

# UNIX System Programming Using C++

*Terrence Chan*



*An imprint of* **Pearson Education**

S I T Library  
Valachil, Mangalore



Accn No: 013002

Srinivas Institute of Technology

Acc No: 13002

Call No: 13002

Copyright © 1997 by Pearson Education, Inc.

This edition is published by arrangement with Pearson Education, Inc. and Dorling Kindersley Publishing, Inc.

This book is sold subject to the condition that it shall not, by way of trade or otherwise, be lent, resold, hired out, or otherwise circulated without the publisher's prior written consent in any form of binding or cover other than that in which it is published and without a similar condition including this condition being imposed on the subsequent purchaser and without limiting the rights under copyright reserved above, no part of this publication may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording or otherwise), without the prior written permission of both the copyright owner and the above-mentioned publisher of this book.

**Pearson Prentice Hall™** is a trademark of Pearson Education, Inc.

**Pearson®** is a registered trademark of Pearson Plc.

**Prentice Hall®** is a registered trademark of Pearson Education, Inc.

ISBN 978-81-317-2280-0

First Impression, 2008

*This edition is manufactured in India and is authorized for sale only in India, Bangladesh, Bhutan, Pakistan, Nepal, Sri Lanka and the Maldives. Circulation of this edition outside of these territories is UNAUTHORIZED.*

Published by Dorling Kindersley (India) Pvt. Ltd., licensees of Pearson Education in South Asia.

Head Office: 482, F.I.E., Patparganj, Delhi 110 092, India.

Registered Office: 14 Local Shopping Centre, Panchsheel Park, New Delhi 110 017, India.

Printed in India by Sheel Print-N-Pack

---

---

## Table of Contents

	Preface .....	xi
<b>1</b>	<b>UNIX and ANSI Standards .....</b>	<b>1</b>
✓	The ANSI C Standard .....	2
	The ANSI/ISO C++ Standard .....	7
	Differences Between ANSI C and C++ .....	7
	The POSIX Standards .....	8
	The POSIX Environment .....	11
	The POSIX Feature Test Macros .....	11
	Limits Checking at Compile Time and at Run Time ..	13
	The POSIX.1 FIPS Standard .....	18
	The X/Open Standards .....	18
	Summary .....	19
	References .....	19
<b>2</b>	<b>C++ Language Review .....</b>	<b>21</b>
	C++ Features for Object-Oriented Programming .....	22
	C++ Class Declaration .....	23
	Friend Functions and Classes .....	28
	Const Member Functions .....	30

---

Table of Contents

C++ Class Inheritance .....	31
Virtual Functions .....	34
Virtual Base Classes .....	36
Abstract Classes .....	39
The new and delete Operators .....	42
Operator Overloading .....	46
Template Functions and Template Classes .....	49
Template Functions .....	50
Template Classes .....	52
Exception Handling .....	57
Exceptions and Catch-Blocks Matching .....	63
Function Declarations with Throw .....	63
The Terminate and Unexpected Functions .....	64
Summary .....	65
References .....	66
<b>3 C++ I/O Stream Classes .....</b>	<b>67</b>
The I/O Stream Classes .....	68
The istream Class .....	68
The ostream Class .....	70
The iostream Class .....	72
The ios Class .....	72
The Manipulators .....	75
The File I/O classes .....	76
The stringstream Classes .....	79
Summary .....	81
<b>4 Standard C Library Functions .....</b>	<b>83</b>
<stdio.h> .....	84
<stdlib.h> .....	88
<string.h> .....	93
strspn, strcspn .....	94
strtok .....	95
strerror .....	97
<memory.h> .....	98
<time.h> .....	103
<assert.h> .....	106
<stdarg.h> .....	107
Command Line Arguments and Switches .....	112

<setjmp.h>	115
<pwd.h>	117
<grp.h>	119
<crypt.h>	121
Summary	123
<b>5</b> ✓ <b>UNIX and POSIX APIs</b>	<b>125</b>
The POSIX APIs	126
The UNIX and POSIX Development Environment	126
API Common Characteristics	127
Summary	128
<b>6</b> ✓ <b>UNIX Files</b>	<b>129</b>
File Types	130
The UNIX and POSIX File Systems	133
The UNIX and POSIX File Attributes	134
Inodes in UNIX System V	136
Application Program Interface to Files	137
UNIX Kernel Support for Files	139
Relationship of C Stream Pointers and File Descriptors	142
Directory Files	143
Hard and Symbolic Links	144
Summary	146
<b>7</b> ✓ <b>UNIX File APIs</b>	<b>147</b>
General File APIs	148
open	148
creat	152
read	152
write	154
close	155
fcntl	156
lseek	158
link	159
unlink	160
stat, fstat	162
access	167
chmod, fchmod	168

---

Table of Contents

chown, fchown, lchown .....	170
utime .....	172
File and Record Locking .....	173
Directory File APIs .....	178
Device File APIs .....	182
FIFO File APIs .....	185
Symbolic Link File APIs .....	188
General File Class .....	191
Regfile Class for Regular Files .....	194
dirfile Class for Directory Files .....	196
FIFO File Class .....	198
Device File Class .....	199
Symbolic Link File Class .....	201
File Listing Program .....	203
Summary .....	205
<b>8 UNIX Processes .....</b>	<b>207</b>
UNIX Kernel Support for Processes .....	208
Process APIs .....	211
fork, vfork .....	211
_exit .....	214
wait, waitpid .....	216
exec .....	220
pipe .....	224
I/O Redirection .....	228
Process Attributes .....	238
Change Process Attributes .....	241
A Minishell Example .....	242
Summary .....	257
<b>9 Signals .....</b>	<b>259</b>
The UNIX Kernel Supports of Signals .....	261
signal .....	262
Signal Mask .....	264
sigaction .....	268
The SIGCHLD Signal and the waitpid API .....	271
The sigsetjmp and siglongjmp APIs .....	272
kill .....	274
alarm .....	276

Interval Timers .....	278
POSIX.1b Timers .....	282
timer Class .....	287
Summary .....	294
<b>10 Interprocess Communication .....</b>	<b>295</b>
POSIX.1b IPC Methods .....	296
The UNIX System V IPC Methods .....	297
UNIX System V Messages .....	297
UNIX Kernel Support for Messages .....	298
The UNIX APIs for Messages .....	300
msgget .....	302
msgsnd .....	303
msgrcv .....	305
msgctl .....	307
Client/Server Example .....	308
POSIX.1b Messages .....	315
POSIX.1b Message Class .....	319
UNIX System V Semaphores .....	322
UNIX Kernel Support for Semaphores .....	323
The UNIX APIs for Semaphores .....	325
semget .....	326
semop .....	327
semctl .....	329
POSIX.1b Semaphores .....	332
UNIX System V Shared Memory .....	335
UNIX Kernel Support for Shared Memory .....	335
The UNIX APIs for Shared Memory .....	337
shmget .....	338
shmat .....	339
shmdt .....	340
shmctl .....	341
Semaphore and Shared Memory Example .....	343
Memory Mapped I/O .....	349
Memory mapped I/O APIs .....	350
mmap .....	350
munmap .....	352
msync .....	353

---

Table of Contents

Client/Server Program Using Mmap .....	354
POSIX.1b Shared Memory .....	357
POSIX.1b Shared Memory and Semaphore Example	359
Summary .....	365
<b>11 Sockets and TLI .....</b>	<b>367</b>
Sockets .....	368
socket .....	371
bind .....	372
listen .....	373
connect .....	373
accept .....	374
send .....	375
sendto ..	376
recv .....	376
recvfrom .....	377
shutdown .....	377
a Stream Socket Example .....	378
Client/Server Message-Handling Example .....	391
TLI .....	395
TLI APIs .....	396
t_open .....	399
t_bind .....	402
t_listen .....	404
t_accept .....	405
t_connect .....	407
t_snd, t_sndudata .....	408
t_rcv, t_rcvudata, t_rcvuderr .....	410
t_sndrel, t_rcvrel .....	413
t_snddis, t_rcvdis .....	414
t_close .....	415
TLI Class .....	416
Client/Server Message Example .....	423
Datagram Example .....	428
Summary .....	434
<b>12 Remote Procedure Calls .....</b>	<b>435</b>
History of RPC .....	436
RPC Programming Interface Levels .....	436



RPC Library Functions .....	437
rpcgen .....	439
clnt_create .....	445
The rpcgen Program .....	446
A Directory Listing Example Using rpcgen .....	447
rpcgen Limitations .....	452
Low-Level RPC Programming Interface .....	452
XDR Conversion Functions .....	452
Lower Level RPC APIs .....	455
RPC Classes .....	457
svc_create .....	474
svc_run .....	476
svc_getargs .....	476
svc_sendreply .....	476
clnt_create .....	477
clnt_call .....	478
Managing Multiple RPC Programs and Versions .....	478
Authentication .....	483
AUTH_NONE .....	484
AUTH_SYS (or AUTH_UNIX) .....	485
AUTH_DES .....	487
Directory Listing Example with Authentication .....	490
RPC Broadcast .....	498
RPC Broadcast Example .....	500
RPC Call Back .....	502
Transient RPC Program Number .....	509
RPC Services Using Inetd .....	514
Summary .....	520
<b>13 Multithreaded Programming .....</b>	<b>521</b>
Thread Structure and Uses .....	523
Threads and Lightweight Processes .....	524
Sun Thread APIs .....	526
thr_create .....	526
thr_suspend, thr_continue .....	528
thr_exit, thr_join .....	528
thr_sigsetmask, thr_kill .....	529
thr_setprio, thr_getprio, thr_yield .....	531

---

Table of Contents

thr_setconcurrency, thr_getconcurrency .....	.531
Multithreaded Program Example .....	.532
POSIX.1c Thread APIs .....	.536
pthread_create .....	.537
pthread_exit, pthread_detach, pthread_join .....	.539
pthread_sigmask, pthread_kill .....	.540
sched_yield .....	.541
Thread Synchronization Objects .....	.541
Mutually Exclusive Locks (mutex Locks) .....	.542
Sun Mutex Locks .....	.543
POSIX.1c Mutex Locks .....	.544
Mutex Lock Examples .....	.545
Condition Variables .....	.550
Sun Condition Variables .....	.550
Condition Variable Example .....	.551
POSIX.1c Condition Variables .....	.554
Sun Read-Write Locks .....	.555
Semaphores .....	.560
Thread-Specific Data .....	.564
The Multithreaded Programming Environment .....	.571
Distributed Multithreaded Application Example .....	.571
Summary .....	.584
<b>Index .....</b>	<b>.585</b>

---

---

## Preface

The content of this book is derived from my several years of teaching Advanced UNIX Programming with C and C++ at two University of California Extensions (Berkeley and Santa Cruz). The objectives of the courses were to teach students advanced programming techniques using UNIX system calls and the ANSI C and C++ programming languages. Specifically, students who took the courses learned the following:

- Advanced ANSI C and C++ programming techniques, such as how to use function pointers and create functions that accept variable numbers of arguments
- The ANSI C library functions and C++ standard classes, and how to use them to reduce development time and to maximize portability of their applications
- Familiarity with the UNIX kernel structure and the system calls. These allow users to write sophisticated applications to manipulate system resources (e.g., files, processes, and system information), and to design new operating systems
- How to create network-based, multitasking, client/server applications which run on heterogenous UNIX platforms

The objective of this book is to convey to readers the techniques and concepts stated above. Furthermore, this book provides more detailed explanations and comprehensive examples on each topic than can be done in a course. Thus, readers can gain a better understanding of the subject matter and can learn at their own pace. This book also describes the latest advanced UNIX programming techniques on remote procedure calls and multithreaded programs. These techniques are important for the development of advanced distributed client/server applications in a symmetrical multiprocessing and network-based computing environment.

---

## Preface

All the aforementioned information will be described in the C++ language. This is because in the last few years more and more advanced software developers are using C++ in applications development. This is due to the fact that the C++ language provides much stronger type-checking and includes object-oriented programming constructs than other procedural programming languages. These features are very useful in facilitating large-scale, complex UNIX system applications development and management.

This book covers the C++ programming language based on the draft version of the ANSI/ISO C++ standard [1, 2, 3]. Most of the latest C++ compilers provided by various computer vendors (e.g., Sun Microsystems Inc., Microsoft Corporation, Free Software Foundation, etc.) are compliant with this standard.

In addition to the C++ language, some significant C library functions, as defined by the ANSI C standard [4], are also described in this book. These functions are not covered by the C++ standard classes or by the UNIX application program interface. Thus, it is important that users be familiar with these to increase their knowledge base and choices of library functions.

The UNIX operating systems covered in this book include: UNIX System V.3, UNIX System V.4, BSD UNIX 4.3 and 4.4, Sun OS 4.1.3, and Solaris 2.4. The last two operating systems belong to SUN Microsystems, where Sun OS 4.1.3 is based on BSD 4.3 with UNIX System V.3 extensions, and Solaris 2.4 is based on the UNIX System V.4.

Although the primary focus of this book is on UNIX system programming, the IEEE (Institute of Electrical and Electronics Engineering) POSIX.1, POSIX.1b, and POSIX.1c standards are also covered in detail. This is to aid system programmers to develop applications that can be readily ported to different UNIX systems, as well as to POSIX-compliant systems (e.g., VMS and Windows-NT). This is important as most advanced commercial software products must run on heterogeneous platforms by various computer vendors. Thus, the POSIX and ANSI standards can help users create highly platform-independent applications.

## Target Audience

The book is targeted to benefit experienced software engineers and managers who are working on advanced system applications development in a UNIX environment. The products they develop may include advanced network-based client/server applications, distributed database systems, operating systems, compilers, or computer-aided design tools.

The readers should be familiar with the C++ language based on the AT&T version 3.0 (or the latest) and should have developed some C++ application programs on their own in the past. Moreover, the readers should be familiar with at least one version of UNIX system (e.g.,

UNIX System V). Specifically, the readers should know the UNIX file system architecture, user accounts assignment and management, file access control, and jobs control methods. Readers who need to brush up on UNIX system knowledge may consult any text book covering an introduction to the UNIX system.

## Book Content

Although this book covers the ANSI C++ and C library functions and UNIX APIs extensively, the primary focus in describing these functions is to convey the following information to readers:

- Purposes of these functions
- Conformance of these functions to standard(s)
- How to use these functions
- Examples of their uses
- Where appropriate, how these functions are implemented in a UNIX system
- Any special considerations (e.g., conflict between the UNIX and POSIX standards) in using these functions

It is not the intention of the author to make this book a UNIX system programmer's reference manual. Thus, the function prototypes and header files required to use the ANSI library and UNIX API functions are described, but the detailed error codes that may be returned by these functions and the archive or shared libraries needed by users' programs will not be depicted. This type of information may be obtained via either the man pages of the functions or the programmer's reference manuals from the users' computer vendors.

The general organization of this book is:

- Chapter 1 describes the history of the C++ programming language and various UNIX systems. It also describes the ANSI/ISO C, ANSI/ISO C++, IEEE POSIX.1, POSIX.1b, and POSIX.1c standards
- Chapters 2 and 3 review the draft ANSI/ISO C++ programming language and object-oriented programming techniques. The C++ I/O stream classes, template functions, and exception handlings are also depicted in detail
- Chapter 4 describes the ANSI C library functions
- Chapter 5 gives an overview of the UNIX and POSIX APIs. Special header files and compile time options, as required by various standards, are depicted.
- Chapters 6 and 7 describe UNIX and POSIX.1 file APIs. These depict APIs that can be used to control various types of files in a system. They also describe file-locking techniques used to synchronize files in a multiprocessing environment

---

## Preface

- Chapter 8 describes UNIX and POSIX.1 process creation and control methods. After reading this chapter, readers can write their own multiprocessing applications, such as a UNIX shell
- Chapter 9 describes UNIX and POSIX.1 signal handling methods
- Chapter 10 describes UNIX and POSIX.1b interprocess communication methods. These techniques are important in creating distributed client/server applications.
- Chapter 11 describes advanced network programming techniques using UNIX sockets and TLI
- Chapter 12 describes remote procedure call. This is important for development of network transport protocol-independent client/server application development on heterogeneous UNIX platforms
- Chapter 13 describes multithreaded programming techniques. These techniques allow applications to make efficient use of multiprocessor resources available on any machines on which they run

Note that although this book is based on C++, the focus on this book is not object-oriented programming techniques. This is because some readers are expected to be new to UNIX system programming and/or C++ language, thus it may be difficult for these readers to learn both object-oriented and system programming techniques at the same time. However, this book includes many useful C++ classes for interprocess communication, sockets, TLI, remote procedure call, and multithreaded programming. These classes encapsulate the low-level programming interface to these advanced system functions, and can be easily extended and incorporated into user applications to reduce their development efforts, time, and costs.

## Example Programs

Throughout the book extensive example programs are shown to illustrate uses of the C++ classes, library functions, and system APIs. All the examples have been compiled by a Sun Microsystems C++ (version 4.0) compiler and tested on a Sun SPARC-20 workstation running Solaris 2.4. These examples are also compiled and tested using the Free Software Foundation GNU g++ compiler (version 2.6.3) on a Sun SPARC-20 workstation. Since the GNU g++ compilers can be ported to various hardware platforms, the examples presented in this book should run on different platforms (e.g., Hewlett Packard's HP-UX and International Business Machines's AIX) also.

Readers are encouraged to try out the example programs on their own systems to get more in-depth familiarity of this subject matter. Users may download an electronic copy of the example programs via anonymous ftp to <ftp.prenhall.com>. The directory that stores the example tar file is `/pub/ptr/professional_computer_science.w-022/chan/unixsys`. There are README files in the tar file that describe the programs and their cross references to chapters in the book. Finally, readers are welcome to send Emails to the author at [twc@tjssystems.com](mailto:twc@tjssystems.com).

---

## Acknowledgments

I would like to thank Peter Collinson, Jeff Gitlin, Chi Khuong, Frank Mitchell, and my wife Jessica Chan for their careful reviewing of the book manuscript. Much of their valuable input has been incorporated in the final version of this book. Furthermore, I would like to extend my appreciation to Greg Doench, Nick Radhuber, and Brat Bartow for their valuable assistance in helping me through the preparation and publication process of this book.

Finally, I am grateful to my former students at the University of California Santa Cruz Extension who took my Advanced UNIX System Calls course and gave me valuable feedback in the refinement of the course material, much of which is used throughout this book.

## References

- [1]. Margaret A. Ellis and Bjarne Stroustrup, *The Annotated C++ Reference Manual*, Addison-Wesley, 1990.
- [2]. Andrew Koenig, *Working Paper for Draft Proposed International Standard for Information Systems -- Programming Language C++ (Committees: WG21/N0414, X3J16/94-0025)*, January 1994.
- [3]. Bjarne Stroustrup, *Standardizing C++. The C++ Report. Vol. 1. No. 1*, 1989.
- [4]. American National Standard Institute, *American National Standard for Information Systems - Programming Language C, X3.159 - 1989*, 1989.

