INDEX

One of the great defects of English books printed in the last century is the want of an index.

—Lafcadio Hearn a.k.a Koizumi Yakumo

Abstract syntax tree (AST), 157, 318, 358
Abstraction, 135–136, 142, 149
Accelerated development, 327
Acceptance
test, 98–99
testing, 153
Accidental reuse, 358
Accounting, 116
Activity, 40, 358
Activity definition, 95
Activity-based classification, 28
Actual impact set (AIS), 225–226, 358
Ad hoc reuse, 338
Ada SDA (System Dependency Analyzer), 171
Adaptability, 16, 326
Adaptive, 1, 30–31, 289
Adaptive maintenance, 27, 358
Adequacy, 227, 359

Adherence, 227–228, 274, 359
Age, 39
Agglomerative algorithms, 281
Agile
programming, 58
software development, 359
software methodology, 13
Aging, 53, 375
Aiken, Peter, 11
Albrecht, A. J., 162
Alteration, 10, 137–138, 140
Amplification, 227
Analysis
knowledge, 317
and planning, 150–151
of protocol data, 315–316
state, 123
Annotation, 164, 300
Annotation of the program, 359

Priyadarshi Tripathy and Kshirasagar Naik.
© 2015 John Wiley & Sons, Inc. Published 2015 by John Wiley & Sons, Inc.
INDEX

Anomaly, 60
Antiregression, 35, 58, 359
Antithesis, 54
Application
domain, 39
engineering (AE), 330–331, 348, 359
gateway, 206, 359
program interface (API), 189
wrappers, 191
Approval, 104
Approval task steps, 106
Architectural reuse, 15, 326
Architecture description languages (ADLs), 330
Architecture extraction, 135
Arnold, Robert S., 176
Arthur, Jay Lowell, 2, 45
Artifacts, 12, 38, 42, 133, 258, 359
Aspect, 360
Aspect-oriented software development, 155, 360
Assembler, 165, 360
Assertion, 164, 265, 360
Assessment model, 336
Asset, 360
Assimilation process, 15, 301
Association coefficient, 280
Assumptions management, 56
Audit, 116
Auditing, 116, 118
Automated analysis, 257
Available support community, 68
Backus–Naur Form (BNF), 157
Backward
search, 235
slice, 159
wrappers (b-wrappers), 189, 191, 360
Bad smell, 360
Baseline, 37, 118, 360
Basic conceptualization, 170
Basic reuse, 338–339
Bazaar, 58
Beacon, 296, 360
Behavior preserving transformation, 266
Belady, Laszlo, 1
Bennett, Keith H., 45
Behavioral dependencies, 69
Big bang, 360
Big Bang approach, 144, 202
Black-box
components, 5
reengineering, 11, 189
requirements, 346
testing, 360
Bottom-up, 301
Branch, 114
Brooks model, 303
Build process, 70
Building, 115
Business model of Poulin and Caruso, 350
Business
process reengineering (BPR), 199
rules, 159
Butterfly, 12, 360
Butterfly methodology, 208
Byrne, Eric J., 138
Call graph, 13, 158, 234–235, 317, 360
Call graph-based analysis, 13, 224
Call preservation, 262
Call, use and define (CUD), 245
Candidate impact set (CIS), 225–226, 231, 360
Canning R. G., 1, 25
Capability Maturity Model (CMM), 64
Carry on maintenance, 11, 188
Cathedral, 58
Causation, 149
Centroid, 281
Certification, 149, 329
Change
control, 42, 361
coupling, 361
management, 115, 361
mini-cycle model, 7, 91
propagation, 13, 245, 265
propagation activity, 94, 225, 361
propagation heuristics, 243, 247
propagation model, 242–243
rate, 228
request (CR), 26, 43, 92, 116, 119, 215, 223, 225, 243
request schema, 121
set, 243–244
source instruction (CSI), 350
system, 42
<table>
<thead>
<tr>
<th>Term</th>
<th>Page Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapin, Ned, 28–29, 45</td>
<td></td>
</tr>
<tr>
<td>Chicken little, 12, 361</td>
<td></td>
</tr>
<tr>
<td>Chicken little migration, 207</td>
<td></td>
</tr>
<tr>
<td>Chicken little strategy, 206</td>
<td></td>
</tr>
<tr>
<td>Chidamber and Kemerer (CK) metrics, 162, 164</td>
<td></td>
</tr>
<tr>
<td>Chikofsky and Cross II, 153–154</td>
<td></td>
</tr>
<tr>
<td>Chrysaliser, 208, 210–212</td>
<td></td>
</tr>
<tr>
<td>Chunk, 293, 361</td>
<td></td>
</tr>
<tr>
<td>Chunking, 15, 295, 361</td>
<td></td>
</tr>
<tr>
<td>City metaphor, 320</td>
<td></td>
</tr>
<tr>
<td>CK metric suite, 162</td>
<td></td>
</tr>
<tr>
<td>Class diagram, 260, 262, 265–266</td>
<td></td>
</tr>
<tr>
<td>Cleansing, 195</td>
<td></td>
</tr>
<tr>
<td>Clearcase, 115, 118</td>
<td></td>
</tr>
<tr>
<td>Cliphé, 171, 361</td>
<td></td>
</tr>
<tr>
<td>Client organization, 43</td>
<td></td>
</tr>
<tr>
<td>Clone, 61, 164, 177, 375</td>
<td></td>
</tr>
<tr>
<td>Closed source software (CSS), 3, 49, 58, 87, 361</td>
<td></td>
</tr>
<tr>
<td>Closed state, 125</td>
<td></td>
</tr>
<tr>
<td>Closedown, 88–89</td>
<td></td>
</tr>
<tr>
<td>Clustering, 278, 280</td>
<td></td>
</tr>
<tr>
<td>Clusters</td>
<td></td>
</tr>
<tr>
<td>business rules, 29, 31, 36</td>
<td></td>
</tr>
<tr>
<td>documentation, 29, 31, 34</td>
<td></td>
</tr>
<tr>
<td>software properties, 29, 31, 35</td>
<td></td>
</tr>
<tr>
<td>support interface, 29, 31, 34</td>
<td></td>
</tr>
<tr>
<td>COCOMO model, 199</td>
<td></td>
</tr>
<tr>
<td>Code</td>
<td></td>
</tr>
<tr>
<td>changes, 275–276</td>
<td></td>
</tr>
<tr>
<td>churn, 361</td>
<td></td>
</tr>
<tr>
<td>cognition, 291</td>
<td></td>
</tr>
<tr>
<td>decay, 53–54, 361</td>
<td></td>
</tr>
<tr>
<td>layout (FIL), 246</td>
<td></td>
</tr>
<tr>
<td>leverage, 289</td>
<td></td>
</tr>
<tr>
<td>readers, 297</td>
<td></td>
</tr>
<tr>
<td>smell, 258, 361</td>
<td></td>
</tr>
<tr>
<td>CodeCrawler, 171</td>
<td></td>
</tr>
<tr>
<td>Cognitions, 317</td>
<td></td>
</tr>
<tr>
<td>Cohesion, 263</td>
<td></td>
</tr>
<tr>
<td>Cohesion metric, 267–268</td>
<td></td>
</tr>
<tr>
<td>Cold turkey, 12, 201, 361</td>
<td></td>
</tr>
<tr>
<td>Commercial off-the-shelf (COTS), 5–6, 36, 62, 361</td>
<td></td>
</tr>
<tr>
<td>Commit state, 123</td>
<td></td>
</tr>
<tr>
<td>Common goals, 44</td>
<td></td>
</tr>
<tr>
<td>Compilation, 141, 152, 165, 361</td>
<td></td>
</tr>
<tr>
<td>Complex refactoring, 14</td>
<td></td>
</tr>
<tr>
<td>Complexity, 263</td>
<td></td>
</tr>
<tr>
<td>Complexity metrics, 162</td>
<td></td>
</tr>
<tr>
<td>Compliance, 327</td>
<td></td>
</tr>
<tr>
<td>Component collaboration, 68</td>
<td></td>
</tr>
<tr>
<td>reconfiguration, 65</td>
<td></td>
</tr>
<tr>
<td>reuse, 167</td>
<td></td>
</tr>
<tr>
<td>selection, 67</td>
<td></td>
</tr>
<tr>
<td>Component-based software (CBS), 5–6, 62</td>
<td></td>
</tr>
<tr>
<td>Composite database, 12, 362</td>
<td></td>
</tr>
<tr>
<td>Composite database approach, 205</td>
<td></td>
</tr>
<tr>
<td>Comprehension strategies, 308</td>
<td></td>
</tr>
<tr>
<td>Computationally redundant data, 213</td>
<td></td>
</tr>
<tr>
<td>Computer-aided software engineering (CASE), 40</td>
<td></td>
</tr>
<tr>
<td>Concept assignment problem, 228</td>
<td></td>
</tr>
<tr>
<td>Concepts, 93, 228, 362</td>
<td></td>
</tr>
<tr>
<td>Conceptual data, 213</td>
<td></td>
</tr>
<tr>
<td>level, 136</td>
<td></td>
</tr>
<tr>
<td>normalization, 170</td>
<td></td>
</tr>
<tr>
<td>phase, 168</td>
<td></td>
</tr>
<tr>
<td>schema, 167, 195, 362</td>
<td></td>
</tr>
<tr>
<td>Concurrent version system (CVS), 118</td>
<td></td>
</tr>
<tr>
<td>Configuration, 362</td>
<td></td>
</tr>
<tr>
<td>baseline, 117</td>
<td></td>
</tr>
<tr>
<td>control board (CCB), 118</td>
<td></td>
</tr>
<tr>
<td>item, 362</td>
<td></td>
</tr>
<tr>
<td>management (CM), 9, 42, 66–67, 103, 111, 148, 362</td>
<td></td>
</tr>
<tr>
<td>Conformance, 150</td>
<td></td>
</tr>
<tr>
<td>Conjectures, 14, 295, 299</td>
<td></td>
</tr>
<tr>
<td>Connectivity matrix, 232</td>
<td></td>
</tr>
<tr>
<td>Conservation of familiarity, 3, 39, 50, 52, 60, 242</td>
<td></td>
</tr>
<tr>
<td>Conservation of organizational stability, 3, 50, 52, 60, 242</td>
<td></td>
</tr>
<tr>
<td>Construction, 362</td>
<td></td>
</tr>
<tr>
<td>Construction algorithms, 280</td>
<td></td>
</tr>
<tr>
<td>Constructive Cost Model II (COCOMO II), 202</td>
<td></td>
</tr>
<tr>
<td>Consultive, 30–31</td>
<td></td>
</tr>
<tr>
<td>Context area, 318</td>
<td></td>
</tr>
<tr>
<td>Continuing change, 3, 50–51, 134, 242</td>
<td></td>
</tr>
<tr>
<td>Continuing growth, 3, 50, 52, 134, 242</td>
<td></td>
</tr>
<tr>
<td>Contours, 320</td>
<td></td>
</tr>
<tr>
<td>Contracting, 12</td>
<td></td>
</tr>
</tbody>
</table>
INDEX

Control
data, 213
dependency edges, 236
flow, 292, 312

Control flow
analysis (CFA), 10, 362
graph (CFG), 142, 158, 317

Control structure diagram (CSD), 318

Controlled interfaces, 69
Controlling, 118
Corrections, 28
Corrective, 1, 30–31, 289
Corrective maintenance, 26, 362

Cost estimation, 12
Cost of Gaffney and Durek, 340
Cost per error, 350
Cost-benefit analysis, 12, 200
Coupling, 54, 263
Coupling between object class (CBO), 163
Coupling metrics, 268

Critical pair analysis, 261
Crosscutting concerns, 362
Cross-referencing, 15, 295–296, 362
Customer, 362
Cut over, 196, 362
Cut-and-run, 196

Cyclomatic complexity, 162–163, 197–198, 241

Dangling purpose, 301
Data
access-allocator (DAA), 208, 210–212
administration, 166, 362
analysis, 169
archival, 109, 111
assets, 166
banker, 214
cleaning, 195
conversion, 167, 195, 362
dependency edges, 235–236
description language (DDL), 168
flow, 292, 312
flow analysis (DFA), 10, 158, 363
flow graph (DFG), 317
forward engineering, 166
locator, 214
management system (DMS), 168
reduction, 148
reengineering, 152
representation, 148
reuse, 15, 326
reverse engineering (DRE), 11, 166, 363
structure conceptualization, 168
structure extraction, 168

Database, 166
Database
first, 12, 189, 203
gateway, 206, 362
last, 12, 204
management system (DBMS), 167–168
reverse engineering (DBRE), 167, 195, 363
wrappers, 189

Data-oriented application, 165–167
Data-Transformer, 208, 212
Davis, Ted, 336
Decay, 363
Decision tree, 32
Decision tree-based criteria, 31
Declining quality, 3, 50, 53, 134, 242
Decomilation, 135, 141, 165, 363
Decompiler, 152, 164
Definition-use, 317
Degree of interest (DOI), 318
Delivery, 99
Density search technique, 280–281
Dependency analysis, 12
Dependency graph, 13, 229, 234
Dependency graph-based analysis, 13, 234
Depth of inheritance tree (DIT), 163

Design, 96
Design
level, 136
recovery, 154, 363
reuse, 16, 326
clones, 164

Developer, 2
Developer information (DEV), 246
Development
cost avoidance, 351
model, 363
process, 111, 363
replay (DR), 246
for reuse, 325, 328, 329
with reuse, 325, 330
staff, 44
technology, 41, 363
Device-drivers, 61
Direct impact, 231
Disassembler, 164, 363
Discard and redevelop, 11, 188
Discovered impact set (DIS), 225–226, 363
Distance measure, 280
Distance-based, 268
Distance-based approach, 233
Distinctive appearance, 160
Divide-and-conquer schema, 297–298
Divisive algorithms, 281
DMS (Design Maintenance System), 171
DMS–DDL text analysis, 169
Documentation, 104, 146, 166, 274 change, 94
task steps, 105
Documenting, 118
Domain, 325, 363
analysis, 15, 325, 329, 36
design, 330, 364
engineering (DE), 328–329, 331, 348, 364
implementation, 330, 364
knowledge, 303, 304, 308, 314
model, 313, 331, 364
plans, 294
qualification (DQ), 331
specific languages (DSLs), 364
Domain analysis and reuse environment (DARE), 331
Draco, 331
Duplicate code, 259
Dynamic
call graph, 158
elements, 295
growth, 44
program slice, 225, 237–238
Economic, 328
Effective size, 346
Effective visual metaphors, 162
Effectiveness, 227, 363
Eighth law, 50, 53
Element connectedness, 320
Elimination-of-goto, 276
Embedded knowledge, 53
Empirical studies, 54, 60
Encapsulation, 146–148
Encoding, 316
Engineering change (EC), 112, 124
Engineering view point, 46
Enhancements, 28
Enhancive, 30–31
Enterprise JavaBeans (EJB), 201
Entity-relationship (ER), 167
Entropy, 53, 375
Environment, 275
Environmental independence, 326
Equivalence tests, 217, 364
Error flow analysis, 238
Establishing baseline, 118
Estimated impact set (EIS), 93
E-type program, 48, 51, 134, 364
E-type software, 56
E-type systems, 3
Euclidean distance, 268
Evaluation, 317
Evaluative, 30–31
Event management, 42, 364
Evidence-based classification, 28
Evolution, 1, 7, 44–45, 87–88
Evolution of E-type software, 49
management, 57
process, 49, 51, 53, 89, 364
of software, 44
Evolutionary approach, 146
model, 92
trajectory, 51
Evolves, 2
Exception handling, 292
Execution history, 237
Explanatory (what/why), 45
Extensibility, 256, 263
External documentation, 307
interface, 192
representation, 300
reuse (a.k.a. public reuse), 364
software value, 256
External traceability, see Horizontal traceability
Extract, transform, and load (ETL), 195
Extractive, 327–328
Extreme programming (XP), 13, 365
Facebook, 330
Failure handling, 70
False positive, 233
False positive impact set (FPIS), 225–226, 365
Familiarity, 320
Fan-in, 162–163
Fan-out, 162–163
Family-oriented abstraction, specification, and translation (FAST), 331
Feature location, 229
Feature-oriented domain analysis (FODA), 330
Feature-oriented reuse method (FORM), 331
Features, 228
Feedback, 365
Feedback system, 3, 49, 50, 53, 242
Feedback, evolution, and software technology (FEAST), 54
FermaT, 172
Fifth law, 50, 52
First law, 50–51
First order reuse cost model, 346
Fisheye view, 318
Focus area, 318
Forward
  engineering, 10, 135, 139, 154, 365
gateway, 203, 205–206, 365
migration, 203, 365
searching, 235
slicing, 159, 225
software engineering principles, 210
wrappers (f-wrappers), 189, 365
FOSS evolution, 60
Fourth law, 50, 52
Framework, 365
Free and Open Source Software (FOSS), 4, 58, 365
Free software, 4, 58
Free software, 58
Free/Libre/Open Source Software (FLOSS), 58, 90
Freeze, 11, 188
Frequency, 246
Friendly user interface, 162
Frozen, 209
Frozen functionality, 63
Full reuse model, 85, 87
Fully decomposable, 204–205, 365
Function, 146, 312
Function
  name, 297
  point (FP), 162, 163
  wrappers, 191
Functional
  knowledge, 314–315
  requirements, 256
  enhancement, 134
Functionality, 292
Functionalization, 149
Gate keepers, 59
Gateway, 203, 365
Gateway co-ordinators, 208
General knowledge, 291
Generation of
  code, 149
  design, 149
Generic knowledge structure, 294
Geographic technique, 280–281
Gestalt’s theory, 319
Gibson’s theory, 319
Global
  factors, 60
  variable (GV), 163
Glue, 62–63, 365
Goals, 155, 164, 291, 299
Goals/models/tools, 10, 155
Good figure, 319
Good use of interactions, 162
Goto-less, 276
Granularity, 191
Graph exchange language (GXL), 172, 366
Graph rewriting, see Graph transformation
Graph theoretical algorithm, 280
Graph transformation, 265–266, 366
Graphical user interface (GUI), 194
Grep, 228
Groomative, 30–31
Groups, 29
Guru, 243
Habitability, 320
Halpern, Mark I., 1, 44
Halstead complexity (HC), 163
Halstead, Maurice, 164
Hardware maintenance, 2
INDEX 385

Hazard, 28, 366
Hex-Rays Decompiler, 172
Hierarchical algorithms, 281
High cohesion, 16, 326
High information content, 160–161
High level of visibility, 70
Higher order reuse cost model, 346, 348
Historical co-change (HIS), 245
Horizontal reuse, 366
Horizontal traceability, 224, 229–230, 366
Horseshoe, 366
Horseshoe model, 138
human resource, 43
Hypertext markup language (HTML), 194
Hypotheses, 14, 293–295, 299, 305, 308, 366
Hypotheses strategies, 14
I/O-emulator, 194
Iceberg, 1, 92
IDA Pro Disassembler and Debugger, 172
Identification, 113
IEEE 1042, 9
IEEE/EIA 1219, 8, 94, 99
Imagix 4D, 172
Immediacy, 317
Impact analysis (IA), 12, 93, 223, 225, 366
Implement
change, 94
parallel operations and training, 110
state, 123
Implementation, 97
details, 293
level, 136
model, 337
Improvement algorithm, 281
Improving maintainability, 134
Improving quality, 134
Incompatibility of upgrades, 63
Increasing complexity, 3, 50–51, 134, 242
Incremental
approach, 145, 233
development, 86, 366
Indirect impact, 231
Individuality, 160
Information system (IS), 187, 366
Information system gateway, 206–207, 366
Information-flow metric, 164
Initial development, 7, 87–88
Initial entity, 244
Initial reuse, 338, 339
Input, 95
Inspection, 276
Integrated, 338
Integrated metamodel, 312, 314
Integration, 166
Intention-based classification, 26
Intentions, 317
Inter-class coupling, 269
Interface converter, 193
Interface description languages (IDLs), 330
Intermodule change propagation, 240
Internal
documentation, 306
interface, 193
reuse, 366
semantic, 302
software value, 256
traceability, see Vertical traceability
Internet Protocol (IP), 39
Interpretation, 149
Interprocedural analysis, 158
Intra-class coupling, 269
Intramodule change propagation, 238
Intraprocedural analysis, 158
Invariants, 265
Inverse square law, 54–56
Investigation activity, 40, 366
Investigation report, 42
IRAP (Input–Output Reengineering and
Program Crafting), 172
is-a, 294, 315
ISO/IEC 12207, 8, 107
ISO/IEC 14764, 8, 25–26, 99
Isolation, 115, 148
Iterative, 12
algorithm, 281
approach, 145
development, 86, 366
enhancement model, 85–86
method, 212
models, 7
process, 99
JAD (JAva Decompiler), 173
Java 2 Enterprise Edition (J2EE), 201
Jazayeri, Mehdi, 2
INDEX

Kaizen, 86
kind-of, 294, 315
Kitchenham, A, 28
Knowledge, 291, 301–302
   acquisition, 166, 291
   base, 15, 298, 314, 366
   elements, 294, 367
   structure, 297, 309, 315, 367
Koala, 332
KobrA, 332

Lack of cohesion in methods (LCOM), 163

Law of
   connectedness, 320
   continuation, 320
   contour, 320
   evolution, 3
   familiarity, 320
   good figure, 320
   orientation, 320
   proximity, 320
   similarity, 320
   software evolution, 49
   symmetry, 320

Legacy, 187

Legacy
   DB, 214
   DB manager, 214
   information systems (LIS), 140, 188, 367
   SampleData, 208
   software, 11
   state, 213
   system, 11, 89, 187–188, 192, 195, 203, 367

Legal, 328
Lehman, Meir M., 1, 3, 44
Lehman’s law, 3, 51, 212
Letovsky model, 298
Leveraged, 338
Lexical analysis, 10, 157, 367
Library, 118
License agreements, 165
Lines of code (LOC), 163
Linus’s law, 59
Linux, 4, 59–60
LIS migration, 195–196
Local area network (LAN) simulator, 260
Locality, 321

Logical
   phase, 168
   schema, 167–168, 195, 367
Long-term memory a.k.a. knowledge base, 315
Low coupling, 16, 326
Low visual complexity, 160–161

Macrostructure, 311
Magnification function, 319
Maintainability, 256, 263, 327
Maintained
   product, 37–38
   product dimension, 37
Maintainer, 2, 291, 367
Maintenance, 1, 4, 25, 45
   activities, 26
   activities-based classification, 4
   adaptive, 4
   corrective, 4
   cost, 327
   of COTS-based systems, 5–6
   engineer, 2
   event, 42, 367
   evidence-based classification, 5
   factors, 37
   iceberg, 1, 25
   intention-based classification, 4
   management, 42, 367
   model, 84, 367
   organization, 41–43
   organization structure, 42, 367
   perfective, 4
   plan, 99, 101–102
   preventive, 4
   process, 37–38, 95, 100, 367
   request, see Change request (CR)
   review, 100, 105
   staff, 44
Make, 112, 115
Management activity, 40, 367
Managerial, 328
Managers and policies, 274
ManSART, 173
Marr’s theory, 319
McCabe IQ, 173
McIlroy, Dough, 15
Mean time to maintenance request (MTMR), 215
INDEX 387

Mediator, 68
Medium access control (MAC), 291
Mental model, 14, 291, 293, 367
Mental representation, 300
Merging, 114
Message
chain, 259
handler, 193
oriented middleware (MOM), 201
Metadata, 214–215, 217
Method, 41, 367
Metric, 95, 265, 267, 368
Metrication, 146
Microstructure, 311
Middleware, 64–65
Migrate, 12, 188
Migrate data, 216
Migrating to a new technology, 134
Migration, 12, 100, 106, 140–141, 368
of legacy information system, 140, 368
of LIS, 195
plan, 107
plan task steps, 108
planning, 196
process, 215
projects, 12
standard, 107
build, 70
Minimal component coupling, 70
Mishap, 28, 368
Mission-critical
legacy system, 201
project, 196
Model, 156, 368
Model-driven engineering, 140, 368
Modification activity, 40, 368
Modification
implementation, 100, 105
of specification, 149
request (MR), 95, 102, 119, 368
Modification request
analysis, 103
task steps, 104
Modularity, 256
Module
connection analysis, 238
level, 192
wrapper, 194
Monolithic systems, 4
Multi-agent, 53
Multi-level, 53
Multi-loop, 53
Neighbors, Jim, 15, 325
New DB, 214
New DB manager, 214
New state, 213
Nondecomposable, 205–206
Nondecomposable information system, 204, 368
Non-functional requirements, 256, 368
Normalization, 141, 148, 368
Notification
of completion, 108, 110
intent, 107
of intent, 110
Number of children (NOC), 163
Obfuscation, 369
Object
identification, 149
interpretation, 149
Objectives, 134
Observable behavior, 13
Occurred set, 244
Off-the-shelf component, 329
Onion model, 58–59
Open source community (OSC), 58
Open source software (OSS), 368
Openness of components, 67
Operations, 312
Opportunistic, 301, 337
Optimization, 141, 369
Optimization algorithm, 281
Option task steps, 104
Options, 104
Organized reuse, 338–339
Orientation, 320
Original equipment manufacturer (OEM), 67
OS/360 operating system, 54
Output, 95
Outsource, 11, 188
Overloading, 195
Paradigm, 41, 369
Parallel operation, 196, 108
Parameter coupling, 263–264, 269–270
Parameterize methods, 273
Paranas, David, 15, 325
Parse tree, 157
Parsing, 148
Partial approach, 145
Partially persistence, 166
Path-based dynamic impact analysis, 235
PBS (Portable Bookshelf), 173
Pennington model, 310
Peopleware, 43
Perceptional organization, 319
Segregation, 319
Theory, 319
Perfective, 1, 289, 327
Perfective maintenance, 27, 49, 369
Modification, 45
Performance, 30, 31, 263
Permanent files, 166
Persistent data structure, 166
Phase model, 150
Phased interoperability, 196
Phaseout, 8, 88–89, 187
Physical phase, 168
Physical schema, 168, 195, 369
Plan, 171, 369
Plan change, 93
Recognition, 171
Planning, 117, 317
Plans, 164, 293–294, 299, 315
PLUS, 332
Pollution, 53
Poor lexicon, 54
Portability, 15, 326
Portfolio analysis, 12, 197, 369
Postconditions, 265
Post-delivery activities, 26
Post-operation review, 109
Practices, 276
Precision, 226–227, 243–244, 246–247, 369
Preconditions, 265
Pre-delivery activities, 26
Predicted set, 244
Presence, 317
Pretty printing, 257
Preventive, 30–31
Preventive maintenance, 27, 51, 137, 369
Primary data, 212
Primitive refactoring, 14
Principle of abstraction, 135, 369
Alteration, 137, 142, 369
Refinement, 135, 370
Priority, 122
Proactive, 327
Probabilistic measures, 280
Problem domain, 299
Domain knowledge, 311, 314
Identification, 95
Procedural granularity, 148
Level, 192
Procedure, 41, 370
Procedure wrapper, 194
Process, 59, 144, 370
Process improvement (how), 45
Information, 246
Level, 191
Metrics, 230
Model, 92, 144, 370
Product, 37, 370
Component, 39
Development, 330
Line development, 328–329
Line, see Domain implementation
Metrics, 230
Quality, 40
Reconfiguration, 68
Upgrade, 37, 370
Productivity, 327
Program analysis, 169, 370
Analysis tool (PAT), 317
Comprehension, 14, 93, 165, 291, 309–310, 370
Conversion, 196
Dependency, 224
Dependency graph (PDG), 235, 370
Documentation, 94
Domain knowledge, 315
Families, 15, 325, 370
Fragments, 294
Graph, 266

INDEX

knowledge, 317
level, 192
metrics, 162
migration, 152, 165, 370
model, 311, 313–314, 315
model structure, 314
plans, 171
reading, 149
rephrasing, see Rephrasing
restructuring, see Restructuring
reuse, 16, 326
slices, 10
slicing, 158, 282, 371
translation, 141, 371
understanding, see Program comprehension
wrapper, 194
Programmers, 274
Programming
  environment, 274
  knowledge, 302, 308
  plans, 294, 308, 315
  rules of discourse, 308
Project justification, 12
Prolog, 262
Properly nested programs, 277
Proprietary software, 49
Protocol
  analysis, 315, 371
  data, 315–316, 371
Proximity, 319
Pruning, 246
P-type program, 47, 371
PuLSE, 332
Push down method, 267, 272
Quality assessment, 167
Quality assurance activity, 40, 371
Query the tool first, 245
Quick fix model, 85
Random, 246
Rationalization of control flow, 148
Raymond, Eric S., 58–59
Reachability
  graphs, 232
  matrix, 233
Reactive, 327–328
Read coupling, 163
Readability, 256
Real-world knowledge, see Domain knowledge
RE-Analyzer, 173
Recall, 226–227, 243–244, 246–247, 371
Recency, 246
Reclamation, 371
Recode, 140, 143, 152, 371
Reconstitute, 166
Redesign, 141, 143, 371
Redocumentation, 152–153, 372
Redocumentation of artifacts, 154
Reductive, 30–31
Redundant structural data, 213
Reengineered components, 214
Reengineered state, 213
Reengineering, 9–10, 133, 138–139, 154, 372
Reengineering assistant (RA), 174
  process, 144
Refactor, 258
Refactoring, 13, 94, 141, 256, 259, 372
Refinement, 135–136, 142
Reflexive property, 240
Reformative, 30–31
Regeneration, 149
Regression testing, 13, 94
Rehosting, 140, 372
Rejuvenation, 27, 372
Release management, 57
Releases, 60, 372
Reliability, 16, 255, 257, 326–327
Remote procedure calls (RPCs), 189
Renovation, 141, 150–151
Rephrasing, 140, 371
Replace, 141, 143, 372
Replace parameter with method, 272
Replace strategy, 142
Replacement strategies, 152
Repository, 118, 372
Repository database, 146
Representation, 160
Requirements level, 136
Residual
  data, 213
  DB, 214
  DB manager, 214
  Resilience to change, 161
| Resource, 372                        | Ripple effect, 13, 39, 93, 238, 374        |
| constraints, 262                    | analysis, 13, 225                           |
| dependencies, 69                    | for a program (REP), 242                    |
| Respecify, 141, 143, 372            | and stability tool (REST), 242             |
| Response for a class (RFC), 163     | Ripple effects measurement, 242            |
| Restored components, 214            | Ripple-sensitivity, 227, 374                |
| Restored state, 213                 | RiSE maturity model, 334, 338               |
| Restructuring, 13–14, 94, 137, 256–258, 372 | Risk, 327                                 |
| Restructuring algorithm, 142        | Risky, 57                                  |
| Rethink, 141, 143, 373              | Robustness, 263                             |
| Retirement, 100, 109, 376           | Roll over, 212                              |
| Retirement plan, 110                | Royce, Winston, 6, 86                       |
| Return value coupling, 263–264, 269 | RSEB, 332                                  |
| Reusability, 16, 263, 373           | Rules of discourse, 300, 315                |
| Reusable                             | programming discourse, 296, 374             |
| artifacts, 326                      | Safe, 57                                   |
| assets, 15, 330                      | Safety constraints, 262                    |
| components, 347–348                 | Sample DataStore, 208–210                  |
| software assets (RSA), 328–329      | Sandbox, 115                                |
| Reuse, 289, 373                     | Scalability, 263                            |
| assets, 346                          | Scalability of visual complexity, 161       |
| capability, 333–334, 373            | Scattering and tangling, 374                |
| capability model (RCM), 334–336     | Schema(s), 293–294, 308, 315, 374          |
| cost avoidance (RCA), 351            | analysis, 166                               |
| effectiveness, 333–334, 373          | conversion, 195                             |
| efficiency, 333, 373                 | de-optimization, 170                       |
| library, 327, 329, 346, 349, 373    | integration, 169                           |
| maturity model (RMM), 334, 373       | translation and redesign, 166              |
| opportunity, 333, 373                | untranslation, 170                         |
| oriented model, 84                   |                                           |
| percent (RP), 351                    | Scientific view point, 46                  |
| proficiency, 333, 373                | Screen scrapping, 194, 374                  |
| program, 329                         | Script, 41, 374                             |
| value added (RVA), 351               | Scrubbing, 195                              |
| Reused source instruction (RSI), 350 | Second law, 50–51                          |
| Reverse                              | SEELA, 174                                 |
| engineering, 10, 135, 139, 153–154, 16, 164, 373 | Self-regulation, 3, 50–51, 60, 242         |
| gateway, 204–206, 373                | Semantic                                    |
| migration, 204, 373                  | distance, 319                               |
| Review state, 122                    | knowledge, 301, 303                        |
| Review task steps, 106               | Semantically redundant data, 213            |
| Revision control system (RCS), 112   | Semidecomposable, 204–205                  |
| Rework, 141, 143, 373                | Semidecomposable information system, 204, 374 |
| Rework strategy, 142                 | Sequential dependency analysis, 261         |
| Rewrite, 141, 143, 373               | Service cost avoidance, 351                 |
| Rewrite strategy, 142                | Service-level agreement (SLA), 42–43, 374   |
| Rigi, 174                           |                                           |
Service-oriented architecture (SOA), 42, 375
Servicing, 8, 88–89
Seventh law, 50, 53
Severity, 122
Shared variable coupling, 263–264, 269–270
Sharpness, 227–228, 375
Shipped source instruction (SSI), 350
Shneiderman and Mayer Model, 301
Short-term memory, 301, 303
Side effect, 93, 154, 236, 375
Similarity, 319
Similarity metrics, 280
Simple navigation, 161
Single entity suggestions, 245
Situation model, 310–311, 314–315
Situation model structures, 314
Sixth law, 50, 52
Size, 38, 44, 263
Slot-filler, 294
Slot-type, 294
Slot fillers, 315
Slot types, 315
Sneed, Harry M., 12
Sockets, 189
Soft-goal graph, 264, 375
Software
agents, 53
aging, see Aging
architecture, 292
artifacts, 258
assessment, 167
asset, 326
certification laboratories (SCL), 64
clone, see Clone
configuration management (SCM), 9, 84, 111, 375
crisis, 15
development, 26
development cost, 350
development error rate, 350
development life cycle (SDLC), 83–84
entropy, see Entropy
evolution, 375
geriatrics, 39
lifecycle objects (SLOs), 231–232, 376
maintainability, 26
maintenance, 2–3, 6, 25–26, 83–84, 376
maintenance life cycle (SMLC), 7
maintenance standards, 8
metric, see Metric
migration, 376
production consortium (SPC), 336
quality, 53
reclamation, 376
reconnaissance, 228
reengineering, 9–10
rejuvenation, 27
restructuring, 256, 275–276
retirement, see Retirement
reuse, 15, 325–326
specific knowledge, 291
structure, 273
value, 256
visualization, 135, 360, 317
Soloway, Adelson, and Ehrlich model, 308
Source code
control system (SCCS), 112
reengineering reference model (SCORE/RM), 146
Source instruction reused by others (SIRBO), 350
Source-to-source translation, 152
SPE taxonomy, 46
Specification
of actions, 149
of constraints, 149
Speech sentences, 316
S-Ratio, 228
Stability
analysis, 238
measure, 238, 376
Stable storage, 292
Staged model, 7, 87, 187
Staged model for CSS, 87–88
for FLOSS, 90
Stakeholders, 117, 198–199, 376
Starting impact set (SIS), 93, 225, 228–231, 376
State, 312
Statecharts, 266
Static
call graph, 158
elements, 293
program slice, 236
### INDEX

<table>
<thead>
<tr>
<th>Static (continued)</th>
<th>Theory of notation, 319</th>
</tr>
</thead>
<tbody>
<tr>
<td>program slicing, 224</td>
<td>Think aloud protocol (TAP), 316</td>
</tr>
<tr>
<td>slice, 237</td>
<td>Thinking-aloud, 316, 376</td>
</tr>
<tr>
<td>Strategy, 295–296, 376</td>
<td>Third law, 50–51</td>
</tr>
<tr>
<td>Structural data, 213</td>
<td>Threshold value, 208–209, 212</td>
</tr>
<tr>
<td>S-type program, 46, 376</td>
<td>Tools, 156, 274, 276</td>
</tr>
<tr>
<td>Submit state, 120</td>
<td>Top-down, 301</td>
</tr>
<tr>
<td>Substitute algorithm, 271</td>
<td>Top-down model, 313, 315</td>
</tr>
<tr>
<td>Suitability for automation, 161–162</td>
<td>Top-down structure, 314</td>
</tr>
<tr>
<td>Support levels, 43</td>
<td>Torvalds, Linus Benedict, 59</td>
</tr>
<tr>
<td>Swanson, E. Burton, 1, 4</td>
<td>Traceability, 224, 229</td>
</tr>
<tr>
<td>Swanson’s classification, 4</td>
<td>Traceability analysis, 12</td>
</tr>
<tr>
<td>Symmetric suggestions, 244</td>
<td>Training, 30–31</td>
</tr>
<tr>
<td>Symmetry, 320</td>
<td>Training task steps, 108</td>
</tr>
<tr>
<td>Syntactic analysis, 10, 157, 376</td>
<td>Transaction level, 192</td>
</tr>
<tr>
<td>dependencies, 69</td>
<td>Transaction wrapper, 194</td>
</tr>
<tr>
<td>distance, 319</td>
<td>Transformation, 148</td>
</tr>
<tr>
<td>knowledge, 301–303</td>
<td>Transition net, 317</td>
</tr>
<tr>
<td>System monitoring, 66, 69</td>
<td>Transitive, 240</td>
</tr>
<tr>
<td>sandwich approach, 278</td>
<td>Transitive closure, 234</td>
</tr>
<tr>
<td>service wrappers, 191</td>
<td>Transmission control protocol (TCP), 292</td>
</tr>
<tr>
<td>test, 98</td>
<td>Trojan horse, 64</td>
</tr>
<tr>
<td>transition, 153</td>
<td>Trouble shooting, 68</td>
</tr>
<tr>
<td>Systematic reuse, 338, 340, 376</td>
<td>Trunk, 114</td>
</tr>
<tr>
<td>Tailorability of components, 68</td>
<td>Tuval Software Industries, 174</td>
</tr>
<tr>
<td>Tailoring, 62, 376</td>
<td>Two-phase commit protocol, 205</td>
</tr>
<tr>
<td>Target SampleData, 208</td>
<td>Type decision question, 33</td>
</tr>
<tr>
<td>Target system, 133–134, 142, 144, 208</td>
<td>Understandability, 16, 326</td>
</tr>
<tr>
<td>Target system testing, 150, 152</td>
<td>Unix operating system, 115</td>
</tr>
<tr>
<td>Team structure, 58</td>
<td>Unreliable COTS components, 64</td>
</tr>
<tr>
<td>Technical, 328</td>
<td>Unsafe, 57</td>
</tr>
<tr>
<td>Technical dimension, 113</td>
<td>Updative, 30–31</td>
</tr>
<tr>
<td>Technique, 41, 276, 376</td>
<td>Usability, 377</td>
</tr>
<tr>
<td>Temporal constraints, 262</td>
<td>User, 377</td>
</tr>
<tr>
<td>TempStore (TS), 208–210, 212</td>
<td>interaction, 319</td>
</tr>
<tr>
<td>Termination-condition, 208–209, 212</td>
<td>interface, 214, 216</td>
</tr>
<tr>
<td>Test generation, 148–149</td>
<td>manual, 94</td>
</tr>
<tr>
<td>Test Maturity Model (TMM), 64</td>
<td>request, 45</td>
</tr>
<tr>
<td>Test Process Improvement (TPI) model, 64</td>
<td>Validation and verification, 149</td>
</tr>
<tr>
<td>Test verdict, 124</td>
<td>Variability, 44</td>
</tr>
<tr>
<td>Testability, 263</td>
<td>Variable name, 297</td>
</tr>
<tr>
<td>Testing, 196, 262</td>
<td>Varying levels of detail, 161</td>
</tr>
<tr>
<td>Testing and debugging, 66</td>
<td>Verification, 103, 262</td>
</tr>
<tr>
<td>Textbase model, 310–311</td>
<td>Verification of hypothesis, 14</td>
</tr>
<tr>
<td>Text-structure, 293, 376</td>
<td>Verification state, 124</td>
</tr>
<tr>
<td>Textual information, 246</td>
<td>Version, 377</td>
</tr>
<tr>
<td></td>
<td>Version control (VC), 113, 377</td>
</tr>
<tr>
<td>Term</td>
<td>Page(s)</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>Versioned staged model</td>
<td>89–90</td>
</tr>
<tr>
<td>Vertical reuse</td>
<td>377</td>
</tr>
<tr>
<td>Vertical traceability</td>
<td>224, 230, 377</td>
</tr>
<tr>
<td>Virtual Reality Modeling Language (VRML)</td>
<td>268</td>
</tr>
<tr>
<td>Visualization</td>
<td>160</td>
</tr>
<tr>
<td>Walkthrough</td>
<td>276</td>
</tr>
<tr>
<td>Waterfall model</td>
<td>6–7, 86</td>
</tr>
<tr>
<td>Weighted methods per class (WMC)</td>
<td>162</td>
</tr>
<tr>
<td>Weighted sum</td>
<td>271</td>
</tr>
<tr>
<td>Weiser, Mark</td>
<td>158</td>
</tr>
<tr>
<td>White-box reuse</td>
<td>377</td>
</tr>
<tr>
<td>White-box testing</td>
<td>377</td>
</tr>
<tr>
<td>Whole path profiling</td>
<td>235</td>
</tr>
<tr>
<td>Wide Spectrum Language (WSL)</td>
<td>172</td>
</tr>
<tr>
<td>Wireless local area networks (WLAN)</td>
<td>292</td>
</tr>
<tr>
<td>Workspace control</td>
<td>115</td>
</tr>
<tr>
<td>Wrap</td>
<td>11, 188</td>
</tr>
<tr>
<td>Wrapper</td>
<td>11, 64–65, 189, 192</td>
</tr>
<tr>
<td>Wrapping</td>
<td>62, 195, 377</td>
</tr>
<tr>
<td>Wrapping legacy systems</td>
<td>12</td>
</tr>
<tr>
<td>Write coupling</td>
<td>163</td>
</tr>
<tr>
<td>YACC</td>
<td>157</td>
</tr>
<tr>
<td>Zero–one (0–1) matrix</td>
<td>239</td>
</tr>
<tr>
<td>Zipf distribution</td>
<td>246</td>
</tr>
</tbody>
</table>