Index

Page numbers followed by an f or t indicate figures and tables.

A
Abstract class, 101, 193, 427
Abstraction analysis, 137
Abstractions
  in architecture functionality, 351
  encapsulations and, 51
  examples of, 47–50
  in hierarchy of complex systems, 4–5
  key
    defined, 112, 138
    identifying, 139–41, 284f
    naming, 140–41
    refining, 139–40
    in traffic management system, 395–96
  levels of, in micro process, 274–76, 275f, 281–83
  meaning of, 44–47
  in programming language evolution, 31
  by programming style, 43
  quality of, measuring, 112–14
  role of, 23–24, 200
  software flexibility and, 10
  static and dynamic properties of, 46–47
  viewers perspective in, 44–45
  in weather monitoring system, 467
Access, package, 161–63
Action abstraction, 45
Activity diagram
  actions in, 186
  black-box, 345f
  decision and merge nodes in, 187–88
  fork and join nodes in, 188, 190
  object flows in, 190
  partitions in, 188, 189f
  for satellite navigation system, 341f, 346f
  starting and stopping points in, 186–87
  for traffic management system, 386f, 387f
  white-box, 356f, 367f
Actors, in use case diagram, 176
Ad hoc approach, 125
Affirmation class, 424–25, 426f
Aggregation
  choice of, 196
  of classes, 109–11
  hierarchy in, 63–64
  of objects, 91–92, 92f
  as physical containment, 110–11
  vs. multiple inheritance, 110–11
Agile development process, 252–53, 255f
Algorithmic decomposition, 19
  vs. object-oriented, 20–23
Alphabet class, 428–29
Alternatives interaction operator, 212
Analysis categories, 21–22
Analysis/design model, 277–78
in vacation tracking system, 508–19
Analysis process. See also Micro process
abstraction levels in, 273–76
classes and objects in, 112
element identification in, 284r–285r
in iterative development, 270
object-oriented. See Object-oriented analysis
Analyst role, 311
Animals, complex structure of, 5–6
Apache Struts framework, 518, 528–29
Application engineers, 310
Architect, project, 309–10
Architectural design, 281, 285r
Architecture analysis, 281
Architecture description, 277
Architecture documentation, 278–80, 349
Architecture of system, 16
defined, 248
in design process, 248–49
in satellite navigation system
activity definition, 348–50
decomposing, 364–70
deployment of, 361–64
developing, 348
nonfunctional requirements, 358–61
validating, 350–57
time-frame-based processing in, 473–74
in traffic management system
defining, 389–91
system functionality and, 385–87
in vacation tracking system, 517–18
in weather monitoring system, 473–74, 475r
Web-centric, 494
Arithmetic/logic unit (ALU), 4
Artifact
defined, 171
notation for, 171–72
reuse of, 314–16
Artificial intelligence, 40. See also Cryptanalysis
system
Assembly connectors, 166, 217
Assertion class, 425
Associations
among classes, 97–98
among elements, 293–94
classes and notes, 205–6
directionality, 199
end names and qualifiers, 200–201
one-to-many, 98r
visibility, 199
Assumption mechanism, 443–44
Astronomy, 6
Backward-chaining, 419
Bandwidth, 535
Behavior
in choice of class, 115
in element collaboration, 291
fundamental, 291
of objects, 81–82
Behavior analysis
description of, 132–33
process of, 290–91
Behavior characterization, 11–12
Behavior diagrams, 150r, 151
Behavioral prototype, 260
Biological classification, 123–24
Black-box activity diagram, 345r, 365
Blackboard framework
architecture of, 417, 418–20
assumption mechanism in, 443–44
integration of, 440–44
knowledge sources in, 418–20, 444–45
Blackboard objects
classes in, 421–23
designing, 427–31
topmost, 440–43
Block diagram, 389r, 392
Botany, 5
Boyle, Robert, 124
Business function, 133
Business logic, 533–34
C
C++, 546–51
C with Classes, 546
CAD/CAM system, 77
Canonical form of complex system, 15–17, 17r
Capability-based architectures, 39
Central processing unit (CPU), 4
Chaos, prediction and, 11n
Chemical classification, 124
Chunking, 23
CK metrics, 320
Class diagram
  association classes and notes, 205–6
  association end names and qualifiers in, 200–201
  constraints in, 201–4, 203f
  element visibility in, 198–200, 199f
  notation in, 192–93
  relationships in, 194–97, 195f
  in schedule planning, 401f
  template classes in, 197–98
  in traffic management system, 398, 399f
Class library tool, 323
Class lifecycle, 96
Class promotion, 140
Class structure, 15–17, 17f
Class(es). See also specific classes
  abstract, 101, 193, 427
  affirmation, 424–25
  alphabet, 428–29
  assertion, 425
  association among, 97–98
  behavior in choice of, 115
  in blackboard framework, 421–23
  collection, 477
  concrete, 101
  defined, 92–94
  dependencies among, 111, 423–26
  design quality of, 112–14
  identifying, 126–38
  implementing, 94–95
  inheritance among, 98–102
  interface of, 94–95, 476
  interplay of, with objects, 111–12
  leaf, 101
  relationships among, 96–97, 204
  sources of, 131–32
  template, 197–98
  for time and date, 452–54
  vs. type, 64f
Classical analysis, 131–32
Classical categorization, 126–27
Classification
  approaches to
    application of, 130–31
    classical categorization, 126–27
    conceptual clustering, 127–29
    prototype theory, 130
    defined, 121
    difficulty of, 122–24
    importance of, 121–22
    incremental and iterative nature of, 124–26
Client state management, 499
CMP beans, 519, 520
Cohesion, 113, 354
Coincidental abstraction, 45
Collection class, 477
Commonality, in complex structures, 5
Communication diagram, 238–43
  in vacation tracking system, 514–15, 516f
Communication gap, 9
Completeness, 114
Completion transition, 221
Complex systems
  attributes of, 12–15
  behavior characterization in, 11–12
  canonical form of, 15–17, 17f
  designing, 24–28
  hierarchical nature of, 4–5
  structure of, 4–7
Complexity
  abstraction of, 23–24, 44
  arbitrary, 7
  external, 9
  human limitations and, 17–18
Component analysis, 281
Component design, 281–82
Component diagram
  defined, 163
  interfaces in, 166–68, 167f
  internal structure of, 169–71, 170f
  notation for, 164–65, 164f
  realizations in, 168, 169f
  for satellite navigation system, 362f
  for traffic management system, 408f
Component lead, 310
Composite design, 21
Composite state, 224, 225f
Composite state notation, 226
Composite structure diagram, 215–18
  for cryptanalysis system, 441f
Composition, 110, 125
Computer languages. See Programming languages
Conceptual clustering, 127–29
Conceptual model, 152
Concerns, separation of
  in complex system, 13–14
  encapsulation in, 52
  in temperature measurement abstraction, 49
Concrete classes, 101
Concurrency
  examples of, 68–69
  heavyweight, vs. lightweight, 67
  meaning of, 66–68
  in state machine diagram, 225–30
Concurrent synchronization, 91
Configuration management, 312–13
Connectors, in composite structure diagram, 217
Constraint
  association, 203
  defined, 201
  placement of, 202
Construction phase
  in cryptanalysis, 427
  description of, 266
  in satellite navigation system, 371
  in traffic management system, 396–97
  in vacation tracking system, 506
  in weather monitoring system, 474
Constructor operation, 82
Context diagram, for satellite navigation system, 337
Contract model of programming, 46
Control constructs, 210–12
Control flow notation, 226
Control objects, 76
Control system. See Traffic management system
Controller object, 89
  in knowledge source activation, 438–39
  in vacation tracking system, 528–29
Coupling, 113
CRC cards, 135–36
Cryptanalysis system
  blackboard framework in
    architecture of, 417, 418–20
    assumption mechanism in, 443–44
    integration of, 440–44
    knowledge sources in, 418–20, 444–45
  blackboard objects
    classes in, 421–23
    designing, 427–31
    construction phase in, 427
  controller design in, 438–39, 440
  defined, 414
  dependencies in, 423–26, 426f
  knowledge sources in
    in blackboard framework, 418–20, 444–45
    designing, 431–34
    generalizing, 435–37
    implementing, 445
    post-transition phase in, 446–47
    problem-solving process in, 416–18
    requirements for, 414–15
    changes in, 448
    substitution cipher, 414–15
    system enhancements, 446–47

D
Darwin, Charles, 123
Data abstraction, 35
Data acquisition system, 449. See also Weather monitoring system
Data-driven design, 22
Data member, 95
Decision nodes, 187–88
Decomposable hierarchic systems, 13
Decomposition
  algorithmic, 19
  vs. object-oriented, 20–23
  object-oriented, 20
  role of, 19–23
  of system architecture, 364–70
  in traffic management system, 409
Defect density, 317
Defect-discovery rates, 266
Delegation, 98
Delegation connectors, 171, 217
Demeter, Law of, 116–17
Dependency class, 111, 423–26, 426f
Dependency relationships, 424–25
  in package diagrams, 158–59
Deployment diagrams
  artifact notation in, 171–72
  connections in, 173–75, 174f
  node notation in, 172–73
  in traffic management system, 394f
  in vacation tracking system, 494–96, 495f
Derivation, 125
Descartes, René, 124
Descriptor-based architectures, 39
Design method categories, 21–22
Design methodologies, 26–27
Design metrics, 318
Design model, in vacation tracking system, 508–19, 521f, 522f
Design process. See also Micro process
abstraction levels in, 275–76
classes and objects in, 112
of complex systems, 24–28
element identification in, 284r–285f
in iterative development, 270
purpose of, 25–26
Destruction events, 208
Destructor operation, 82
Development legacy, 320–21
Development process. See also Object-oriented development
agile, 252–53, 255f
phases in, 267–68
architectural vision in, 248–49
choice of, 254
in design methodology, 27
iterative
advantages of, 251
analysis and design in, 270
duration for, 269
focus shifts in, 268f
macro
construction phase in, 266
disciplines in, 259, 261
elaboration phase in, 264–66
inception phase, 262–64
iterations in, 268–69
milestones and phases in, 261–68
purpose of, 257
release planning in, 270–73
transition phase in, 267
micro
abstraction levels in, 274–76, 275f, 280–83
activities in, 276–77
element collaborations in, 288–92
element identification in, 283–88
element relationships in, 292–95
macro process and, 273f
products of, 277–78
semantic detail in, 296–300
plan-driven, 253, 255f
prototyping in, 260
software architecture documentation, 278–80
Development reviews, 306–7
Development team roles, 309–12
Device, 172
Diagram notation
activity, 185–91
behavior, 150f, 151
class, 192–206
communication, 238–43
component, 163–71
composite structure, 215–18
deployment, 171–75, 174f
interaction, 150f
interaction overview, 213–15
object, 235–38
package, 156–63
sequence, 206–12
state machine, 218–31
structure, 149–50, 150f
timing, 231–35
use case, 175–85, 178f
Diagrams. See also specific types
connectivity among, 154
context, 337f
practical use of, 151–52
state transition, 440f
Discrete systems, behavior characterization in, 11–12
Display mechanism, in weather monitoring system, 482–83
Do activity, 222
Documentation, 320–22
Documenter, 311
Domain analysis, 133–34
Domain expert, 134
Domain-specific issues, 324–25
Drawing mechanism, 143
Duration timer, 223
Dynamic typing, 66

E

Early binding, 66
Elaboration phase
of cryptanalysis system, 421
description of, 264–66
of satellite navigation system, 347–48
of traffic management system, 385
of vacation tracking system, 494
of weather monitoring system, 463
Elements
  associations among, 293–94
  collaborations among, 288–92
  milestones in, 291–92
  identification of, 283–88, 284t–285t
  milestones in, 288
  pattern scavenging, 291
  protocols of, 297
  relationships among, 292–95
  semantics of, 296–300
  visibility of, 157–58
Encapsulation
  in architecture functionality, 351
  examples of, 52–54
  meaning of, 50–52
  separation of concerns in, 52
Energy management system, 390
Engineering, 25
English description, in object-oriented analysis, 136
Entities
  finders in filtering, 528
  persistent, 519–20
  in vacation tracking system, 519–23
Entity abstraction, 45
Entity bean, 521
  primary key generation for, 525
Entry activity, 222
Environmental control system
  component notation in, 164f/
  deployment diagram for, 174f/
  internal structure of, 170f/
  specification of two interfaces for, 167f/
Exclude relationships, 180–84
Execution environment, 172
Execution specification, 209
Exit activity, 222
Extend relationships, 181–83

F
  Façade object, 523, 524f/
  Factorization, 125
  Federal Bureau of Investigation, 303
  Field, 95
  Final node, 187
  Final state, 219–20
  Finder methods, 528
Fine-grained operations, 115
Finite state machine, 485–86
Firing the transition, 220
Fork node, 188, 190
Fork vortex, 228
Forward-chaining, 419
Found message, 208
4+1 architecture view model, 279–80
Frames
  in interaction overview diagrams, 213
  in weather monitoring system, 474–79
  in web applications, 535
Frameworks, 143
Function points, 133
Functional semantics, of operation, 115
Functionality allocation, 353
Fuzzy set theory, 128

G
  Global Positioning System (GPS), 334, 335
Glossary, 338
Grainsize conflict, 140
Guard conditions, 243
Guarded synchronization, 91

H
  Hardware
    post-transition changes to, 372–73
    in weather monitoring system, 451, 452f/
Heavyweight concurrency, 67
Hierarchical structure
  aggregation, 63–64
  in architecture functionality, 352
  class and object in, 15–17
  in complex system, 4–5, 12
  inheritance in
    multiple, 61–63
    repeated, 62
    single, 58–61
  of knowledge source class, 434f/
  meaning of, 58–64
  role of, 24
  in weather monitoring sensor class, 460f/
HTTP state management mechanism, 499
Human intelligence, 40
Hydroponics gardening system
  communication diagram for, 240f
  composite structure diagram for, 216f
  package import in, 162f
  package notation for, 156f, 160f
  transitions in, 221
Hyperlinks, 534

I
Identity
  defined, 85
  of object, 85–87, 86f
Idioms, 142–43
Implementation model
  choosing, 117–18
  of class, 94–95
  mechanisms of, 112
  in vacation tracking system, 501
Import package, 161–63
Inception phase
  of cryptanalysis system, 414
  description of, 263–64
  of satellite navigation system, 334
  of traffic management system, 376
  in vacation tracking system, 490
Include relationships, 180–84
Incremental development approach, 250–52
  integration events in, 313–14
Information display, 404–5
Informational capacity
  abstraction in transcending, 23
  of human, 18
Inheritance
  element semantics and, 299
  metrics of, 319
  multiple
    among classes, 106–9
    name collisions in, 107–8
    in object model, 61–63
  vs. aggregation, 110–11
  polymorphism and, 102–3
  repeated, 62, 109
  single
    among classes, 100–102
    defined, 100
    in object model, 58–61
Initial node, 186
Initial state, 219–20
Instance variable, 95
Integrated development environment (IDE), 323
Integration events, 313–14
Integration manager role, 311
Intelligent system. See Cryptanalysis
Interaction operator loop, 210–12
Interaction overview diagram, 213–15
  for traffic management system, 388f
Interaction use, 210
Interface
  of class, 94–95
  in vacation tracking system, 529–34
  in weather monitoring system, 476f, 483–87
Interface connectors, 166, 217
Intermediate form stability, 14
Invariance, 46
Iteration clause, 242–43
Iteration release, 272
Iterative development
  advantages of, 250–52
  duration in, 269, 270–71
  focus shifts in, 268f
  integration events in, 313–14
  in macro process, 268–69
Iterator operation, 82

J
Java, 551–56
Java scripting, 535
Java Server Pages (JSP), 517, 532
J2EE technologies, 518
Join node, 188, 190
Join vertex, 228
JSF, 518

K
Key abstractions
  defined, 112, 138
  identifying, 139–41, 284f
  naming, 140–41
  refining, 139–40
  in traffic management system, 395–96
Knowledge sources
- analysis of, 420–21
- in blackboard framework, 418–20, 444–45
- dependencies among, 423–26
- designing, 431–34
- generalizing, 435–37
- implementing, 445
- pattern-matching, 433

Late binding, 66
- polymorphism and, 103
Lavoisier, Antoine, 124
Law of Demeter, 116–17
Leaf classes, 101
Legacy system, 325
Lifeline, 206, 232
Lightweight concurrency, 67
Links
- in communication diagrams, 239
- defined, 88
- between objects, 88–91, 89f
Linnaeus, Carolus, 123
Logical model, 152
- for satellite navigation system, 341f
- in satellite navigation system, 368f
- in vacation tracking system, 496–98, 497f
Lorenz design metrics, 319

Machines, objects as, 84–85
Macro process
- construction phase in, 266
- disciplines in, 259, 261
- elaboration phase in, 264–66
- inception phase, 262–64
- iterations in, 268–69
- milestones and phases in, 261–68
- purpose of, 257
- release planning in, 270–73
- transition phase in, 267
Maintenance of software, 10, 258–59
Management, project, 304–8
Many-to-many relationships, 98
Matter, structure of, 6

Mechanisms
- defined, 112, 138
- examples of, 143–44
- identifying, 140–44
- in object structure, 24
- as patterns, 142–43
- in relationship choice, 117
Member function, 46
Mendeleyev, Dmitry, 124
Merge nodes, 187–88
Message dictionary, 105
Message passing
- between objects, 88–89
- in traffic management system, 397–400, 400f
Messages
- in communication diagram, 239
- defined, 81
- found, 208
- guard conditions in, 243
- notation for, 206–7
Method
- selection of, 104–6
- as term, 46
- vs. methodology, 21
Method-dispatch algorithm, 105–6
Metrics, object-oriented, 317–20
Milestones
- in construction phase, 266
- in elaboration phase, 265–66
- in inception phase, 264
- in micro process
  - element collaboration, 291–92
  - element identification, 288
  - element relationships, 295
  - element semantics, 299–300
- in transition phase, 267
Minsky, Marvin, 331
Mission use case, 339–43
Mixin classes, 61, 109
Model building
- importance of, 26
- in micro process, 277
  - Unified Modeling Language and, 148–49
Model-view-controller paradigm, 143
Modeling language. See Unified Modeling Language
Modifier operation, 82
Modularity
- in architecture functionality, 352
INDEX

defined, 56
examples of, 57–58
guidelines for, 56
meaning of, 54–57
size limitations in, 57
Monomorphism, 66
Multiple inheritance
among classes, 106–9
name collisions in, 107–8
in object model, 61–63
vs. aggregation, 110–11
Multiplicity among classes, 98
Multivalue set theory, 128
MVC pattern, 143

N
Name collisions, 107–8
Navigation. See Satellite Navigation System
Nearly decomposable hierarchic systems, 13
Nested state, 224, 225f
Node
defined, 172
notation for, 172–73
Notation. See also Diagram notation; Unified Modeling Language
in design methodology, 27
purpose of, 147–48
Noun-verb analysis, 508

O
Object-based programming languages, 32
topology of, 35–36, 36f, 37f
vs. object-oriented, 41
Object diagram, 235–38
in vacation tracking system, 514–15, 515f
Object flows, 190
Object Management Group (OMG), 148
Object model
application of, 71–72
benefits of, 71–72
elements of, 43–44
evolution of, 29–37, 39
foundations of, 37–43
as principle, 18
in programming language evolution, 538
Object nodes, 191f
Object-oriented analysis
approaches to
behavioral, 132–33
classical, 131–32
CRC cards, 135–36
domain analysis, 133–34
structured analysis, 136–38
use case analysis, 135
defined, 42–43
Object-oriented decomposition, 20
vs. algorithmic, 20–23
Object-oriented design
defined, 38, 42
models of, 27
in object-oriented decomposition, 42
Object-oriented development. See also Development process
adoption of, 325–26
benefits of, 326–27
documentation in, 320–22
domain specific issues in, 324–25
failures in, 303
management of
development reviews, 306–25
risk management, 304–5
task planning, 305–6
metrics in, 317–20
quality assurance and in, 316–17
release management in, 312–14
reuse in, 314–16
risks of, 327–29
staffing in
development team roles, 309–12
resource allocation, 308–9
tools in, 322–24
Object-oriented metrics, 317–20
Object-oriented operating systems, 39
Object-oriented programming (OOP)
defined, 41
languages in support of, 41–42
Object structure, 15–17, 17f
Objects
aggregation of, 91–92, 92f
behavior of, 81–82
in communication diagrams, 239
control, 76
defined, 76–78
design quality of, 112–14
Objects (continued)
façade, 523, 524f
identity of, 85–87, 126–38
interplay of, with classes, 111–12
links between, 88–91
as machines, 84–85
operation of, 82
relationships among, 88–92
roles and responsibilities of, 83–84
sources of, 131–32
state of, 78–81
One-to-one relationships, 98
One-to-many relationships, 98
Operations
choosing, 114–16
defined, 81
functional semantics of, 115
of objects, 82
as term, 46
time and space semantics of, 115–16
Organizational relationships, 7
Orthogonal hierarchies, 15, 22
Overloading, 103
Ownership, in aggregation, 63–64

P
Package diagrams
benefits of, 155–56
dependency relationships in, 158–59
element visibility in, 157–58
import and access, 161–63
notation for, 156–57, 156f
for satellite navigation system, 341f
in use case organization, 160f
Package export, 161–63
Package import, 161–63
Package interface, 94
Packaging, in relationship choice, 118
Parameter formatting, 497
Parameterized classes, 197–98
Partitions, 188, 189f
Pattern commonality, 14
Pattern-matching knowledge source, 433
Pattern scavenging, 291
Patterns
mechanisms as, 142–43
MVC, 143
Persistence, 69–71
Persistent data storage, 519
Personal computers, 4–5
Petroski, Henry, 145
Physical model, 152
Plan-driven process, 253, 255f
Plants, complex structure of, 5–6
Polymorphism
ad hoc, 103
defined, 102–3
late binding and, 103
method selection and, 104–6
in type theory, 66
Ports, in component notation, 164–65
Post data formatting, 497
Post-transition phase
in cryptanalysis system, 446–47
hardware changes in, 372–73
in satellite navigation system, 371–73
in traffic management system, 411–12
in vacation tracking system, 534–35
in weather monitoring system, 487–88
Postconditions, 46
in weather monitoring system, 472–73
Preconditions, 46
Preservation of software, 10
Primary keys, 525
Primitive components, 113
Primitiveness, 114
Principle of least astonishment, 44
Principle of least commitment, 44
Private interface, 94
Private visibility, 157
Problem domain, complexity of, 8–10
Process metrics, 317
Process view, in Web applications, 498–501
Product metrics, 318
Products
of construction phase, 266
of elaboration stage, 264
of element collaboration definition, 288–89
of element identification, 284
of element relationship definition, 293
of inception phase, 264
of micro process, 277–78
of semantic detailing, 296–97
of transition phase, 267
Programming-in-the-large, 34
Programming languages
C++, 546–51
decomposition in, 21
evolution of, 30–32
genealogy of, 539f
idioms in, 142–43
Java, 551–56
object-based, 32
topology of, 35–36, 36f, 37f
vs. object-oriented, 41
object-oriented, 32
adoption of, 326
criteria for, 41
evolution of, 537–39, 539f
persistence support in, 70
topology of, 35–36, 36f, 37f
vs. object-based, 41
popularity index for, 538, 540f
Smalltalk, 541–42
topology of
defined, 32
first- and early second-generation, 32–33, 33f
late second- and early third-generation, 33–34, 34f
late third-generation, 34–35, 35f
object-based and object-oriented, 35–36, 36f, 37f
Programming styles
contract model of, 46
main kinds of, 43
Project architect, 309–10
Project management, 304–8
Project manager role, 311
Project metrics, 317
Project staffing, 308–12
Property
in classical categorization, 127
defined, 79
of object, 78–79
value of, 79
Protected interface, 94
Protocol
defined, 46
of object, 83
semantics of, 297
Prototype theory, 130
Prototypes, 260
in weather monitoring system, 459
Proxy object, 89
Public interface, 94
Public visibility, 157

Q
Qualifier, 238
Quality assurance, 311, 316–17

R
Railroads. See Traffic management system
Real-time, 324–25
Realizations, component, 168, 169f
Reflexive association, 194
Relationships
among classes, 96–97
among elements, 292–95
among object, 88–92
choosing, 116–17
include and exclude, 180–84
independent and direct, 513f
one-to-one, 98
one-to-many, 98
many-to-many, 98
organizational, 7
Release planning
in macro process, 270–73
in object-oriented development, 312–14
for traffic management system, 406–7
for weather monitoring system, 479–80
Repetitive inheritance, 62, 109
Representation, in implementation choice, 117–18
Requirement changes, 9–10
Requirements analysis, 135
Resource allocation, 308–9
Responsibilities
in behavior analysis, 133
of object, 83
Reuse, 314–16
Reusable role, 311, 323
Risk management, 304–5
Role
defined, 83
in interaction overview diagram, 218

S
Satellite Navigation System (SNS)
architecture for
activity definition, 348–50
Satellite Navigation System (SNS) (continued)
decomposing, 364–70
deployment of, 361–64
developing, 348
functionality allocation, 353
interface specifications, 358–61
logical, 341f
nonfunctional requirements, 358–61
validating, 350–57
component diagram for, 362f
context definition of, 336–39, 337f
launch time allocations for, 360f
package diagram for, 341f
post-transition phase, 371–73
requirements for, 334–36
use cases for
mission, 339–43
system, 343–47
Scenario, 281n
in weather monitoring system, 462–63
Schedule planning, 401–4
SCRUM lifecycle, 268
Segment use case, 357f–358t
Selector operation, 82
Semantic dependencies, 97–98
Semantics
of associations, 294
detailing, 297–99
of elements, 296–300
abstraction level in, 299
inheritance and, 299
functional, 115
time and space, 115–16
in Unified Modeling Language, 154–55
in weather monitoring system, 471f
Sensor data acquisition, 405–6
Sensor mechanism, in weather monitoring system, 480–82
Separation of concerns
in complex system, 13–14
encapsulation in, 52
in temperature measurement abstraction, 49
Sequence block strategy, 525–26
Sequence diagram
control constructs in, 210–12
destruction events in, 208
execution specification in, 209
interaction use in, 210
in knowledge source evaluation, 436f
lifelines and messages in, 206–7
purpose of, 206
scripts in, 209, 210f
in traffic management system, 392f
Sequence expression, 239–41
Sequential synchronization, 91
Server object, 89
Service Data Objects (SDO), 520, 523–24, 525f
Set theory, 128
Short-term memory capacity, 18
Simple name, 163
Simple state, 219–20
Simula, 39, 538
Single inheritance
among classes, 100–102
defined, 100
in object model, 58–61
Smalltalk
development of, 541–42
feature index of, 543t
method dispatch in, 105
MVC paradigm in, 143
purpose of, 542
SNS. See Satellite Navigation System
Social institutions, structure of, 7
Software architecture documentation, 278–80
Software complexity
defining, 7–8
development process management in, 10
discrete system behavior characterization in,
11–12
problem domain in, 8–10
Software development, 250–52. See also
Development process
Software evolution and maintenance, 10, 258–59
Software flexibility, 10–11
Software maintenance, 10
Software quality assurance, 316–17
Species diversity, 124
Staffing, 308–12
State
activity in, 222
client, management of, 499
composite, 224, 225f
defined, 219
nested, 224, 225f
notations for, 220–21
of object, 78–81
State machine, finite, 485–86
State machine diagram
activities in, 222
concurrency in, 225–30
control in, 225–30
initial, final, and simple states in, 219–20
purpose of, 218–19
submachine state in, 230
transition control in, 222–24
transitions and events in, 220–22
in vacation tracking system, 510f
in weather monitoring system, 471f
State transition diagram, 440f
Static binding, 66
Static typing, 66
Steady state, 149
Strong typing, 66
Structure diagrams, 149–50, 150f
Structured analysis, 136–38
Structured design, 21–22
Style sheets, 535
Subclass
defined, 100
purpose of, 101–2
Submachine state, 230
Subprograms, in composite design, 21
Substitution cipher, 414–15
Subsystems
defined, 170
for traffic management system, 408–10
Success scenario, 341–42
Sufficiency, 113
Superclass
collection, 477
defined, 100
in weather monitoring system, 457–58
Supplementary specification, 338
Synchronization
in communication diagram, 242
between objects, 91
Synchronous message, 208
Syntax, of Unified Modeling Language, 154–55
System administrator, 311
System architecture. See also Satellite Navigation System
for traffic management system, 407–8
System functions, in behavior analysis, 133
System use cases, 343–47
Systems engineering, 334

T
Task planning, 305–6
Team meetings, 305
Team roles, 309–12
Technology adoption, 325–26
Telemetry data, 99
Temperature measurement
abstraction in, 47–50
in weather monitoring system, 454–55
Template classes, 197–98
Testing, 314
Thomas Aquinas, 126
Time and space semantics, of operation, 115–16
Time-frame-based processing, 473–74, 474f
Timing diagrams, 231–35
TIOBE Programming Community Index, 538, 540f
Tools
in design methodology, 27
kinds of, 322–23
organizational implications of, 323–24
role of, 153
visual. See Diagrams and diagram notation
Toolsmith, 311, 323–34
Top-down structured design, 21
Topology of structured design, 21
Traffic management system
architecture for
block diagram in, 389f, 392
defining, 389–91
deployment diagram in, 394f
hardware and software allocations, 391–94
sequence diagram in, 392f
system functionality and, 385–87
class diagram in, 398, 399f
construction phase in, 396–97
decomposition in, 408–10
elaboration phase of, 385
inception phase of, 376
information display in, 404–5
key abstractions in, 395–96
Traffic management system (continued)
message passing in, 397–400
post-transition phase in, 411–12
release management in, 406–7
requirements for, 377–79
schedule planning in, 401–4
sensor data acquisition in, 405–6
software design for, 407–8
subsystem specification fin, 408–10
use cases determination, 378–84
Transition phase, 267
Transitions
completion, 221
controlling, 222–24
defined, 220
Type, vs. class, 64n
Type consistency, 66
Typing
benefits of, 65–66
meaning of, 64–66
static and dynamic, 65–66

U
Unified Modeling Language (UML)
background of, 148
classification of, 149–51, 150f
defined, 148
diagrams in. See Diagrams
evolution of, 154
models in, 148–49, 152–53
purpose of, 147–48
resources on, 155
subset use in, 151–52
syntax and semantics of, 154–55
Unqualified name, 163
URL redirection, 499
U.S. Global Positioning System (GPS), 334, 335
Use case model, 135
actors in, 176, 493
generalization in, 185
include and exclude relationships in, 180–84
noun-verb analysis in, 508
purpose of, 175–76
for satellite navigation system, 346f
mission, 339–43
segment, 357–358r
system, 343–47
specifying details in, 177–80
success scenario and, 341–42
for traffic management system, 378–84, 380f
for vacation tracking system, 492–93, 501–6
for weather monitoring system, 464–73
Web-centric systems and, 492–93
User experience model, 506–8, 507f
User interface mechanism
in vacation tracking system, 529–34
in weather monitoring system, 483–87
UX model, 506, 530

V
Vacation tracking system. See also Web applications
analysis and design models in, 508–19
communication diagram in, 516f
construction phase in, 506
controllers in, 528–29
deployment view in, 494–96, 495f
elaboration phase in, 494
entities in, 519–23
finder methods in, 528
implementation view of, 501
logical view in, 496–98, 497f
object diagram in, 514–15, 515f
post-transition phase in, 534–35
primary key generation in, 525–27
process view in, 498–501
requirements of, 490–92
rule types in, 511
service data objects in, 523–24, 525f
state machine diagram in, 510f
technologies in, 517
use case model in, 492–93
user experience model in, 506–8
Version control, 312–13
Virtual case file system, 303
Virtual machine abstraction, 45
Visibility
in class diagram, 198–200, 199f
in class interface, 95
of elements, 157–58
between objects, 89–91
in package diagram, 157–58
in relationship choice, 117
von Neumann architectures, 39
Weak typing, 66
Weather monitoring system
   architectural framework of, 473–74, 475f
   construction phase in, 474
   display mechanism in, 482–83
   elaboration phase of, 463
   frame mechanism in, 474–79
   hardware platform for, 451, 452/
   post-transition phase in, 487–88
   release planning for, 479–80
   requirements for, 450
   scenarios of use in, 462–63
   sensor class hierarchy in, 460/
   sensor mechanism in, 480–82
   superclass in, 457–58
   temperature sensors in, 454–55
   time and date class in, 452–54
   timer class in, 461–62, 462f
   use cases in, 464–73
   user interface mechanism in, 470f, 483–87
Web applications, 489–90. See also Vacation
   tracking system
      architecture of, 494
      business logic in, 533–34
      changes in, 535
      client state management in, 499
      logical components in, 496
      processes in, 498
      server-side software in, 535
      user interface in, 529–34
Web page design, 529–34
White-box activity diagram, 356f, 365, 367f

XP lifecycle, 267–68

Zooming in and out, 225