4 + 1 model, 302
90% complete syndrom, 9
abstraction, 330–333
acceptance testing, 449
activity-on-node network, 204
adaptive maintenance, 16, 468
ADL, 588
afferent coupling, 350
agile cost estimation, 183–184
agile manifesto, 55
agile methods, 27, 51, 54
agile planning, 206
agile team, 108
allocation viewpoint, 301
analyst workbench, 515
antipatterns, 388
application engineering, 76
application framework, 304
application generator, 581
architectural styles, 303
abstract datatype, 311
blackboard, 316
component types, 307
connector types, 308
implicit invocation, 312
layered, 315
main program with subroutines, 310
pipes and filters, 313
repository, 314
Architecture Business Cycle (ABC), 291
Architecture Description Language (ADL), 588
Architecture Tradeoff Analysis Method (ATAM), 318
architecture, service-oriented, 639
Ariane 5, 18
ATAM, 318
risk, 320
sensitivity point, 320
tradeoff point, 320
Attribute-Driven Design (ADD), 295
activity-on-arrow network, 204
backlog, 295
bad smell, 478
ball-and-socket notation, 285
baseline, 87
basic execution time model, 450
beacon, 481
Big Bang, 60
black-box testing, 409
BOOTSTRAP, 143
bottom-up design, 354
bottom-up testing, 449
BPEL4WS, 641, 647
BPR, 232
Brooks’ law, 44, 181
Business Process Redesign (BPR), 232
business service layer, 639
C&C viewpoint, 301
calendar time, 451
call graph, 345
Capability Maturity Model (CMM), 137
critical notes, 143
maturity levels, 138–142
process areas, 138–142
CBSE, 606
benefits of, 607
central transform, 360
Change Control Board (CCB), 88
change request, 88, 497
chief programmer team, 107
choreography, 640
CLM, 72
class diagram, 276
  abstract class, 279
  association, 276
  generalization, 276
  interface, 279
CLG, 547
CMM, 137
CMMI, 138
COCOMO 2, 168
cost drivers, 171
COCOMO, 162
cognitive walkthrough, 563
cohesion, 333
Command Language Grammar (CLG), 547
Commercial Off-The-Shelf (COTS), 239, 627
communication diagram, 284
competent programmer hypothesis, 440
complexity, 6, 337–334
cyclo-matic, 342
  Halstead's method, 339
  information flow metric, 347
  inter-component attributes, 338
  intra-component attributes, 338
  lines of code, 338
McCabe, 342
object-oriented metrics, 348–351
size-based metrics, 338
software science, 339
Stroud's number, 341
structure-based metrics, 338
system structure, 344–348
tree impurity metric, 346
component diagram, 285
component model, 614–619
  and architecture, 614–619
  common features, 617
  definition, 609
  quality properties and, 615
component technology, definition, 610
component
  assessment, 622
  definition, 609
  development process, 623
  forms, 610–614
  lifecycle of, 610
Component-Based Software Engineering
  (CBSE), 606
Computer Aided Software Engineering
  (CASE), 508, 514
cost drivers, 161
corrective maintenance, 16, 468
correctness proof, 428
cost distribution, 4
cost drivers, 45
cost estimation, 44
  agile, 183–184
  algorithmic models, 158
COCOMO 2, 168
COCOMO, 162
cone of uncertainty, 178
cost drivers, 161
extpert, 177
FPA, 165
guidelines, 175–179
ideal days, 184
impossible region, 181
Putnam, 163
Rayleigh curve, 164
story point, 183
Walston-Felix, 160
cost of software development, 3
COTS selection, 239
COTS, 239, 627
coupling effect hypothesis, 440
coupling, 335
coverage-based testing, 409, 430–437
CPM, 205
CRC cards, 263, 267
Critical Path Method (CPM), 205
critical path, 204
crowdsourcing, 28, 240
CSCW, 556
customer-developer links, 234
cyclomatic complexity, 342, 432
daily build, 55
data abstraction, 332
data flow design (SA/SD), 356–361
Data Flow Diagrams (DFD), 263, 267, 357
dataflow coverage, 433
definition-clear path, 434
deployment viewpoint, 302
design decisions, 296
design patterns, 304, 385–389
Command Processor, 387
MVC, 385
Proxy, 387
design recovery, 475
design, 12
abstraction, 330–333
Booch method, 376
bottom-up, 354
classical methods, 351–369
classification of methods, 381
cohesion, 333
complexity, 337–344
coupling, 335
data flow, 356–361
data structures, 360
divide-and-conquer, 353
documentation, 389–393
functional decomposition, 353–356
Fusion, 377
how to select a method, 380
information hiding, 336
Jackson, 361
modularity, 333–336
object-oriented, 369–380, 382–384
properties, 330
RUP, 379
Scandinavian school, 328
stepwise refinement, 353
structured analysis (SA), 357
structured design (SD), 357
top down, 354
V&V, 393
wicked problem, 327
development for reuse, 583
development with reuse, 583
DFD, 263
divide-and-conquer, 353
documentation, 14, 44
domain adequate test set, 441
domain engineering, 76
Dynamic Systems Development Method (DSDM), 64
principles, 65
Eclipse, 513
efferent coupling, 350
egoless programming, 426
Enterprise Service Bus (ESB), 640
Entity-Relationship Modeling (ERM), 263
Entity-Relationship Diagram (ERD), 264
ERM, 263
error seeding, 437
error, 410
error-based testing, 409, 440–441
ethnography, 229
evolutionary prototyping, 58
execution time, 451
Extensible Markup Language (XML), 641, 643
extreme programming, 66–68
Fagan inspection, 426
failure intensity, 452
failure, 410, 450
fault, 410, 450
fault-based testing, 409, 437–440
Finite State Diagram (FSD), 366
Finite State Machines (FSM), 263, 265–266
follow-the-sun development, 657
fourth-generation language, 581
Function Point Analysis (FPA), 165
application characteristics, 167
complexity levels, 166
functional decomposition, 353–356
Fusion, 377

Gantt chart, 204, 205
glBIS, 236
global software development
and culture, 662
and distance, 664–669
challenges, 657–664
common ground, 664
organizing work, 668
technology, 666
goal-driven requirements engineering, 235
Groupware Task Analysis (GTA), 556

groupware, 552

Halstead’s method, 339
heavyweight methods, 51
heuristic evaluation, 563
hierarchical organization, 104–105
hierarchical task analysis (HTA), 563
human information processing, 542
human-computer interaction models, 540

idiom, 304
IDL, 590
IEEE 1012, 423
IEEE 1016, 389
IEEE 1219, 497
IEEE 1471, 299, 392
IEEE 730, 136
IEEE 828, 92
IEEE 829, 423
IEEE 830, 241
IEEE 983, 137
implementation viewpoint, 302
implementation, 13
incremental development, 60–62
information flow metric, 347
information hiding, 336
information planning, 38
infrastructure service layer, 639
inspection, 100, 426
installation test, 449
integrated environment, 509, 514
Integrated Project Support Environment (IPSE), 514, 520
integration testing, 449
interaction diagram, 283
Interface Definition Language (IDL), 590
ISO 9000, 134
ISO 9126, 125–130
ISO 9241, 566
ISO/IEC 15504, 143

Jackson diagram, 361
Jackson Structured Programming (JSP), 361
Jackson System Development (JSD), 361–369
Joint Application Design (JAD), 63
Joint Requirements Planning (JRP), 63
JUnit, 421

Kano model, 238
Keystroke model, 547
KLOC, 155

language-centered environment, 508, 513
LAS, 21
laws of software evolution, 73, 473
lightweight methods, 51
logarithmic Poisson execution time model, 450
logical viewpoint, 302
London Ambulance Service (LAS), 21
long-term memory, 544

Make, 518
management styles, 102–103
management workbench, 520
manpower distribution, 179–182
matrix organization, 106–107
maturity level, 138
MDA, 71–72
measure, 122
direct, 122
indirect, 122
valid, 123
measurement framework, 120
mental model, 532, 544
middleware, 588
migration, 477
MIL, 586
minispecs, 359
Model View Controller (MVC), 385, 534
Model-Driven Architecture (MDA), 71–72
Module Interconnection Language (MIL), 586
module viewpoint, 300
MoSCoW, 63, 237
mutation testing, 438
strong, 439
weak, 439
MVC, 385
object point, 168
object
aggregation, 274
composition, 274
generalization, 272
hierarchy, 273
is-a hierarchy, 272
member-of, 272
properties, 271
specialization, 272
types of, 269–271
whole-part, 272
object-oriented analysis and design, 369–380
object-oriented metrics, 348–351
afferent coupling, 350
CBO, 349
DIT, 349
efferent coupling, 350
Law of Demeter, 349
LCOM, 350
NOC, 349
RFC, 350
WMC, 348
offshoring, 657
Ontology Web Language (OWL), 644
open source, 28, 108–111
onion structure, 109
open standards, 638
operational profile, 454
orchestration, 640
outsourcing, 657
overfunctionality syndrome, 60
OWL, 644
pair programming, 66
Participatory Design, 64
Paul Principle, 112
peer reviews, 100, 425, 427
people management, 99–104
perfective maintenance, 16, 468
Personal Software Process (PSP), 142
PERT, 203
Peter Principle, 106
Petri nets, 78
PIM, 72
planning
agile, 206
task oriented, 206
planning-driven methods, 51
preventive maintenance, 16, 468
procedural abstraction, 332
process area, 138
process modeling, 77–80
process programming language, 77
Process Structure Diagram (PSD), 366
process viewpoint, 302
process-centered environment, 509, 521
product line development, 626
program comprehension, 304, 481
program inversion, 364, 368
programmer workbench, 516
programming plan, 304, 481
programming-in-the-large, 6
programming-in-the-small, 6
project characteristics, 193
project control, 43–45, 190–198
allocation type, 195
design type, 195
project control (continued)
exploration type, 196
realization type, 194
project management, 14
project plan, 40–43
prototype, 566
prototyping, 56–60, 233
advantages, 58
disadvantages, 58
evolutionary, 58
throwaway, 58
PSM, 72
PSP, 142

QoS, 132, 637
quality attribute
external, 122
internal, 122
taxonomy, 123
quality factors, 124
quality in use, 127
Quality of Service (QoS), 132, 637
quality, 44
assurance, 135
conformance, 116
external, 127
improvement, 116
internal, 127
manufacturing-based, 131
measurement, 118–123
perspectives on, 130
process, 116
product, 116
product-based, 131
transcendent, 131
user-based, 131
value-based, 131
quality-attribute scenario, 132
Questions, Options, Criteria (QOC), 249

Rapid Application Development (RAD), 62–66
Rational Unified Process (RUP), 68–71, 302, 379
rationale, 296
Rayleigh curve, 180
Rayleigh distribution, 164
redocumentation, 475
reengineering, 476
refactoring, 55, 67, 388, 477, 478
regression testing, 420
release, 499
renovation, 477
representation condition, 123, 350
requirements creep, 248
requirements documentation, 241–247
requirements elicitation, 218, 220–241
asking, 227
business process redesign, 232
Delphi technique, 227
ethnography, 229
form analysis, 230
from existing system, 231
natural language description, 231
prototyping, 233
scenario based, 228
task analysis, 228
techniques, 227
requirements engineering, 12
functionalism, 225
goal-driven, 235
neohumanism, 225
paradigms, 224–226
prioritization, 237
radical-structuralism, 225
social-relativism, 225
V&V, 219, 253
requirements management, 247–249
requirements negotiation, 219
requirements specification, 218, 241
non-functional, 252
response set, 350
restructuring, 476
return on investment, 159
reuse
black-box, 575
component libraries, 576
development for, 575, 583
development with, 575, 583
devil’s loop, 596
dimensions, 574
economics of, 596
management of, 597
non-technical aspects, 594–599
perspectives, 591–594
psychology, 598
software architecture, 581
software life cycle and, 582–585
tools, 585
white-box, 575
reverse engineering, 475–486
limitations of, 480
tools, 484
reverse Peter Principle, 112
risk factors, 200
risk management, 198–201
risk, 41
RUP, 68–71, 302, 379
practices, 70
process structure, 69
workflows, 69
scale types, 121
Scandinavian school, 64
scenario-based analysis, 228
schematic logic, 362
Scrum, 295
Seeheim model, 533
sequence diagram, 283
lifeline, 283
service bus, 639
Service Level Agreement (SLA), 637
service, 634
characteristics, 634–639
choreography, 640
contract, 635
discovery, 634
orchestration, 640
Service-Oriented Architecture (SOA), 639
business service layer, 639
coordination layer, 639
infrastructure service layer, 639
Service-Oriented Software Engineering (SOSE), 650
Simple Object Access Protocol (SOAP), 641, 644
SLA, 637
SOAP, 641, 644
software architecture, 581
4 + 1 model, 302
allocation viewpoints, 301
and CBSE, 625
architectural styles, 304
assessment, 317
business-oriented viewpoint, 303
C&C viewpoints, 301
conceptual viewpoint, 302
definition, 292
deployment viewpoint, 302
design decisions, 296
design workflow, 296
design, 294
implementation viewpoint, 302
logical viewpoint, 302
module viewpoints, 300
place in life cycle, 293
process viewpoint, 302
purposes, 290
rationale, 296
view, 299
viewpoint, 299
software development productivity, 3
software development
effort distribution, 15
phases in, 10
software engineering
characteristics of, 6–8
continuity, 9
definition, 5
ethics, 24–26
process model, 10
visibility of, 9
software evolution, 7, 72–75, 486–488
history-centered analysis, 486
laws of, 73, 473
version-centered analysis, 486
software factory, 82
software lifecycle, 582–585
software maintenance, 13, 72–75
adaptive, 16, 468
categories of, 468
control of, 497
corrective, 16, 468
cost of, 468
definition, 467
distribution, 469
software maintenance (continued)
  iterative-enhancement model, 498
  life cycle stages, 470
  organization of, 488
  perfective, 16, 468
  preventive, 16, 468
  quick-fix model, 499
  service perspective, 492
Software Process Improvement (SPI), 142, 145
software product lines, 75–77, 626
software productivity, 99, 156, 180–182
Software Quality Assurance (SQA), 135
software reliability, 450–457
software science, 339
SOSE, 650
Source Code Control System (SCCS), 516
SPI, 142
SPICE, 143
spiral model, 61
SQA, 135
state machine diagram, 279
State Transition Diagram (STD), 266
statechart, 266
STD, 266
stepwise abstraction, 429
stepwise refinement, 353
story point, 183
structure chart, 360
structure clash, 362
structure diagram, 361
structure text, 362
structured analysis (SA), 357
structured design (SD), 357
Studio .NET, 513
SWAT team, 64, 107–108
synchronize and stabilize, 55
System Specification Diagram (SSD), 368
system testing, 449

Task Action Grammar (TAG), 547
task analysis, 228, 554
Taylorian approach, 224
team coordination, 101
team organization, 104–112
test-driven development (TDD), 421
testing, 13
  adequacy criterion, 412, 442
  all-edges coverage, 432
  all-nodes coverage, 432
  all-paths coverage, 431
  branch coverage, 432
  comparison of techniques, 441
  confidence building, 414
correctness proofs, 428
cost of error correction, 407
coverage-based, 409, 430–437
documentation, 422–424
dynamic, 425
error seeding, 437
error-based, 409, 440–441
experimental results, 446
fault detection, 413
fault prevention, 415
fault-based, 409, 437–440
inspection, 426
manual techniques, 425–430
mutation testing, 438
objectives, 410
process, 411
requirement specification, 435
software life cycle and, 417–421
stages, 448
statement coverage, 432
static, 425
stepwise abstraction, 429
walkthrough, 426
white-box, 409
Therac 25, 19
throwaway prototyping, 58
time box, 62
toolkit, 508, 511
top-down design, 354
top-down testing, 449
Total Quality Management (TQM), 133, 137
traceability, 249
tree impurity, 346
triage, 63

UDDI, 641, 646
Unified Modeling Language (UML), 274
ball-and-socket notation, 285
class diagram, 276
communication diagram, 284
component diagram, 285
diagram types, 275
interaction diagram, 283
sequence diagram, 283
state machine diagram, 279
use case, 286
unadjusted function points, 166
unit testing, 449
Universal Description, Discovery and Integration (UDDI), 641, 646
Universe of Discourse (UoD), 220
use case points, 173
use case, 286
user interface, 536
evaluation, 562
User Virtual Machine (UVM), 536, 559
utility tree, 319

validation, 410
verification, 410
version control, 90, 517
very-high-level language (VHLL), 582
view, 299
viewpoint, 236, 299
V-model, 53
walkthrough, 100, 426
waterfall model, 52
WBS, 201
Web Services Description Language (WSDL), 641, 644
Web services, 641
stack, 642
Weighted Scoring Method (WSM), 240
white-box testing, 409
wicked problem, 327
winner's curse, 176
Work Breakdown Structure (WBS), 201
workbench, 508, 514
working memory, 543
WSDL, 641, 644
WSM, 240

XML, 634, 641
XP, 66
X practices, 67
X principles, 68
yesterday's weather, 184, 486