INDEX

Abiteboul, S., 287
Abstract Factory, 61
Abstraction, 11, 25
Ada 95, 103ff
Adapter, 120, 276
   object adapter, 149
Aggregation, 19
Alexander, C., 30, 41, 88
Algorithm, 8
   choosing, 28
Anderson, B., 41
Andrews, G., 199
Antipatterns, 86
Apple computer, 35, 37
   BENTO, 155
   human interface guidelines, 72
   Interapplication Communication Architecture (ICA), 160
   Macintosh Style Guide, 72
   OpenDoc, 99, 154
   QuickTime, 130
Appleton, B., 29, 35
Applicative programming, 22
Architecture Tradeoff Analysis Method (ATAM), 339
Automated teller machine (ATM), 244
AWT, 67, 134
Barth, T. 150
Basili, V., 7
Bass, L., 199
Batch sequential systems, 227
Beck, K., 41
Berge, C., 21
Berczuk, S., 45
Blackboard architecture, 293
Blackboard systems, 293ff
   structure, 294
   problem-solving models, 294
   role of knowledge sources, 295
Black box, 223
Boehm, B., 199
Booch, G., 41, 195
Bosch, J., 94, 371
Brockschmidt, K., 37
Broker systems, 282ff
   design, 282
   remote method call, 282
Brooks, F., 81, 204
Brown, A. W., 33
Buschmann, F., 29, 45, 204
C, C++, 103, 160, 164ff, 173, 316, 330
Callback function, 67
Cambridge Event Architecture, 278

Software Paradigms. By Stephen H. Kaisler

433
INDEX

CAML, 24
Card stack, 78
Carnegie-Mellon University (CMU), 233
Chen, Q., 360
Classification, 17
Clements, P., 34, 199
Client, 243
Client-server systems, 243
designing servers, 249
equivalent in Rapide, 330
middleware mechanisms, 249
two-tier, 245
Clustering, 19
Command languages processors (CLP), 258
Commercial off the shelf (COTS), 189, 217
dependability, 193
reliability, 218
Communicating sequential processes (CSP), 270ff
concepts, 270
problems, 271
Component object model (COM), 65, 93, 162ff,
access to objects, 169
COM objects, 163
interfaces, 164
interobject communication, 165
marshalling arguments, 170
using an object, 170
Common Ground (Tidwell), 73
Common Object Request Broker Architecture
(CORBA), 37, 113ff, 139, 146ff, 181, 195
benefits, 149
concepts, 147
container model, 113
distributed objects, 251
event service, 277
interfaces, 148ff
Object Request Broker (ORB), 147ff
performance issues, 150ff
services, 149
Common operating environment (DoD), 141ff
CommonPoint framework, 384
Component-based software engineering (CBSE), 97, 185ff
building components, 187
defining, 186
finding components, 187
harvesting for reuse, 189
problems with, 186
testing components, 189
using components, 188
Components, 12, 30, 97ff, 200
attributes, 200
dependencies among, 32
design issues, 140
design tips, 126
distributed, 138ff
dynamic data exchange (DDE), 166ff
event-based, 129ff
event-based integration, 131
executable code, 100
identifying reusable, 127
implementation design, 100
integration, 111
interface, 101, 201
interoperability, 32
models, 113
ports, 200
properties of, 123
reuse, 123
reuse metrics, 125ff
specification, 100
substitution, 112
use of, 108
Computation, 11
Concurrent programming, 15
Concurrent system, 300
coarse-grained, 301
degree of concurrency, 300
fine-grained, 301
Connectors, 201
Contracts, 353
Control binding, 120
Controllers, 377
Convolution, one-dimensional, 234
Coplien, J., 43, 80ff
CORBA, see Common Object Request Broker
Architecture
Database systems, 287ff
active, 289ff
database management systems (DBMS), 287
object-oriented, 288
Data flow systems, 221ff
applications, 223
characteristics, 222
course-grained, 225
fine-grained, 224
implementation, 223
model, 221
DARPA, see Defense Advanced Research
Projects Agency
Data binding, 122
Data parallel systems, 311ff
  examples, 312
  languages, 313
  use of, 314
DB2, 288
DBFactory, 62
Declarative languages, 22
Decision, 18
Defense Advanced Research Projects Agency (DARPA), 12, 225
Design by committee, 86
Design patterns, 12, 28–29, 41ff, 353
  adapter, 276
  architectural, 29
  consistency in, 47
  describing, 44
  discovering, 45
  idioms, 29
  learning, 47
  limitations, 93
  mediator, 276
  observer, 369
  types of, 44
  using, 46
DeSouza, D., 103
Distributed component object model (DCOM), 171, 187
  access security, 174
  activating objects, 172
  architecture, 170
  connection policy, 175
  launch security, 174
  marshalling arguments, 173
  monikers, 172
  security support, 174
Distributed system object model (DSOM), 154
Distributed systems
  advantages and disadvantages, 143
Domain-specific software architectures, 12
Dynamic binding, 121, 353
Dynamically Loadable Library (DLL), 99, 152
Domain, 208
Domain analysis, 213
Domain environments, 213
Domain model, 209, 210
Domain-Specific Software Architecture (DSSA), 12, 208
  approach, 209
  lessons learned, 213
Edit-Compile-Link-Execute (ECLE) model, 120
Encapsulation, 113
Enterprise JavaBeans (EJB), 144, 177, 390
  container, 179
  entity Beans, 179
  need for, 180
  server, 179
  session Beans, 179
  types, 179
Enumeration, 18
ET++, 35, 374, 386
Even, S., 21
Event(s), 130
  handlers, 130, 272
  in active database systems, 290
  mediating, 275
  models, 132
  notification, 272
  system design, 273
Event-driven system, 272ff
  characterizing, 278
  event buffering, 281
  exception handling, 281
  issues, 280
  quality of service, 281
  scalability, 280
  security, 281
Fayad, M., 372
Feature creep, 86
Few Roles (Coplien), 82
Flanagan, D., 180
Floyd, R., 3
Foote, B., 345
Fortran, 4, 313
  Fortran D tools, 321
  high performance, 313
Foster, I., 310
Fowler, M., 43
Framework(s), 12, 34–37, 343, 345ff
  advantages, 357
  aggregation in, 363
  application, 35
  application development using, 405
  black-box, 349
  breakage, 372
  challenges, 404
  component overlap, 370
  composition, 366
  composition issues, 367
  concurrency, 363
  control, 367
  description, 359
  designing, 361
  design guidelines, 364
  determining behavior, 363
Framework(s) (Continued)
- developing, 404
- disadvantages, 357
- documenting, 358
- domains, 372
- domain scope, 366
- elements, 350, 352
- flexible design, 362
- gap, 368
- graphical user interface, 374
- gray-box, 349
- heuristics, 360
- IBM’s San Francisco Project, 398
- implementing, 355
- instantiation, 364
- integration, 364
- interoperability, 364
- Java, 390
- J2EE, 391
- legacy System components, 370
- Microsoft’s .NET, 395
- POOMA, 402
- problems, 365
- roles, 355
- services, 352
- software, 35
- structural, 347
- Sun One, 395
- testing, 408
- usage issues, 409ff
- using, 354
- white-box, 348

Function binding, 120
Functional programming, 23–24

Gamma, E., 386
Gang of Four, 42ff, 49ff, 61
Garlan, D., 34, 111, 197, 214, 228, 233, 241, 328, 330
Gartner Group, 30
Generation, 18
Gettys, J., 67, 245
Glue code, 194ff
Goldberg, A., 35, 378
Graphical user interface (GUI), 67
Graph traversal, 19
GUI, see Graphical user interface

Hartigan, J., 19
Haskell, 24
Hayes-Roth, F., 199
HCI, see Human-computer interface
Henderson-Sellers, B., 103
Hidden assumptions, 193
Hierarchical abstraction, 254
Hoare, C. A. R., 25, 270
Hollywood principle, 346
Hooking into legacy components, 195
HotDraw, 380
Hub, Spoke, and Rim (Coplien), 82
Human-computer interface (HCI), 72
Hunt, J. F., 43

IBM
- 360/370, 258
distributed system object model (DSOM), 154
job control language, 259
San Francisco Project, 398
system object model (SOM), 151ff
Websphere, 398

Imperative languages, 21–23
Interapplication Communication Architecture (ICA), 160
Interconnection, 31
Interface, human-computer, 72
Interface definition language (IDL), 170, 173
Interpreters, 258
command language processors (CLP), 258

Interoperability, 32
components, 140ff
interviews, 35
using components, 190ff
iWarp, 235

James, M., 18
Java
- as a framework, 390
distributed object model, 180
events, 134
event handling, 136
virtual machine, 262
Java Application Framework, 177
JavaBeans, 176ff
- concept of, 176
difference from OLE, 176
test case, 177
Java Media Framework, 393ff
Java Remote Method Invocation (RMI), 180
Java 2 Platform, Enterprise Edition (J2EE), 390ff
advantages, 393

JINI, 181
- leases, 182
lookup service, 181
Job control language, 259
Johnson, R., 345, 380, 405

Kaisler, S., 103, 258, 300, 304
Kazman, R., 199, 339
Keller, J., 15
Kesselman, C., 310
Knuth, D., 19
Kogut, P., 34
Krutchen, P., 30, 100
Kuhn, T., 2
Kung, H. T., 233
LaLonde, W. R., 378
Leeb, A., 31
Lego, 97, 127, 187
Linda, 296ff
issues, 298
objects, 296
operators, 297
tuple matching, 297
Linnaeus, Carl, 42
Linton, M., 35, 374
Listeners (Java), 131, 134
Local object proxy (COM), 169
Logic programming languages, 2
MacApp, 35, 374, 381ff
example, 383
Macintosh
OS component model, 114
Style Guide, 72
Maclean, S., 49
Main program and subroutines, 240
Mapping between object models, 194
Marshalling, 170, 173
Massengill, D., 199, 310, 324
Master-slave architecture, 242
Mattson, M., 199, 310, 324, 371
McDermid, J. A., 33
Mediator, 276
Medvidovic, N., 330
Message passing interface (MPI), 316
communication, 317
program structure, 317
topologies, 318
Message passing systems, 315ff
synchronous versus asynchronous, 315
Message transformation function, 275
Meyer, B., 25, 92ff, 185, 217
Meijers, M., 46
Michael, G., 4
Microsoft
.NET, 162, 395
COM/DCOM, 162ff, 187
Foundation Classes, 99
transaction server, 144
Visual Basic, 162
Windows Design Guide, 72
Middleware, 249ff
Model-view-controller (MVC), 369, 375, 380
Monikers, 172
types, 172
Multiprogramming system, 302
Netscape Navigator, 97
Mowbray, T., 86
Newell, A., 46
Noble, J., 83ff
Norman, D., 72
Obendorf, P., 217
Object fusion, 59
Object, Linking, and Embedding
(OLE)(Microsoft), 37, 162, 167ff
Object-oriented programming, 24–26, 343
Object-oriented programming language (OOPL),
99, 151
Object-oriented systems, 250ff
core concepts, 251
database systems, 288
delegation, 251
distributed objects, 253
inheritance, 251
Object Request Broker (ORB), 147ff
Observable, Java class, 69
Observer pattern, 66, 130
Open database connectivity (ODBC), 168
OLE, see Object, Linking, and Embedding
OOPL, see Object-oriented programming
language
Opdyke, W. F., 405
OpenDoc, 99, 154ff
building a component, 157
part hierarchy, 157
Open systems, 217
Oracle, 288
Papaconstantinou, M., 59
Paradigm, 2–3
Parallel programming, 15, 306ff
choosing a model, 324
data decomposition, 325
embarrassingly, 308
implementing a parallel algorithm, 325
methodology, 322ff
motivation, 311, 323
problem description, 324
selecting an algorithm structure, 326
task decompositon, 324
Parallel random access machine (PRAM), 15
Parallel virtual machine (PVM), 319ff
task communication, 320
INDEX

Paralysis through analysis, 87
Pattern design, 90
Patterns
action-related (Tidwell), 75
anchor (Coplien), 81
antipatterns (Mowbray), 86
as building blocks, 91
content-related (Tidwell), 75
design principles, 92
object-oriented (Noble), 83
object relationship (Noble), 84
Web-design (Welie), 75
Web (Tidwell), 77
Pattern languages, 30
human-computer interface, 73
Pattern Languages of Program (PloP), 41
Pattern space, 42
Perlis, A., 84
Perry, D., 199
Pipe and filter architecture, 228ff
assessment, 237
cyclic configurations, 235
design principles, 92
equations, 229
flow control, 235
model, 230
multiple inputs/multiple outputs, 235
ordering of filters, 235
PloP, see Pattern Languages of Program
POOMA, 402
POAM, see Parallel random access machine
Prieto-Diaz, R., 7
Problem class, 8
Problem paradigm, 13
Procedural programming, 23
Process, 301, 304
interaction, 302
Program, 13
reactive, 16
transformational, 16
Programming, 11
applicative, 22
concurrent, 15
object-oriented, 24
parallel, 15
procedural, 23
sequential, 14
Programming language(s), 4
choosing, 28
declarative, 22
functional, 23–24
imperative, 21–22
logic, 26
object-oriented, 24–26
understanding, 27
Prolog, 26
Protocol interoperability, 192
Pugh, J. R., 378, 381
Putnam, J. R., 140
QuickTime, 130
Raj, G. S., 65
Rechtin, E., 81, 127, 204
Reference architecture, 209, 213
Reference requirements, 209, 213
Registration mechanism, 274
Reusability, 4, 94
guidelines, components, 124
identifying components, 127
metrics, 125
Remote object proxy (COM), 170
Riehle, D., 44
Rising, L., 75
Robson, D., 378
Rombach, H. D., 7
Routing mechanism, 274
Rule(s), 263, 289ff
actions, 291
conditions, 290
execution model, 291
Rule-based systems, 263
backward chaining system, 266
forward chaining system, 265
operation, 264
types, 264
Russo, V. F., 346, 405
Sammet, J., 4, 23
Scheifler, R.W., 67, 245
San Francisco Project, 398ff
benefits, 401
Common Business Objects, 400
Core Business Processes, 400
disadvantages, 401
Foundation layer, 398
Sandarae, R., 319
Schmucker, K., 35
Schneider, F. B., 199
Search, 17
Schmidt, D., 47, 149
Schmucker, K. J., 381
Semantic interoperability, 191
Sequential programming, 14
Server, 243
designing, 249
stateless, 248
Shaw, M., 34, 199, 228, 233, 241
Shull, F., 46, 348
INDEX

Simon, H., 42, 46
Singleton, 49
describing, 50
implementing in C++, 51
implementing in Java, 53
issues, 54
Site navigation, 77
Size the schedule (Coplien), 81
Smalltalk-80, 35, 41, 374, 375ff
virtual machine, 262, 379
Smith, J., 46
SML, 24
Software
component, 100
component-based, 31
Software architecture, 3, 33, 197ff
analysis, 331, 336
architectural description, 203
architectural elements, 200
architectural styles, 203
benefits, 216
call-and-return systems, 240ff
 concurrent, 300ff
configuration, 201
cost, 218
data-centric systems, 286ff
description, 328
event-based systems, 272ff
extensibility, 334
flexibility, 334
hierarchically layered systems, 254ff
independent component systems, 270ff
interpreters, 258
monolithic middleware, 334
motivation for, 204
object-oriented systems, 250ff
organizational use, 205
performance analysis, 339
pipe and filter, 228ff
reflective, 335
roles, 214
rule-based systems, 263
scalability, 332
structural analysis, 337
systems engineering properties, 338
technical use, 205
transparency, 332
virtual machines, 258
Software architecture description languages, 34, 329
Software engineering, 206, 207
Software Engineering Institute (SEI), 97, 217
Software invisibility, 204
Software programming, 207
Sorting, 19
Squeak, 379
SQL Server 2000, 288
StandUp meeting (Coplien), 82
Static binding, 120
Stotts, D., 46
Structural similarity, 91
Sybase, 288
Syntactic interoperability, 190
System development life cycle (SDLC), 214
System object model (SOM), 151ff, 187
communication models, 153
Systolic arrays, 225
Szyperski, C., 30, 100
Taligent Corp., 345, 349
CommonPoint framework, 384
framework, 384
model, 384
Thread(s), 301, 303ff
models, 304
types, 305
Tichy, W. F., 43
Tidwell, J., 72
Tracz, W., 199
Transactions
concepts, 144
processing, 144
properties, 144
Transaction flow, 222
Transform flow, 222
Travassos, G., 347
Unix, Shell, 233
User interface, 11
Views, 376
Virtual machines, 258ff
early VM systems, 259
IBMs VM/SP, 260ff
Java VM, 262
Smalltalk-80 VM, 262
Vlissides, J., 65, 380
Von Neumann machine, 21, 23
Wallnau, K. C., 217
WARP, 233
Warren, H., 18
Web services, 133ff
Weinand, A., 35, 386
Welie, M., 75
Wills, A.C., 103
Wilson, D.A., 381
Wirth, N., 25
Wolf, A.L., 199
Wrapper, 54
combination, 55, 57
data, 58
issues, 60
object, 54–55
principles for, 60
procedural, 55–56

simple, 57
types of, 54
XML, 133
Xu, F., 360
X Windows system, 67, 245
Zullighoven, H., 44