Testi del Syllabus

Resp. Did.	**BROGGI Alberto**	Matricola:	**004802**

Anno offerta:	**2014/2015**
Insegnamento:	**05613 - SISTEMI OPERATIVI**
Corso di studio:	**3050 - INGEGNERIA INFORMATICA, ELETTRONICA E DELLE TELECOMUNICAZIONI**
Anno regolamento:	**2013**
CFU:	**6**
Settore:	**ING-INF/05**
Tipo Attività:	**B - Caratterizzante**
Anno corso:	**2**
Periodo:	**Secondo Semestre**

Testi in italiano

Tipo testo	*Testo*
Lingua insegnamento	Italiano
Contenuti	Parte I (Teoria) Introduzione ai sistemi operativi. Sistemi batch, time-sharing, spooling. Parallelizzazione di elaborazione e I/O. Il sistema di interruzione: interruzione da dispositivo e da timer. Gestione delle interruzioni. Modello di un semplice sistema operativo e tecniche di incremento delle prestazioni. Multiprogrammazione e modello a processi. Sistema di protezione. Modi di funzionamento della CPU. Chiamate di sistema. Gestione dell'I/O. Struttura a livelli del sistema operativo. Virtualizzazione. Concetto di processo. Stato del processo. Descrittore del processo. Processi concorrenti. Modello di interazione a memoria condivisa. Mutua esclusione e sezioni critiche. Semafori e primitive di sincronizzazione. Cooperazione e competizione tra processi mediante semafori. Il modello di interazione a scambio di messaggi e le primitive send/receive. Concetto di Deadlock e tecniche di gestione. Algoritmi di scheduling della CPU. Parte II (UNIX) Introduzione a UNIX e Linux. Struttura del file system di UNIX. Diritti e meccanismi di protezione. Principali comandi di sistema. Redirezione e piping di comandi. Interpreti comandi. Modalità di esecuzione foreground/background. Organizzazione fisica del file system. L'immagine in memoria di un processo UNIX. Sviluppo di programmi in UNIX/LINUX. Primitive per la gestione dei file e dell'I/O. Primitive per la gestione dei processi. Creazione, esecuzione e terminazione. Sincronizzazione e comunicazione tra processi: segnali, pipe/fifo e socket. Attività di esercitazione (massimo 240 caratteri) Esercitazioni in laboratorio informatica di base (Linux Suse 11 virtualizzato con Virtualbox) su interazione utente e programmazione di sistema Linux.
Testi di riferimento	Sistemi operativi - Concetti ed esempi - 8/ed, A. Silberschatz, P. B. Galvin, G. Gagne Pearson Education 2009, ISBN 9788871925691 (oppure) Sistemi Operativi 2/ed, P. Ancilotti, M. Boari, A. Ciampolini, G. Lipari McGraw Hill, ISBN: 9788838664328, GaPiL -- Guida alla Programmazione in Linux (http://gapil.truelite.it)
Obiettivi formativi	L’obiettivo del corso è fornire allo studente la capacità di comprendere come i sistemi operativi gestiscano e controllino le risorse del sistema di calcolo con obbiettivi di efficienza e facilità d’uso, e in particolare : - il ruolo del sistema operativo come intermediario tra le applicazioni e l’ hardware - la necessità di sovrapporre attività di CPU e di I/O per aumentare l’ efficienza - il concetto di processo e i principali modelli di interazione tra processi - i criteri e gli algoritmi di scheduling della CPU per sistemi interattivi - le nozioni di base per l’utilizzo di UNIX/Linux e per la gestione dei comandi - le principali chiamate di sistema offerte da UNIX/Linux alle applicazioni Le capacità di applicare le conoscenze e comprensione elencate risultano essere in particolare: - analizzare e valutare le caratteristiche dei sistemi operativi general-purpose

Tipo testo	*Testo*
	- analizzare semplici interazioni tra processi in ambiente globale e individuare forme di sincronizzazione attraverso semafori - valutare le prestazioni dei principali algoritmi di scheduling della CPU per sistemi interattivi con un workload prefissato - analizzare interazioni tra processi UNIX, individuare gli strumenti di comunicazione interprocesso e le primitive di sistema necessari, e realizzare programmi in C che li utilizzano
Prerequisiti	Fondamenti di informatica + Laboratorio di programmazione Fondamenti di programmazione
Metodi didattici	Lezioni frontali su teoria e UNIX/Linux. Esercitazioni in laboratorio informatica di base (Linux Suse 11 virtualizzato con Virtualbox) su interazione utente e programmazione di sistema Linux.
Altre informazioni	Sito del corso su http://lea.unipr.it
Modalità di verifica dell'apprendimento	Una prova scritta con tre domande aperte sulla parte di teoria (Parte I) oppure prova in itinere (20 aprile circa). Assegnamento opzionale di programmazione sulla gestione di processi UNIX. Una prova pratica al calcolatore costituita da un esercizio in C/C++ sulla interazione tra processi UNIX (Parte II). Gli studenti possono sostenere il giorno dell'esame una sola o entrambe le prove. I risultati parziali conseguiti rimangono validi per tutto l'anno accademico (sessione di gennaio/febbraio quindi compresa), ed e' possibile ripetere, al più una volta, una prova per migliorare il voto conseguito. In ogni caso si mantiene il voto migliore tra quelli conseguiti per ciascun tipo di prova (teoria o UNIX). Per calcolare il voto finale, sarà eseguita una media pesata dei risultati ottenuti nei due tipi di prova (con pesi 0.6 per la prova UNIX e 0.4 per la prova di teoria).
Programma esteso	Teoria Introduzione al corso. Programma e modalità di esame. Introduzione a Linux e alle possibilità di installazione. S.O come gestore di risorse. Attività del OS per gestione risorse. Tipi e utenti di SO. SO proprietari e standard. Introduzione a evoluzione storica dei sistemi di calcolo e dei OS Sistemi Batch. Time Sharing. Spooling. Gestione I/O Gestione I/O a polling. Sovrapposizione di attività di I/O. Gestione delle interruzioni. (4 ore) Multiprogrammazione. Concetto e stati di un processo. Gestione processi. PCB. Commutazione tra processi. Sistema di protezione. Gestione del cambio di contesto. System calls . Struttura di un OS con esempi (UNIX, MSDOS, Windows NT/7). Concetti e tecniche di virtualizzazione. Nucleo del S.O (5 ore) Modelli di interazione tra processi. Modello ambiente globale. Strumenti per la programmazione concorrente. Cenni sulle thread . Interazione tra processi. Esempi di interferenza. Mutua esclusione. Sezioni critiche. Semafori. Primitive wait e signal. Atomicità semafori e di wait/signal. Semafori per interazione produttore/consumatori. Modello ad ambiente locale. Classificazione designazione/sincronizzazione. Modalità di designazione diretta/indiretta. Sincronizzazione send e receive. Chiamate a procedura remota. (5 ore) Deadlock e tecniche di gestione (2 ore) Livelli di scheduling. Criteri per la valutazione degli algoritmi. Algoritmi di scheduling della CPU (FCFS, SJF, Priorità, Round-robin). Metodi per la valutazione. Scheduling UNIX. Cenni allo scheduling real.time. (3 ore) UNIX Introduzione a UNIX. File system. Diritti di accesso a file/direttori. Principali comandi UNIX. Filtri. Redirezione I/O e piping. Esecuzione comandi da shell. Modalità di esecuzione dei comandi. Metacaratteri .

Tipo testo	*Testo*
	Controllo espansione riga di comando. Script: sintassi ed esempi. Esercitazione UNIX in laboratorio su file system e comandi. (4 ore) Strumenti di sviluppo UNIX. Immagine di un processo. Argomenti di invocazione e ambiente Primitive per la gestione dell'I/O (open, close, read, write, lseek, etc.). Esempi. Esercitazione UNIX in laboratorio su primitive di I/O. (4 ore) Primitive per la gestione dei processi (fork, wait, exec). Esempi. Esercitazione UNIX in laboratorio su primitive di gestione processi. Presentazione dell'assegnamento opzionale sulla gestione dei processi. (4 ore) Gestione segnali UNIX. Primitive per la gestione inaffidabile dei segnali. Problemi della gestione inaffidabile. Primitive per la gestione affidabile dei segnali. Esempi. Comunicazione via pipe. Esempi. Esercizi su segnali e pipe. Comunicazione via FIFO. Esercitazione UNIX in laboratorio su pipe e segnali (5 ore) Comunicazione via socket. Tipi di socket. Cenni a TCP e UDP. Primitive per la gestione delle socket. Socket connesse. Server concorrente. Socket datagram. Esempi. Primitiva select. Esercitazione UNIX in laboratorio su gestione socket (4 ore) Esercizi d'esame. (2 ore).

Testi in inglese

Tipo testo	*Testo*
Lingua insegnamento	Italian
Contenuti	Part I Introduction to Operating Systems. Batch, time-sharing and spooling systems. Processing and I/O parallelization. The interrupt systems: device and timer interrupts. Interrupt handling. Modelling of a simple OS and technique for increasing the performance. Multitasking e process model. Protection system. CPU operating modes. Systems calls. I/O handling. Layered structure of OSs. Virtualization. Concept of process. Process state and descriptor. Concurrent processes. Cooperation and competition among processes. Shared memory interaction. Mutual exclusion and critical sections. Semaphores e synchronization primitives. Message passing interaction. Send/receive primitives, designation and synchronization. Deadlock management. CPU scheduling and main algorithms. Part II Introduction to UNIX and LINUX. UNIX File System. File access rights. Main system commands. I/O redirection and piping. Command shell. Foreground and background command execution. Physical organization of the file system. Process image. Program development in UNIX. File and I/O primitives. System calls for process management: creation, execution and exit. Interprocess communication facilities: unreliable and reliable signals, pipes, FIFOs and sockets. Client-server examples.
Testi di riferimento	Operating Systems Concepts - 8/ed, A. Silberschatz, P. B. Galvin, G. Gagne Wiley 2008,ISBN-13: 978-0470128725 Advanced Linux Programming, http://www.advancedlinuxprogramming.com
Obiettivi formativi	The aim of the course is to provide students with the ability to understand how the operating systems manages and controls computing system resources with the objectives of efficiency and ease of use, and in particular: - the role of the operating system as an intermediary between applications and hardware - the need to overlap CPU and I/O activities to increase system efficiency - the concept of process and the main models of process interaction - the criteria and CPU scheduling algorithms for interactive systems - the basic use of UNIX / Linux OSes and command usage - the main system calls offered by UNIX / Linux to applications The abilities to apply the knowledge and understanding listed above are in particular: - the analysis and evaluation of the characteristics of general-purpose operating systems - the analysis of a simple interaction among processes in the global environment and the identification of simple forms of synchronization using semaphores - the performance evaluation (on a given workload) of the main CPU scheduling algorithms for interactive systems

Tipo testo	*Testo*
	- the analysis of the interactions among UNIX processes, the identification of required interprocess communication primitives and system calls, and the implementation of C programs that take advantage of them
Prerequisiti	Programming course
Metodi didattici	Class lectures on OS theory and UNIX/Linux usage and programming. Lab practice on Linux PCs at the “Laboratorio Informatica di base” lab. Exercises on UNIX/LINUX commands and shells. Programming exercises on process interaction under UNIX/LINUX.
Altre informazioni	Course website on http://lea.unipr.it
Modalità di verifica dell'apprendimento	Written examination on the theory of Operating Systems (Part I of the course) or midterm exam. Computer exercise on UNIX/LINUX system programming. Optional programming assignment on Linux process management. Final grade is weighted on theory (40%) and UNIX (60%) grades.
Programma esteso	Theory Course introduction. Course contents and student evaluation methods. Introduction to the Linux OS and installation methods. The OS as a resource manager. OS activities for resource management. OS types and users. Proprietary and standard OSes. Brief history of computing systems and OS evolution. Batch and Time-Sharing systems.. Spooling. I/O management. Polling. Overlap between I/O activities. Interrupt management. (4 hours) Multiprogramming. Concept of process and possible states. Process management. PCB.. Process switching. OS protection and security. Context switching. System calls. OS structure with examples (UNIX, MS-DOS, Windows NT / 7). Virtualization concepts and techniques. OS kernel (5 hours) Models of process interaction. Global environment model. Tools for concurrent programming. Some background on threads. Type of interaction among processes. Examples of interference. Mutual exclusion. Critical sections. Semaphores. Wait and signal primitives. Atomicity of wait / signal. Use of semaphores for producer / consumer interaction. Local environment model. Classification of designation and synchronization alternatives. Direct and indirect designation. Synchronization for send and receive. Remote procedure calls. (5 hours) Deadlock and management techniques (2 hours) Levels of scheduling. Criteria for algorithm evaluation. CPU scheduling algorithms (FCFS, SJF, Priority, Round-robin). Methods for evaluation. UNIX scheduling. Overview of real-time scheduling. (3 hours) UNIX An introduction to UNIX. File system. Access rights to files / directories. Basic UNIX commands. Filters. I/O redirection and command piping. Running shell commands. Execution mode of commands. Wildcards. Expansion control in the command line. Scripts: syntax and examples. Laboratory practice on UNIX file system and commands. (4 hours) UNIX development tools. Process image. Command line arguments and environment. System calls for I/O (open, close, read, write, lseek, etc..). Laboratory practice on I/O. primitives (4 hours) System calls for process management (fork, wait, exec). Examples. Laboratory practice on primitives for process management. Presentation of the optional assignment on process management. (4 hours) UNIX signals. Systems calls for the unreliable signals. Issues with unreliable signals. Primitives for reliable signals. Examples. Inter-process communication with pipes. Examples. Exercises on signals and pipes. Communication with FIFOs. Laboratory practice on signals and pipe (5

Tipo testo	*Testo*
	hours) Socket communication. Socket types. Notes on TCP and UDP protocols. Socket systems calls. Connected sockets. Concurrent servers. Datagram sockets. Examples. The select system call. Laboratory practice on socket communication (4 hours) Sample final exams. (2 hours).