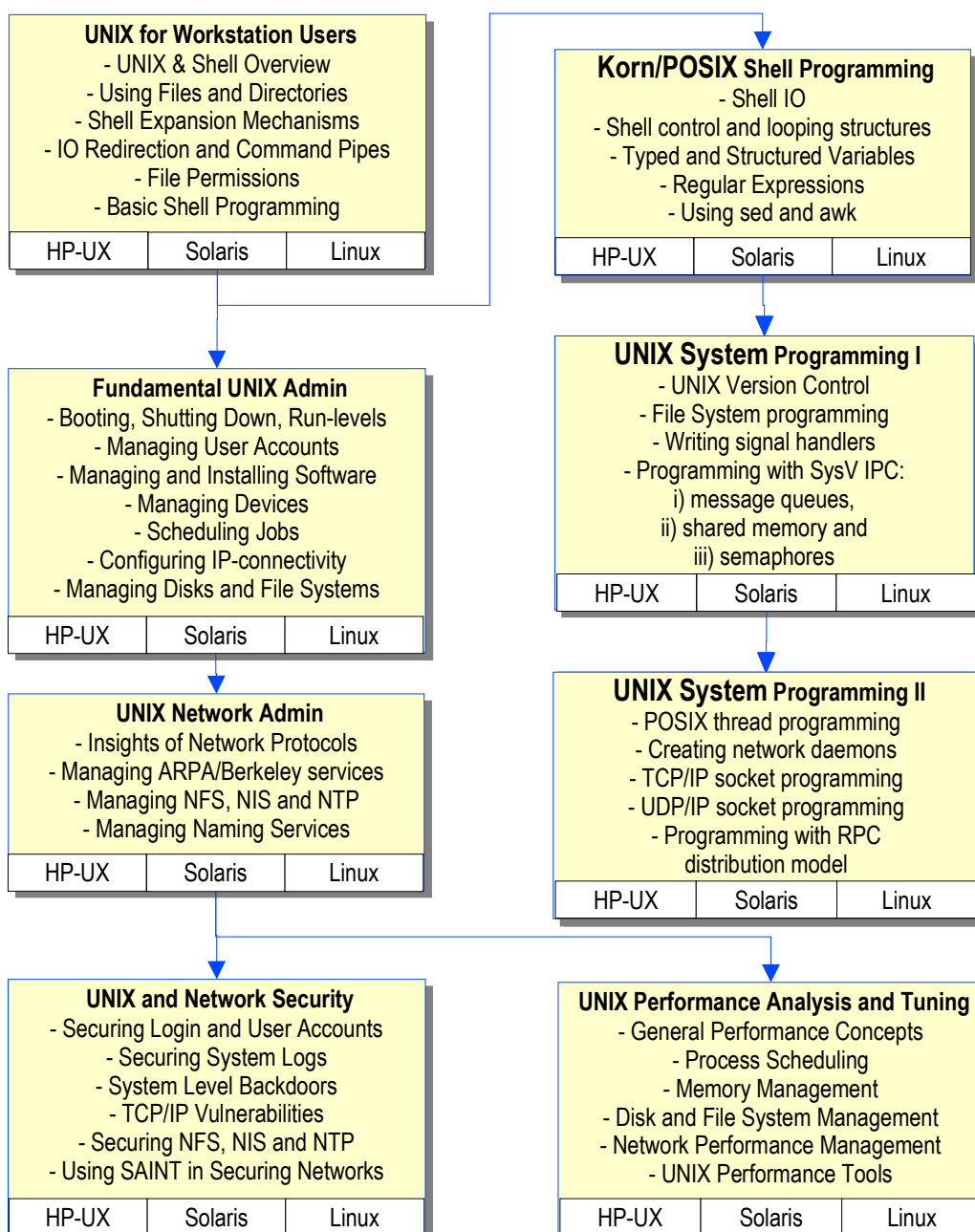


ReSolution produces and delivers a highly customized set of UNIX seminars for demanding IT organizations. Courses are designed to be held on-site and are always tailored, case by case, to meet the exact needs of the seminar in question.

The seminars are designed to serve organizations operating in multi-UNIX environments: the core of learning experience is in-depth discussion of each topic, putting emphasis on conceptual structures and general UNIX design. On top of that the practical issues of the most widely used UNIX flavors, HP-UX, Solaris and Linux, are covered.

As the end result students get a solid knowledge of fundamental and advanced UNIX topics, as well as a clear and detailed picture of differences of leading UNIX flavors – a highly valued dimension in management of multi-UNIX platforms.



3 days, 50% lecture 50% lab exercises
target audience: users with no or little UNIX experience

The course is an intensive coverage of UNIX fundamentals from shell user's point of view. It covers all powerful features of standard UNIX shells, including using shell expansion capabilities, input-output redirection and command pipes. It also includes the core concepts of shell programming. The course concentrates on Korn and POSIX shells, making it 100% compatible with all major UNIX flavors, including HP-UX, Solaris and Linux.

The course assumes some working experience with graphical user interfaces such as Microsoft Windows™ – the basic usage of UNIX graphical user interfaces such as Gnome, KDE and CDE is not included.

The goal is to give broad capabilities to use UNIX services effectively as an application power user and as a system administrator.

MODULE 1: UNIX AND SHELL OVERVIEW

- UNIX history
- Shell basics, different shells available
- Shell initialization files

MODULE 2: NAVIGATING DIRECTORIES

- Relative and absolute paths
- Directory navigation (cd, pwd, special directories)
- Managing directories (ls, mkdir, rmdir)
- Finding files based on attributes (find)

MODULE 3: MANIPULATING FILES

- Displaying files (cat, more, head, tail)
- Managing File Instances (cp, mv, rm)
- Basic regular expressions
- Searching regular expressions (grep)

MODULE 4: SHELL POWER USAGE

- Shell expansion concepts and meta characters
- File name expansion
- Input-output redirection
- Command pipes
- Shell quoting mechanisms

MODULE 5: FILE PERMISSIONS

- UNIX user and group accounts
- User and group id's of a process
- Switching user id (su)
- Standard file and directory permissions (read, write, execute)
- Controlling standard permissions (chmod)
- Setting default permissions (umask)
- Changing file ownership (chown, chgrp)

MODULE 6: INTRODUCTION TO KORN/POSIX SHELL PROGRAMMING

- Script execution and debugging
- Getting input (environment variables, positional parameters, read command)

MODULE 7: KORN/POSIX SHELL BRANCHING STRUCTURES

- The concept of a process return value
- Evaluating numerical, string and file attribute expressions (test)
- Conditional branching (if-then-else-fi)
- Multibranch testing (case-in-esac)

MODULE 8: KORN/POSIX SHELL LOOPING STRUCTURES

- Arithmetic expressions (expr, let)
- Predefined iteration (for-do-done)
- Conditional looping (while-do-done, until-do-done)
- Breaking out from a loop (break, continue)

APPENDIX A: VI-EDITOR SURVIVAL KIT

APPENDIX B: EMACS-EDITOR SURVIVAL KIT

3 days, 50% lecture 50% lab exercises
target audience: new UNIX system administrators

The course covers the most important topics of configuring and maintaining standalone UNIX systems, including creating and managing user accounts, configuring network connectivity and installing pre-packaged application software.

The course puts a heavy emphasis on proper disk and file system management – an area which has a huge impact on system integrity and performance. The course contains in-depth discussion of general UNIX administration principles and, on top of that, detailed modules for HP-UX, Solaris and Linux command and services.

The goal is to give a solid basis for administrating standalone HP-UX, Solaris and Linux systems, as well a clear picture of their differences for multi-UNIX administrators.

MODULE 1: BOOTING AND SHUTTING DOWN A UNIX SYSTEM

- Booting sequence and boot area data structures
- Switching UNIX run-levels
- Controlling system initialization scripts
- Adding own services to standard boot up and shut down

MODULE 2: MANAGING USER ACCOUNTS

- User account data bases (/etc/passwd, /etc/group, /etc/shadow)
- Adding, modifying and deleting users from command line
- Managing group accounts
- Managing user passwords
- Securing UNIX user accounts

MODULE 3: MANAGING SOFTWARE

- HP-UX: Managing software with SD-UX
- Solaris: Managing software with admintool and SVR4 package commands
- Linux: Managing rpm packages

MODULE 4: CONTROLLING DEVICES

- Examining device configuration in different hardware and UNIX platforms
- The concept of a device file
- Creating, controlling and using device files

MODULE 5: CONFIGURING IP-CONNECTIVITY

- Standard network configuration files (/etc/host, /etc/services, /etc/inetd.conf)
- Configuring networking parameters (network card, hostname, naming service switch)

MODULE 6: SCHEDULING JOBS

- Scheduling service (at, batch, cron, anacron)
- Controlling access to cron daemon
- Crontab format and command
- Cron file and directory structures

MODULE 7: MANAGING LOGICAL DISKS

- HP-UX: Managing Logical Volumes (LVM)
- Solaris: Managing meta devices with Solstice DiskSuite
- Linux: Managing Logical Volumes (LVM)

MODULE 8: CREATING AND USING UNIX FILE SYSTEMS

- Kernel structures for file systems
- File system mounting and unmounting
- HP-UX: Creating HFS and JFS file systems
- Solaris: Creating journaling UFS file systems
- Linux: Creating ext2 and ext3 file systems
- Mounting file systems at boot time

MODULE 9: MANAGING FILE SYSTEMS

- Viewing file system utilization level (df, bdf)
- Viewing file and directory utilization levels (du)
- Trimming files: log files, old temporary files, core files
- Correcting file system corruption (fsck)

3 days, 40% lecture 60% lab exercises
target audience: UNIX system administrators

The course covers the configuration of the most widely used standard network services in UNIX environments, including the configuration of NFS and DNS servers. The course puts a special emphasis on general concepts and insights of the topics, giving needed knowledge to configure services in all major UNIX platforms, including HP-UX, Solaris and Linux.

The course also contains a detailed discussion of UNIX networking protocols, routing dynamics and commands, building a solid basis for basic network troubleshooting.

The goal is to give capabilities to manage networks of UNIX machines in multi-platform environment.

MODULE 1: TCP/IP NETWORKING CONCEPTS

- OSI networking model v. TCP/IP addressing schemes
- Essential protocol insights (ip, arp, tcp, udp, rpc, application protocols)
- Socket connections and routing
- Configuring network connectivity and static routing
- UNIX tools for network troubleshooting

MODULE 2: MANAGING ARPA/BERKELEY SERVICES

- ARPA and Berkeley network service families
- Configuring service data base (/etc/services)
- Configuring inetd daemon (/etc/inetd.conf)
- HP-UX: Configuring inetd access (/var/adm/inetd.sec)
- Trusted host configurations (/etc/hosts.equiv, \$HOME/.rhosts)

MODULE 3: MANAGING NETWORK FILE SYSTEM (NFS)

- NFS concepts, protocols and daemons
- Configuring NFS server and share options
- Starting NFS server services
- Configuring NFS client
- NFS client mount options
- Starting NFS client services

MODULE 4: MANAGING NETWORK INFORMATION SYSTEM (NIS)

- NIS concepts, protocols and daemons
- Configuring NIS master server for a domain
- Configuring NIS slave server for a domain
- Configuring NIS client
- Controlling access to a NIS domain
- Overview of NIS++ as an alternative for NIS

MODULE 5: MANAGING NETWORK TIME PROTOCOL

- NTP concepts, protocols and daemons
- Configuring NTP server to distribute time from a selected time source (radio clock, another server, internal clock etc.)
- Controlling NTP server stratum levels and time broadcasting
- Configuring NTP client to receive time from a set of NTP servers
- NTP security issues

MODULE 6: MANAGING NAMING SERVICES

- DNS concepts and resolver routines
- DNS domains and zones
- Configuring DNS servers (bind 9 implementation for primary, secondary and cache-only servers)
- Record structures in DNS data bases
- Configuring a DNS client system
- Using nslookup to verify a DNS server functionality
- Local naming service (/etc/hosts)
- Configuring name service switch to select desired naming service (/etc/nsswitch.conf)

3 days, 50% lecture 50% lab exercises
target audience: experienced UNIX system administrators

The course describes the various known security loopholes in a typical UNIX workstation and server, such as creating backdoors, breaking into a file system to gain unauthorized privileges and vulnerabilities caused by common network configurations and services.

The course will give skills to plug these loopholes and to monitor the UNIX system and network configurations effectively in order to locate and stop unwanted hacker activities.

The goal is to give experienced administrators the skills needed in plugging and preventing security threats in networked UNIX systems.

MODULE 1: SECURITY CONCEPTS

- Computer security areas
- UNIX security hot spots
- Network security hotspots
- Requirements for "Orange Book" C2 security level

MODULE 2: GATHERING INFORMATION FROM UNIX SYSTEMS

- Commands and services that provide system information (login banners, finger, rwho, rusers)
- Standard network port scanning issues
- Sendmail issues
- NFS issues

MODULE 3: GAINING ACCESS TO UNIX SYSTEMS

- Getting to a login prompt
- System console, terminal and modem line issues
- Inetd network daemon issues
- X-windows issues

MODULE 4: GAINING PRIVILEGES IN UNIX SYSTEMS

- User account and password structures (/etc/passwd, /etc/shadow), password algorithms
- Cracking UNIX passwords
- Root account issues
- SUID privilege issues
- Configuring sudo for secure root privileges

MODULE 5: MONITORING AND COVERING UNIX SYSTEM ACTIVITIES

- Monitoring log files and login attempts
- Monitoring shutdown and cron activities
- Covering and hiding unauthorized activities
- Logging network services
- Securing system logging (syslogd)

MODULE 6: SECURING UNIX FILE SYSTEMS

- Basic file permissions (read, write, execute)
- Special file permissions (SUID, SGID, sticky bit)
- Securing and monitoring SUID executables
- Securing world writable directories
- Monitoring file systems with Tripwire

MODULE 7: CLOSING SYSTEM LEVEL BACKDOORS

- System level backdoor types (write access to system and home directories, device files, system startup scripts etc.)
- Available UNIX and open source tools for monitoring backdoor threats

MODULE 8: NETWORK SERVICE VULNERABILITIES

- Inetd network daemon issues
- FTP Issues
- NFS, NIS and RPC issues
- NTP issues
- Trusted host issues
- Using tcpwrapper to control network service access

MODULE 9: MONITORING NETWORK VULNERABILITIES WITH SAINT

- SAINT Overview
- Installing and configuring open source version of SAINT
- Running SAINT against a set of ip-addresses
- Analyzing SAINT data reports
- SAINT best practices

3 days, 40% lecture 60% programming exercises
target audience: UNIX power users and administrators

The course is a practical coverage of advanced UNIX shell programming. It includes features needed in most shell programming projects, such as command line argument processing and usage of structured shell variables. It also introduces UNIX version control services – an important service area to all UNIX programming needs.

A special emphasis is put on the concept of regular expressions and two powerful utilities that use those: sed and awk. Both utilities are covered in-depth, giving skills to write very powerful and compact text processing routines. The course concentrates on Korn and POSIX shells features, making it 100% compatible with all major UNIX flavors, including HP-UX, Solaris and Linux.

The goal is to give skills needed to write powerful shell scripts, namely to control binary program execution environment and to process textual input data.

MODULE 1: UNIX VERSION CONTROL

- CVS and RCS version control systems
- Comparing CVS and RCS
- Overview of RCS functions
- Basic RCS commands (ci, co)
- Basic RCS Usage: process flow
- Installing RCS in different UNIX flavors

MODULE 2: PROCESSING SHELL ARGUMENTS AND OPTIONS

- General syntax guideline of command line arguments and options
- Parsing arguments with special variables \$*, @\$ and \$#
- Parsing arguments with getopt(1)
- Parsing arguments with ksh/POSIX shell compatible getopt builtin.

MODULE 3: STRUCTURED SHELL VARIABLES

- Default substitution of shell variables
- Special variable substitution mechanisms with \${...}
- Variable type specifications (typeset)
- Shell arrays

MODULE 4: REGULAR EXPRESSIONS

- Regular expressions overview
- Regular expression meta characters
- Extended regular expressions
- Standard usage of regular expressions (grep, egrep, vi/more search)

MODULE 5: PROGRAMMING WITH SED

- Sed overview
- Using regular expressions with sed
- Basic stream editing with sed (substitute command 's')
- Addressing input lines for editing
- Advanced sed commands and functions
- Using a sequence of sed expressions v. using sed in command pipes

MODULE 6: PROGRAMMING WITH AWK

- Awk programming language overview
- Different mechanisms to invoke an awk program
- Awk syntax: patterns and actions
- Awk input line addressing with different pattern types (line numbers, regular expression matches)
- Special pattern types (BEGIN, END)
- Programming awk actions
- Usage of predefined awk variables
- Defining and using awk functions
- Walkthrough of awk commands and builtin functions

APPENDIX A: VI-EDITOR SURVIVAL KIT

APPENDIX B: EMACS-EDITOR SURVIVAL KIT

1 day, 60% lecture 40% programming exercises
target audience: UNIX power users and administrators
prerequisites: working knowledge of UNIX shell programming

Day1 is a seminar concept for experienced students: 1 day advanced seminar, with possibility to extend the day beyond the normal office hours if students feel a need for that during the day.

Day1 seminar for sed and awk programming builds on top of standard UNIX shell programming, introducing a broad coverage of programming with regular expressions, sed and awk.

The goal is to give skills needed to enhance standard UNIX shell programs to powerful text processing applications, including web-related CGI programs.

MODULE 1: REGULAR EXPRESSIONS

- Regular expressions overview
- Regular expression meta characters
- Extended regular expressions
- Standard usage of regular expressions (grep, egrep, vi/more search)

MODULE 2: PROGRAMMING WITH SED

- Sed overview
- Using regular expressions with sed
- Basic stream editing with sed (substitute command 's')
- Addressing input lines for editing
- Advanced sed commands and functions
- Using a sequence of sed expressions v. using sed in command pipes

MODULE 3: PROGRAMMING WITH AWK

- Awk programming language overview
- Different mechanisms to invoke an awk program
- Awk syntax: patterns and actions
- Awk input line addressing with different pattern types (line numbers, regular expression matches)
- Special pattern types (BEGIN, END)
- Programming awk actions
- Usage of predefined awk variables
- Defining and using awk functions
- Walkthrough of awk commands and builtin functions
- Programmatic access to input files

APPENDIX A: VI-EDITOR SURVIVAL KIT

APPENDIX B: EMACS-EDITOR SURVIVAL KIT

3 days, 40% lecture 60% programming exercises
target audience: experienced UNIX administrators

This course introduces the performance factors affecting standalone and networked UNIX servers. It includes identifying and tuning all major system bottle neck types. A special emphasis is put on practical lab exercises, through which students learn to analyze the output of UNIX performance monitoring tools and commands.

The course concentrates on standard UNIX design and performance theory, on top of which the tools and specialties of different UNIX flavors are described, making it useful in management of multi-UNIX environments running HP-UX, Solaris and Linux.

The goal is to give skills needed to analyze UNIX systems that face performance problems, as well guidelines how to tune UNIX systems to meet performance goals.

MODULE 1: INTRODUCTION TO PERFORMANCE

- Different performance types and factors (response time, throughput, latency)
- Queuing theory of performance
- Different performance bottle neck types
- UNIX commands to identify a system bottle neck

MODULE 2: CPU BOTTLE NECK: IMPROVING PROCESS SCHEDULING

- The concept of execution context (process v. thread)
- User and system level CPU utilization
- UNIX process scheduling principles
- Controlling process priorities and nice values
- Running real-time processes
- Tuning a CPU-bound UNIX system

MODULE 3: MEMORY BOTTLE NECK: IMPROVING MEMORY ALLOCATION

- UNIX memory management concepts (virtual address space, paging, swapping)
- Swap space configuration, reservation and allocation
- Buffer cache hit rate optimization
- Tuning a memory-bound UNIX system

MODULE 4: DISK BOTTLE NECK: DISK CONFIGURATION ISSUES

- Disk access types (block and character)
- Logical io v. physical io – the role of buffer cache
- Disk allocation types (standalone disks, logical disks, RAID implementations) and performance
- HP-UX: Optimizing logical volumes for performance
- Solaris: Optimizing DiskSuite meta disks for performance
- Linux: Optimizing logical volumes for performance
- Tuning a disk-bound UNIX system

MODULE 5: DISK BOTTLE NECK: FILE SYSTEM ISSUES

- Disk io read and write data flow
- Logical structure of i-node
- Path names and i-node traversal
- The impact of file system parameters and mount options
- The impact of symbolic links
- Tuning file systems to speed up logical disk io

MODULE 6: NETWORK BOTTLE NECK AND WRAP UP

- Identifying a server bottle neck (data base server, NFS server)
- Identifying a network bottle neck
- Performance monitoring flowchart and UNIX tools
- The impact of kernel parameters
- Tuning systems with server bottle necks

4 days, 40% lecture 60% programming exercises
target audience: experienced HP-UX programmers

This course introduces the performance factors affecting standalone and networked HP-UX systems, including identification and analysis of all major system bottle neck types. A special emphasis is put on programming exercises, through which students learn how to write programs to analyze HP-UX system performance, as well as the performance impacts of application programming tasks such as management of processes and threads.

The course covers HP-UX related design topics to backup standard UNIX performance theory, on top of which the tools and specialties of HP-UX are described.

The goal is to give skills needed to analyze HP-UX performance from application programmers perspective.

MODULE 1: INTRODUCTION TO PERFORMANCE

- Different performance types and factors (response time, throughput, latency)
- Queuing theory of performance
- Different performance bottle neck types
- HP-UX commands to identify a system bottle neck (standard UNIX v. HP-UX specific)

MODULE 2: CPU BOTTLE NECK: IMPROVING PROCESS SCHEDULING

- PA-RISC CPU architecture (CPU instructions, TLB & Cache & Memory access)
- The concept of execution context (process v. thread)
- User and system level CPU utilization
- HP-UX process scheduling principles (process components, life cycle, states)
- Programmatic control of processes
- Tuning principles of a CPU-bound HP-UX system

MODULE 3: RELEASING CPU BOTTLE NECK: PROGRAMMING WITH POSIX THREADS

- Thread creation and destruction
- Canceling threads
- Thread scheduling
- Basic synchronization resources (mutexes, condition variables)
- Performance considerations of multi-threading

MODULE 4: MEMORY BOTTLE NECK: IMPROVING MEMORY ALLOCATION

- UNIX memory management concepts (virtual address space, paging, swapping)
- Swap space configuration, reservation and allocation
- Buffer cache hit rate optimization
- Tuning a memory-bound UNIX system

MODULE 5: RELEASING MEMORY BOTTLE NECK: PROGRAMMING WITH IPC SHARED MEMORY

- Overview of SysV shared memory structures
- Administering SysV shared memory (ipcs, ipcrm)
- Programming with SysV shared memory segments

MODULE 6: DISK BOTTLE NECK: DISK CONFIGURATION ISSUES

- Disk access types (block and character)
- Logical io v. physical io – the role of buffer cache
- Disk allocation types (standalone disks, logical disks, RAID implementations) and performance
- HP-UX: Optimizing logical volumes for performance

MODULE 7: RELEASING DISK BOTTLE NECK: FILE SYSTEM ISSUES

- Disk io read and write data flow
- Logical structure of i-node
- Path names and i-node traversal
- The impact of symbolic links
- HP-UX system calls for file system access

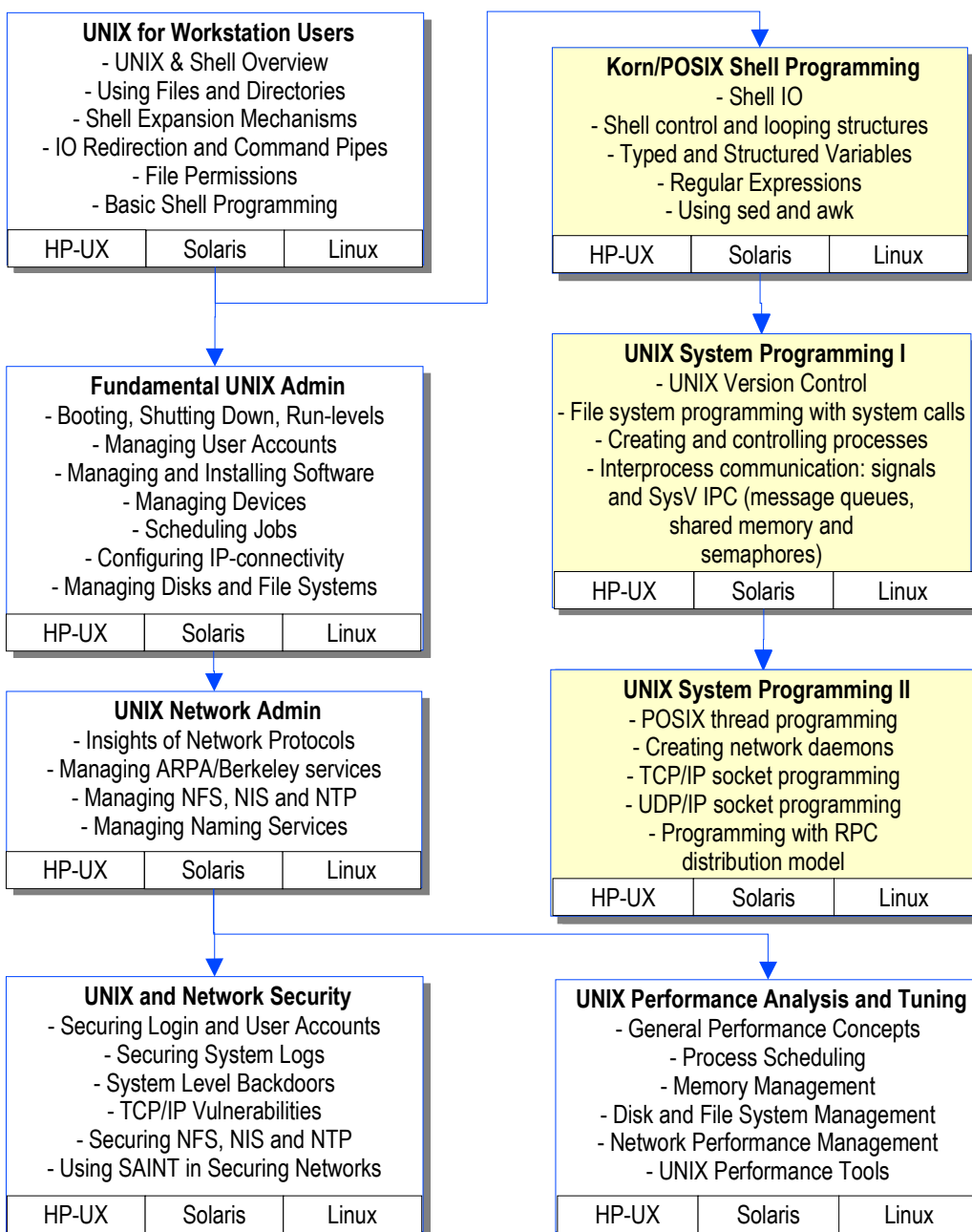
MODULE 8: NETWORK BOTTLE NECK AND WRAP UP

- Identifying a server bottle neck (data base server, NFS server)
- Identifying a network bottle neck
- Performance monitoring flowchart and HP-UX tools
- The impact of kernel parameters

UNIX flavors such as HP-UX, Solaris and Linux provide an outperforming environment for application development. The effective usage of those platforms, how ever, require a broad knowledge of of many UNIX services including:

- i) version control services and make routines for application build-up
- ii) effective shell programming for environment control and data processing
- iii) programming with system calls and socket libraries for creation of large distributed C and C++ applications.

ReSolution offers a 3 course road map designed to cover all skills needed in effective UNIX system programming – an unique path for all those who use UNIX platforms in demanding application development.



**4 days, 50% lecture 50% programming exercises
target audience: UNIX power users and administrators**

The course is a practical coverage of basic and advanced UNIX shell programming. It includes features needed in most shell programming projects, such as shell control structures, command line argument processing and usage of structured shell variables.

A special emphasis is put on the concept of regular expressions and two powerful utilities that use those: sed and awk. Both utilities are covered in-depth, giving skills to write very powerful and compact text processing routines. The course concentrates on Korn and POSIX shells features, making it 100% compatible with all major UNIX flavors, including HP-UX, Solaris and Linux.

The goal is to give skills needed to write powerful shell scripts, namely to control binary program execution environment and to process textual data.

**MODULE 1: SHELL ENVIRONMENT
OVERVIEW**

- Shell basics, different shells available
- Shell initialization files

**MODULE 2: INTRODUCTION TO
KORN/POSIX SHELL PROGRAMMING**

- Script execution and debugging
- Getting input (environment variables, positional parameters, read command)

**MODULE 3: KORN/POSIX SHELL
BRANCHING STRUCTURES**

- The concept of a process return value
- Evaluating numerical, string and file attribute expressions (test)
- Conditional branching (if-then-else-fi)
- Multibranch testing (case-in-esac)

**MODULE 4: KORN/POSIX SHELL
LOOPING STRUCTURES**

- Arithmetic expressions (expr, let)
- Predefined iteration (for-do-done)
- Conditional looping (while-do-done, until-do-done)
- Breaking out from a loop (break, continue)

**MODULE 5: PROCESSING SHELL
ARGUMENTS AND OPTIONS**

- General syntax guideline of command line arguments and options
- Parsing arguments with special variables \$*, @\$ and \$#
- Parsing arguments with getopt(1)
- Parsing arguments with ksh/POSIX shell compatible getopt builtin.

**MODULE 6: STRUCTURED SHELL
VARIABLES**

- Default substitution of shell variables
- Special variable substitution mechanisms with \${...}
- Variable type specifications (typeset)
- Shell arrays

MODULE 7: REGULAR EXPRESSIONS

- Regular expressions overview
- Regular expression meta characters
- Extended regular expressions
- Standard usage of regular expressions (grep, egrep, vi/more search)

MODULE 8: PROGRAMMING WITH SED

- Sed overview
- Using regular expressions with sed
- Basic stream editing with sed (substitute command 's')
- Addressing input lines for editing
- Advanced sed commands and functions
- Using a sequence of sed expressions v. using sed in command pipes

MODULE 9: PROGRAMMING WITH AWK

- Awk programming language overview
- Different mechanisms to invoke an awk program
- Awk syntax: patterns and actions
- Awk input line addressing mechanisms
- Special pattern types (BEGIN, END)
- Programming awk actions
- Usage of predefined awk variables
- Defining and using awk functions
- Walkthrough of awk commands and builtin functions

APPENDIX A: VI-EDITOR SURVIVAL KIT

**APPENDIX B: EMACS-EDITOR SURVIVAL
KIT**

3 days, 50% lecture 50% lab exercises
target audience: UNIX system programmers

pre-requisites: working knowledge of UNIX admin. and C-language

The course describes the core aspects of writing UNIX system programs by using standard UNIX system call layer. The topics covered include using systems calls to do blocking and non-blocking file system io, controlling processes and their environment, using IPC process communication mechanisms and programming using POSIX compliant threads. The lectures emphasis on UNIX structures and operational principles to build a basis for implementations which are written using C-language.

The goal is to give a solid and practical knowledge of UNIX internal structures and system programming.

MODULE 1: UNIX VERSION CONTROL

- CVS and RCS version control systems
- Comparing CVS and RCS
- Overview of RCS functions
- Basic RCS commands (ci, co)
- Basic RCS Usage: process flow
- Installing RCS in different UNIX flavors

MODULE 2: UNIX SYSTEM CALL LAYER

- UNIX layers for programmers
- System call layer documentation
- Programmatic access to kernel data structures

MODULE 3: FILE SYSTEM OPERATIONS

- General file system concepts (i-node, v-node)
- File system related system calls
- Traveling and examining directories
- Getting file related information (fstat, fcntl)

MODULE 4: FILE SYSTEM IO

- Different modes for file io (open, read, write)
- Creating file descriptors for files, pipes and sockets
- Doing blocking and non-blocking io
- Parallel file descriptor operations (select, poll)

MODULE 5: MANAGING PROCESSES

- The concept of a process
- Process components and environment (putenv, getenv, process id's)
- Process life cycle and states
- Programmatic control of processes (fork, exec, exit, wait)
- Programmatic access to process table (pstat)
- Creating daemon processes

MODULE 5: INTERPROCESS COMMUNICATION - SIGNALS

- Overview and concept of signals
- Signal types and usage scenarios
- Writing signal handlers (sigaction(), signal())

MODULE 6: INTERPROCESS COMMUNICATION – SYSV MESSAGE QUEUES

- Overview of message queues and FIFO queues
- Administering SysV message queues (ipcs, ipcrm)
- Programming with SysV message queues

MODULE 7: INTERPROCESS COMMUNICATION – SYSV SHARED MEMORY

- Overview of SysV shared memory structures
- Administering SysV shared memory (ipcs, ipcrm)
- Programming with SysV shared memory segments

MODULE 8: INTERPROCESS COMMUNICATION – SYSV SEMAPHORES

- Concepts of SysV semaphore sets and semaphores
- Administering SysV semaphore sets (ipcs, ipcrm)
- Theory of SysV Semaphore operations
- Programming with SysV semaphores

3 days, 50% lecture 50% lab exercises
target audience: UNIX system programmers

pre-requisites: working knowledge of UNIX admin. and C-language

The course describes the core aspects of UNIX multithreaded and network programming. A special emphasis is put on all aspects of writing state-of-the-art standalone network servers – these aspects include writing multithreaded programs, proper creation of daemon processes, as well as programming with TCP/IP and UDP/IP socket libraries. The course also covers an introduction to RPC programming, offering extra power in defining application protocols, and deploying the created networking applications under the control of standard UNIX services.

The goal is to give a solid and practical knowledge of writing multithreaded, distributed and secure UNIX applications on top of standard UNIX networking protocol libraries.

MODULE 1: POSIX THREAD PROGRAMMING

- POSIX thread overview and concepts
- Thread creation and destruction
- Creating and using thread-specific data
- Controlling and accessing thread attributes
- Canceling threads

MODULE 2: POSIX THREAD SYNCHRONIZATION

- Thread scheduling
- Basic synchronization resources (mutexes, condition variables, read-write locks)
- Using thread signals
- Performance considerations of multi-threading

MODULE 3: TCP/IP NETWORKING CONCEPTS

- OSI networking model v. TCP/IP addressing schemes
- Essential protocol insights (ip, arp, tcp, udp, rpc, application protocols)
- Socket connections and routing
- Overview of UNIX tools for network troubleshooting

MODULE 4: CREATING NETWORK DAEMONS

- Inetd concepts and usage
- Creating and adding own services to inetd control
- Requirements for standalone daemon processes
- Creating standalone daemon processes

MODULE 5: SOCKET PROGRAMMING WITH TCP TRANSPORT LAYER

- Creating a server side socket with predefined port number
- Creating a client side socket
- Using BSD socket services to manage socket connections

MODULE 6: SOCKET PROGRAMMING WITH UDP TRANSPORT LAYER

- UDP transport concepts
- Creating an UDP server process
- Creating an UDP client process
- Sending and receiving datagrams
- Managing reliability of datagram transfer

MODULE 7: INTRODUCTION TO RPC PROGRAMMING

- Remote Procedure Call (RPC) concepts
- Defining RPC application protocols (rpcgen)
- Creating RPC-based server and client processes

MODULE 8: SECURITY ISSUES IN UNIX NETWORK PROGRAMMING

- Inetd network daemon issues
- TCP/UDP/IP protocol issues
- RPC issues
- Using ssh to authenticate and encrypt socket connections
- Using tcpwrapper to control network service access