esame dei temi relativi ai sistemi multicore.
- **Gestione della memoria.** Nei Capitoli 8 e 9 è trattata la gestione della memoria centrale durante l'esecuzione di un processo. Al fine di migliorare sia l'utilizzo della CPU sia la velocità di risposta ai propri utenti, un calcolatore deve essere in grado di mantenere contemporaneamente più processi in memoria. Esistono molti schemi per la gestione della memoria centrale; questi schemi riflettono diverse strategie di gestione della memoria e l'efficacia dei diversi algoritmi dipende dal particolare contesto in cui si applicano.
- **Gestione della memoria secondaria.** Nei Capitoli dal 10 al 13 si spiega come gli elaboratori moderni gestiscano il file system, la memoria di massa e l'I/O. Il file system fornisce il meccanismo per l'accesso ai dati e ai programmi che risiedono sui dischi, nonché per la loro memorizzazione in linea. Descriviamo gli algoritmi interni fondamentali e le strutture di gestione della memoria e forniamo una solida conoscenza pratica degli algoritmi utilizzati, analizzandone le proprietà, i vantaggi e gli svantaggi. La nostra trattazione comprenderà temi relativi alla memoria secondaria e terziaria. Dal momento che i dispositivi di I/O collegabili a un calcolatore sono del più vario genere, è necessario che il sistema operativo possa contare su funzionalità ampie e diversificate per le applicazioni, cosicché queste possano tenere sotto controllo i dispositivi in ogni loro aspetto. La trattazione dell'I/O mira ad approfondirne progettazione, interfacce, strutture e funzioni interne del sistema. Per molti aspetti, i dispositivi di I/O si guadagnano il primato di componenti più "macchinose" del calcolatore (tra quelle più importanti): analizzeremo, dunque, i problemi che derivano da questo collo di bottiglia nelle prestazioni.
- **Protezione e sicurezza.** I Capitoli 14 e 15 illustrano gli aspetti principali che riguardano protezione e sicurezza dei sistemi d'elaborazione. Tutti i processi di un sistema operativo devono essere reciprocamente protetti; a tale scopo esistono meccanismi capaci di garantire che solo i processi autorizzati dal sistema operativo possano impiegare le risorse del sistema, come file, segmenti di memoria, cpu e altre risorse. La protezione è il meccanismo attraverso il quale il sistema controlla l'accesso alle risorse da parte di programmi,

A scopo di esempio sono stati inoltre inseriti alcuni programmi in C e in Java, concepiti per i seguenti ambienti di programmazione.

- **Sistemi Windows.** Il più rilevante ambiente di programmazione per i sistemi Windows è l'API Win32 (*interfaccia per la programmazione di applicazioni*), che dispone di un insieme completo di funzioni per la gestione di processi, thread, memoria e periferiche. Per illustrare l'uso di API Win32 ci si è avvalsi di alcuni programmi in C. I programmi dimostrativi sono stati testati su piattaforme Windows Vista, Windows 2000 e Windows XP.
- POSIX. La sigla POSIX (ossia *interfaccia portabile del sistema operativo*) rappresenta una serie di standard creati essenzialmente per sistemi operativi della famiglia UNIX. Sebbene Windows Vista, Windows XP e Windows 2000 possano eseguire alcuni programmi POSIX, la nostra trattazione è prettamente incentrata sui sistemi UNIX e Linux. I sistemi compatibili con POSIX devono possedere lo standard di base (POSIX.1): è questo il caso di Linux, Solaris e Mac OS X. Esistono poi numerose estensioni degli standard di base; tra queste, l'estensione real-time (POSIX1.b) e quella per i thread (POSIX1.c, meglio nota come Pthreads). Vari programmi, scritti in C, fungono da esempio per chiarire non solo il funzionamento dell'interfaccia API di base, ma anche quello di Pthreads e dello standard per la programmazione real-time. Questi programmi dimostrativi sono stati testati sulle versioni 2.4 e 2.6 di Linux Debian, sul Mac OS X 10.5 e su Solaris 10 con l'ausilio del compilatore `gcc` 3.3 e 4.0.
- Java. Java è un linguaggio di programmazione largamente utilizzato, dotato di una ricca API, oltre che di funzionalità integrate per la creazione e la gestione dei thread. I programmi Java sono eseguibili da qualsiasi sistema operativo, purché vi sia installata una macchina virtuale Java (o JVM). Si illustrano vari concetti in relazione ai sistemi operativi e alle architetture di rete, grazie a programmi testati con la JVM 1.5.

Abbiamo scelto i tre suddetti ambienti di programmazione poiché li riteniamo i più adatti a rappresentare quei modelli che, tra i sistemi operativi, vantano la maggiore popolarità: Windows e UNIX/Linux, al pari dell'ambiente Java, ampiamente diffuso. I programmi dimostrativi, scritti prevalentemente in C, presuppongono una certa dimestichezza da parte dei lettori con tale linguaggio; i lettori con buona padronanza di Java, oltre che del linguaggio C, dovrebbero comprendere senza problemi la maggior parte dei programmi.

In alcuni casi – come per la creazione dei thread – ci serviamo di tutti e tre gli ambienti di programmazione per illustrare un dato concetto, invitando il lettore a confrontare le diverse soluzioni delle tre librerie, in riferimento al medesimo problema. In altri frangenti, soltanto una delle API è chiamata in causa per esemplificare un concetto. Per descrivere la memoria condivisa, per esempio, ricorriamo esclusivamente alla API di POSIX; la programmazione con le socket nell'ambito del protocollo TCP/IP è illustrata tramite la API di Java.

Ottava edizione

L'ottava edizione di questo testo è stata scritta considerando i numerosi suggerimenti ricevuti riguardo alle precedenti edizioni del testo, insieme con le osservazioni proprie degli Autori, sul mutevole campo dei sistemi operativi e delle reti di calcolatori. È stato riscritto molto nella maggior parte dei capitoli, aggiornando il vecchio materiale ed eliminando quello non più interessante.

È stata fatta una revisione sostanziale e sono state apportate modifiche nell'organizzazione di molti capitoli. In particolare, nel Capitolo 1 è stato dato ampio spazio ai sistemi operativi open-source. Sono stati aggiunti esercizi pratici per gli studenti, le cui soluzioni so-

no presenti su WileyPLUS, dove si trovano anche nuovi simulatori per esemplificare il funzionamento dei sistemi operativi.

Ecco un breve riepilogo delle principali modifiche apportate a ogni capitolo.

- Il **Capitolo 1, Introduzione**, è stato arricchito includendo CPU multicore, cluster e sistemi operativi open-source.
- Il **Capitolo 2, Strutture dei sistemi operativi**, contiene significativi aggiornamenti sulle macchine virtuali, oltre che sulle CPU multicore e sul debugging dei sistemi operativi.
- Il **Capitolo 3, Processi**, dà ampio spazio alle pipe come forma di comunicazione tra processi.
- Il **Capitolo 4, Thread**, amplia la trattazione sulla programmazione dei sistemi multicore.
- Il **Capitolo 5, Scheduling della CPU**, approfondisce le tematiche riguardanti lo scheduling delle macchine virtuali e le architetture multithread e multicore.
- Il **Capitolo 6, Sincronizzazione dei processi**, comprende nuovi punti riguardanti i lock per la mutua esclusione, l'inversione delle priorità e la memoria transazionale.
- Il **Capitolo 8, Memoria centrale**, include le architetture ad accesso non uniforme (NUMA).
- Il **Capitolo 9, Memoria virtuale**, arricchisce la trattazione di Solaris, includendo la gestione della memoria in Solaris 10.
- Il **Capitolo 10, Interfaccia del file system**, è stato aggiornato sulla scia delle tecnologie più recenti.
- Il **Capitolo 11, Realizzazione del file system**, presenta una descrizione esaustiva del file system ZFS di Sun e sviluppa le tematiche relative ai volumi e alle directory.
- Il **Capitolo 12, Memoria secondaria e terziaria**, contiene novità riguardanti l'ISCSI, i volumi e i pool ZFS.
- Il **Capitolo 13, Sistemi di I/O**, è stato integrato con tematiche riguardanti PCI-X PCI Express e l'HyperTransport.
- Il **Capitolo 16, Strutture dei sistemi distribuiti**, comprende ora anche la trattazione delle reti wireless 802.11.
- Il **Capitolo 21, Linux**, è stato aggiornato all'ultima versione del kernel Linux.
- Il **Capitolo 23, Prospettiva storica**, presenta informazioni sui primi calcolatori e sistemi operativi, come il TOPS-20, il CP/M, MS-DOS, Windows e il Mac OS originale.

Problemi di programmazione e progetti

Il testo è stato arricchito con diversi progetti ed esercizi di programmazione, che prevedono l'utilizzo delle API POSIX, Win32 e Java. Sono stati aggiunti più di 15 nuovi problemi di programmazione, che pongono in rilievo processi, thread, memoria condivisa, sincronizzazione dei processi e questioni attinenti alle reti. Abbiamo inoltre inserito vari progetti di programmazione, che presentano maggiore complessità rispetto agli ordinari esercizi di programmazione. Tra questi progetti vi è l'aggiunta di una chiamata di sistema al kernel di Linux, dell'utilizzo di pipe sia in Windows sia in Linux e di code di messaggi UNIX, la creazione di

Ringraziamenti

Questo testo deriva dalle precedenti edizioni, le prime tre delle quali sono state scritte insieme con James Peterson. Tra le altre persone che sono state d'aiuto per quanto riguarda le precedenti edizioni ci sono Hamid Arabnia, Rida Bazzi, Randy Bentson, David Black, Joseph Boykin, Jeff Brumfield, Gael Buckley, Roy Cambell, P.C. Capon, John Carpenter, Gil Carrick, Thomas Casavant, Ajoy Kumar Datta, Joe Deck, Sudarshan K. Dhall, Thomas Doeppner, Caleb Drake, M. Rasit Eskicioglu, Hans Flack, Robert Fowler, G. Scott Graham, Richard Guy, Max Hailparin, Rebecca Hartman, Wayne Hathaway, Christopher Haynes, Bruce Hillyer, Mark Holliday, Ahmed Kamel, Richard Kieburz, Carol Kroll, Morty Kwestel, Thomas LeBlanc, John Leggett, Jerrold Leichter, Ted Leung, Gary Lippman, Carolyn Miller, Michael Molloy, Yoichi Muraoka, Jim M. Ng, Banu Ozden, Ed Posnak, Boris Putanec, Charles Qualline, John Quarterman, Mike Reiter, Gustavo Rodriguez-Rivera, Carolyn J.C. Schauble, Thomas P. Skinner, Yannis Smaragdakis, Jesse St. Laurent, John Stankovich, Adam Stauffer, Steven Stepanek, Hal Stern, Louis Stevens, Pete Thomas, David Umbaugh, Steve Vinoski, Tommy Wagner, John Werth, James M. Westall, J.S. Weston e Yang Xiang.

Parti del Capitolo 12 sono state tratte da una relazione di [Hillyer e Silberschatz 1996]; parti del Capitolo 17 da una relazione di [Levy e Silberschatz 1990]; il Capitolo 21 da un manoscritto inedito di Stephen Tweedie; il Capitolo 22 da un manoscritto inedito di Dave Probert, Cliff Martin e Avi Silberschatz; l'Appendice C da un manoscritto inedito dello stesso Cliff Martin, che è stato anche d'aiuto nell'aggiornamento al sistema FreeBSD. Quest'edizione è stata arricchita di molti nuovi esercizi e relative soluzioni a cura di Arvind Krishnamurthy.

Mike Shapiro, Bryan Cantrill e Jim Mauro hanno risposto a diverse domande riguardanti Solaris. Bryan Cantrill della Sun Microsystems ci ha aiutato per la parte su ZFS. Steve Robbins dell'Università del Texas a San Antonio ha disegnato i simulatori presenti su WileyPLUS. Rees Newman del Westmister College ha analizzato i simulatori e valutato quanto fossero adatti a questo libro. Josh Deed e Rob Reynolds hanno contribuito alla parte su Microsoft .NET. John Trono, dell'Università di Saint Michael di Colchester, Vermont, ha contribuito al progetto sulle code di messaggi POSIX.

Marilyn Turnamian ha contribuito alla creazione di immagini e lucidi per le presentazioni. Mark Wogahn si è preoccupato del corretto funzionamento del software utilizzato per la stesura di questo volume (ad esempio Latex, macro, e font).

Il nostro publisher associato, Dan Sayre, ci ha guidato in maniera esperta nella preparazione di questo testo. La sua assistente, Carolyn Weisman, ha curato scrupolosamente molti dettagli. L'editor di produzione, Ken Santor, ci ha dato un valido supporto nella cura dei dettagli di produzione. Lauren Sapira e Cindy Johnson sono state di notevole aiuto nel rendere disponibile il materiale per WileyPLUS.

L'immagine di copertina è di Susan Cyr, e il disegno della copertina è di Howard Grossman. Beverly Peavler si è occupato del copy editing. La correzione delle bozze è a cura di Katrina Avery; l'indicizzazione è stata effettuata da WordCo, Inc.

Abraham Silberschatz, New Haven, CT, 2008
Peter Baer Galvin, Burlington, MA, 2008
Greg Gagne, Salt Lake City, UT, 2008

un'applicazione multithread e la condivisione della memoria quale soluzione al problema dei produttori e consumatori.

L'ottava edizione include anche una serie di simulatori di sistemi operativi progettati da Steven Robbins dell'Università del Texas di San Antonio. I simulatori sono finalizzati a modellare il comportamento di un sistema operativo nel momento in cui esegue diverse attività, quali lo scheduling della CPU e del disco, la creazione di processi e la comunicazione tra processi, la starvation e la traduzione di indirizzi. Tali simulatori sono scritti in Java e funzionano su tutti i computer dotati di Java 1.4.

Supplementi didattici

I supplementi didattici in lingua inglese, disponibili su http://pearson.it/silberschatz, comprendono:

- le tre appendici dedicate a Unix BSD, Mach e Windows 2000;
- i simulatori di sistemi operativi;
- tutti i sorgenti, sia in C sia in Java.

I professori italiani che adottano il testo potranno richiedere, compilando l'apposito modulo on-line che troveranno nello spazio docenti, le slide in italiano e il *Solutions Manual* (in inglese) che comprende anche le soluzioni agli esercizi di programmazione.

Mailing list

Come mezzo di comunicazione tra i lettori del volume abbiamo scelto il gestore di mailing list di mailman. Se desiderate fruire di questa possibilità, vi preghiamo di consultare il seguente URL, seguendo le istruzioni fornite per iscriversi:

http://mailman.cs.yale.edu/mailman/listinfo/os-book

Le mailing list di mailman offrono numerosi vantaggi, tra cui un archivio dei messaggi e la possibilità di scegliere fra varie opzioni di ricezione (per esempio, la cernita dei messaggi o la lettura esclusiva in rete). Per inviare un messaggio alla lista, mandate un messaggio e-mail a:

os-book@cs.yale.edu

A seconda del contenuto del messaggio, potremo rispondervi personalmente o inoltrare il messaggio a tutti coloro che fanno parte della lista; la lista è moderata: non riceverete messaggi impropri. Gli studenti che usano questo manuale come libro di testo si astengano dal chiedere le risposte agli esercizi attraverso la lista: non saranno fornite.

Suggerimenti

In questa nuova edizione si è cercato di eliminare tutti gli errori ma, come accade con i sistemi operativi, qualche oscuro baco probabilmente rimane. È benvenuta qualsiasi segnalazione riguardante errori o omissioni riscontrate nel testo.

Sono altrettanto benvenuti il suggerimento di miglioramenti e i contributi per nuovi esercizi. La corrispondenza deve essere inviata a os-book-authors@cs.yale.edu.

Abraham Silberschatz è titolare della cattedra Sidney J. Weinberg e direttore del Dipartimento di Informatica dell'Università di Yale. Prima di arrivare a Yale è stato vice presidente del Centro Ricerche Informatiche presso i Laboratori Bell. Prima ancora è stato titolare di una cattedra presso il Dipartimento di Informatica dell'Università del Texas ad Austin.

Il Professor Silberschatz è socio ACM e IEEE. Nel 2002 è stato insignito dell'Education Award Taylor L. Booth della IEEE, nel 1998 ha ricevuto l'Outstanding Educator Award Karl V. Karlstrom dell'ACM e nel 1997 il Contribution Award SIGMOD ACM. In riconoscimento dell'alto livello di innovazione e dell'eccellenza tecnica, è stato premiato dal presidente dei Laboratori Bell per tre differenti progetti: il Progetto QTM (1998), il Progetto DataBlitz (1999) e il Progetto NetInventory (2004).

Gli articoli del Professor Silberschatz sono comparsi su numerose pubblicazioni dell'ACM e dell'IEEE, oltre che in conferenze e altre riviste del settore. Silberschatz è uno degli autori del volume *Database System Concepts*. Ha inoltre scritto articoli come opinionista per alcuni giornali, tra cui il New York Times, il Boston Globe e l'Hartford Courant.

Peter Baer Galvin è chief technologist presso la Corporate Technologies (www.cptech.com), società che si occupa di integrazione e rivendita di servizi per l'informatica. In precedenza, è stato amministratore di sistema per il Dipartimento di Informatica della Brown University. Galvin scrive per la rivista *;login:* la rubrica sul sistema Sun. Ha inoltre scritto articoli per Bytes e per altre testate, oltre a rubriche per *SunWorld* e *SysAdmin*. In qualità di consulente e trainer ha tenuto in tutto il mondo conferenze e seminari sulla sicurezza e la gestione dei sistemi.

Greg Gagne dirige il Dipartimento di Informatica del Westminster College a Salt Lake City, dove insegna dal 1990. Oltre a essere docente di sistemi operativi, insegna reti, sistemi distribuiti e ingegneria del software. Tiene inoltre corsi di aggiornamento per insegnanti di informatica e professionisti.

Parte prima

Generalità

Un *sistema operativo* è un programma che agisce come intermediario tra l'utente e gli elementi fisici di un calcolatore. Lo scopo di un sistema operativo è fornire un ambiente nel quale un utente possa eseguire programmi in modo *conveniente* ed *efficiente.*

Un sistema operativo è il programma che gestisce il *sostrato materiale* di un calcolatore; deve fornire meccanismi idonei che assicurino il corretto funzionamento dell'elaboratore e lo preservino da eventuali interferenze improprie da parte dei programmi utenti.

La struttura interna dei sistemi operativi è soggetta a notevole variabilità ed è adattabile a criteri di organizzazione estremamente differenti. La progettazione di un nuovo sistema operativo è un compito impegnativo che richiede, in via preliminare, una chiara definizione degli obiettivi del sistema. In base a tali obiettivi si selezionano le possibili strategie e si individuano i relativi algoritmi.

I sistemi operativi sono programmi complessi e di vaste dimensioni, e vanno pertanto realizzati un pezzo per volta, per moduli. Ciascuno di loro dovrebbe costituire una parte del sistema chiaramente identificata: è necessario definire scrupolosamente sia le funzionalità sia i dati in ingresso e in uscita.