Contents

Preface. xvii

1. Tracing Program Execution With the TNF Utilities 1

Defining Types of Users 2

Using Existing Probe Points 2

Debugging a Program 2

Inserting Probe Points in a Library 2

Tracing Kernel Execution 2

How TNF Works 3

Inserting Probe Points 4

Starting prex 4

Running prex 7

Reading the Trace File 22

Kernel Tracing 24

Controlling Kernel Tracing (`prex`) 25

Extracting Kernel Trace Data (`tnfxtract`) 29

Examining Kernel Trace Data (`tnfdump`) 30
Available Kernel Probes (`tnf_probes`) 30
Thread Probes 31
Shell Script for Kernel Tracing 35
Advanced Topics 39
Inserting Probe Points 39
Defining User Types for Probe Points 44
Performance Issues 48
`/proc` 48
`dlopen()` and `dlclose()` and History. 49
Signals 49
Failure of Event-Writing Operations 49
Target Executing a `fork()` or `exec()` 49
2. `lex`–Lexical Analysis 51
Generating a Lexical Analyzer Program 52
Writing lex Source 54
The Fundamentals of lex Rules 54
Advanced `lex` Features 61
C++ Mangled Symbols 72
Using lex and yacc Together 73
Automaton 77
Summary of Source Format 77
3. yacc — A Compiler Compiler 81
Basic Specifications 84

Actions . . . 87

Lexical Analysis . . . 91

Parser Operation . . . 93

Ambiguity and Conflicts . . . 99

Precedence . . . 104

Error Handling . . . 109

The yacc Environment . . . 112

Hints for Preparing Specifications . . . 114

Input Style . . . 114

Left Recursion . . . 114

C++ Mangled Symbols . . . 116

Lexical Tie-Ins . . . 116

Reserved Words . . . 117

Advanced Topics . . . 117

Simulating error and accept in Actions . . . 117

Accessing Values in Enclosing Rules . . . 118

Support for Arbitrary Value Types . . . 119

yacc Input Syntax . . . 121

Examples . . . 125

A Simple Example . . . 125

An Advanced Example . . . 130

4. make Utility . . . 143

Dependency Checking: make vs. Shell Scripts . . . 144

Writing a Simple Makefile . . . 145

Basic Use of Implicit Rules 147
Processing Dependencies 148
Null Rules 152
Special Targets 152
Unknown Targets 152
Duplicate Targets 153
Reserved make Words 153
Running Commands Silently 154
Automatic Retrieval of SCCS Files 155
Passing Parameters: Simple make Macros 157
.KEEP_STATE and Command Dependency Checking . . . 158
.KEEP_STATE and Hidden Dependencies 159
Displaying Information About a make Run 161
Using make to Compile Programs 164
A Simple Makefile 164
Using make's Predefined Macros 165
Using Implicit Rules to Simplify a Makefile: Suffix Rules . 166
When to Use Explicit Target Entries vs. Implicit Rules . . . 168
Implicit Rules and Dynamic Macros 168
Adding Suffix Rules 171
Pattern-Matching Rules:An Alternative to Suffix Rules . . 173
Building Object Libraries 180
Libraries, Members, and Symbols 180
Library Members and Dependency Checking 181

Using make to Maintain Libraries and Programs 182
More about Macros 182
Linking with System-Supplied Libraries 185
Compiling Programs for Debugging and Profiling 186
Compiling Debugging and Profiling Variants 188
Maintaining Separate Program and Library Variants 189
Maintaining a Directory of Header Files 194
Compiling and Linking with Your Own Libraries 194
Nested make Commands 195
Passing Parameters to Nested make Commands 197
Compiling Other Source Files 200
Maintaining Shell Scripts with make and SCCS 202
Running Tests with make 203
Maintaining Software Projects 206
Organizing a Project for Ease of Maintenance 207
Building the Entire Project 209
Maintaining Directory Hierarchies with the Recursive Makefiles 211
Maintaining Recursive Targets 212
Maintaining a Large Library as a Hierarchy of Subsidiaries 213
Reporting Hidden Dependencies to make 215
make Enhancements Summary 216
Default Makefile 216
The State File .make.state 216

Hidden-Dependency Checking 216
Command-Dependency Checking. 216
Automatic Retrieval of SCCS Files 217
Pattern-Matching Rules 217
Pattern-Replacement Macro References 218
New Options 220
Support for C++ and Modula-2 220
Naming Scheme for Predefined Macros 220
New Special-Purpose Targets 221
New Implicit lint Rule 221
Macro Processing Changes 221
Improved ar Library Support 223
Target Groups. 224
Incompatibilities with Previous Versions 224
The -d Option. 224
Dynamic Macros 224
Tilde Rules 225
Target Names 225
5. SCCS Source Code Control System 227
The sccs Command 228
The sccs create Command 228
Basic sccs Subcommands 229
Deltas and Versions 229
sccs Subcommands 231

Checking Files In and Out 231
Incorporating Version-Dependent Information by Using ID Keywords 235
Making Inquiries 236
Deleting Committed Changes 240
Version Control for Binary Files. 243
Maintaining Source Directories 244
Branches 246
Using Branches 250
Administering SCCS Files 251
Interpreting Error Messages: sccs help 251
Altering History File Defaults: sccs admin 251
Validating the History File 252
Restoring the History File 253
Reference Tables 254
6. m4 Macro Processor 259
Overview. 259
m4 Macros. 261
Defining Macros. 261
Quoting 262
Arguments 264
Arithmetic Built-Ins. 267
File Inclusion 268
Diversions. 268

System Commands 269
Conditional Testing 269
String Manipulation 270
Printing 272
Summary of Built-In m4 Macros 273
A. A System V make 275
Basic Features 276
Description Files and Substitutions 281
Comments 281
Continuation Lines 281
Macro Definitions 282
General Form 282
Dependency Information 282
Executable Commands 283
Extensions of $*, $@, and $< 284
Output Translations 284
Recursive Makefiles 285
Suffixes and Transformation Rules 285
Implicit Rules 286
Archive Libraries 288
Source Code Control System (SCCS) Filenames 291
The Null Suffix 292
Included Files 293
SCCS Makefiles 293

Dynamic-Dependency Parameters 293
Command Usage 294
The make Command 294
Environment Variables 296
Suggestions and Warnings 297
Internal Rules 298
Special Rules. 299
Index 305