

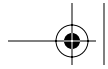


Index

A

- A (abstractness) metric, 432, 456
- Abbott, Edwin A., 331
- Abstract classes
 - in class diagrams, 250–251
 - for Open/Closed Principle, 430
- ABSTRACT SERVER pattern, 496–497
- AbstractDbGatewayTest class, 536
- Abstractions
 - in CoffeeMaker, 265–266, 270–279
 - in Dependency-Inversion Principle, 154, 156–159
 - metrics for, 432
 - in Open/Closed Principle, 123–124, 128–131, 430
 - in payroll system, 360–363
 - for repetition reduction, 106
 - in Stable Abstractions Principle, 431–435
- Abstractness (A) metric, 432, 456
- AbstractTransactions class
 - class allocation in, 458, 464
 - metrics for, 465
- Accept method
 - Assembly, 553–554
 - ErnieModem, 550
 - HayesModem, 550
 - Modem, 544, 548
 - PiecePart, 555
 - ZoomModem, 551
- Acceptance tests
 - in extreme programming, 15–16
 - purpose of, 36–37
- Actions in state diagrams, 184
- Activations in sequence diagrams, 183, 226
- ACTIVE OBJECT pattern, 299, 305–310
- Active objects
 - in object diagrams, 213–217
 - in sequence diagrams, 240
- ActiveObjectEngine class, 305–308, 310
- Actors in use cases, 222
- Acyclic Dependencies Principle (ADP), 420–426
- ACYCLIC VISITOR pattern, 548–552
- ADAPTER pattern, 498
 - class-form of, 498–499
 - for modem problem, 499–505
- Add method
 - Assembly, 554, 572
 - CompositeShape, 468
 - Frame, 59–62
 - Game, 64, 74, 88, 90, 93
 - GameTest, 68, 70–71
 - PersistentSet, 146
 - Set, 145
 - TransactionContainer, 669
 - TreeMap, 179–180, 182–184
 - TreeNode, 180
- AddAction method
 - PayrollPresenterTest, 659
 - TransactionContainer, 668–669
- AddClassification method, 635
- AddCommand method, 306
- AddCommissionedEmployee class, 371, 386
- AddCommissionedEmployeeTransaction class, 352
- AddEmployee method
 - AddEmployeePresenter, 648, 650
 - AddEmployeePresenterTest, 644
 - AddEmployeeWindowTest, 652–653, 657
 - Blah, 607–610
 - InMemoryPayrollDatabase, 606
 - PayrollDatabase, 368, 605
 - in SqlPayrollDatabase, 608, 614–616, 618–620, 622–624





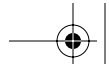
- AddEmployee method (*Continued*)
 - in SqlPayrollDatabaseTest, 611, 613
- AddEmployee_Static method, 605
- AddEmployeeAction method
 - PayrollPresenterTest, 659
 - PayrollWindowTest, 666
- AddEmployeeActionInvoked method, 661
- addEmployeeButton_Click method, 655
- addEmployeeMenuItem_Click method, 667
- AddEmployeePresenter class, 641–642, 645–650
- AddEmployeePresenterTest class, 642–644
- AddEmployeeTransaction class
 - dynamic model of, 369–370
 - fields in, 302–304
 - hierarchy of, 352
 - listing, 370–371
 - PayrollDatabase for, 368
 - static model of, 367
 - for user interface, 641, 648
- AddEmployeeView interface, 642, 649
- AddEmployeeWindow class, 641–642, 653–657
- AddEmployeeWindowTest class, 650–653, 656–657
- AddExtension method, 570
- AddHourlyEmployee class, 371, 382
- AddHourlyEmployeeTransaction class, 352, 460
- Adding employees
 - into databases, 607–617
 - into payroll system, 302–304, 352–353, 366–371
- AddingTransaction method, 668
- AddingTransactionTriggersDelegate method, 668
- AddItem method
 - AddItemTransaction, 509
 - Order, 508, 510, 534
 - OrderImp, 521
 - OrderProxy, 522
- AddItemTransaction class, 509
- AddPaymentMethod method
 - LoadEmployeeOperation, 630
 - LoadEmployeeOperationTest, 628
 - LoadPaymentMethodOperation, 635
- Address class, 253–254, 257
- Address property
 - AddEmployeePresenter, 645–646
 - Employee, 407
- addressTextBox_TextChanged method, 654
- AddSalariedEmployee class, 371
- AddSalariedEmployeeTransaction class, 352
- AddSalariedTransaction method, 366–368
- AddSchedule method
 - LoadEmployeeOperation, 627–628
 - LoadPaymentMethodOperation, 635
- AddServiceCharge method, 379
- AddSubNode method, 180
- AddThrow method, 88, 94
- addTimeSink method, 479
- AddTransition method, 585–586
- AdjustCurrentFrame method, 70–72, 74–77, 88, 90–93
- AdjustFrameForStrike method, 91–92
- ADP (Acyclic Dependencies Principle), 420–426
- AdvanceFrame method, 91–92, 94
- Afferent coupling (Ca), 427–428, 456
- Affiliation class, 389–392
 - in Common Closure Principle, 450
 - in Employee, 357
 - for service charges, 379–380
 - structure of, 362–363
 - for union dues, 403
- Affiliation property, 408
- Affiliation table, 605
- Affiliations class
 - in Common Closure Principle, 450
 - merging into PayrollImplementation, 462
 - metrics for, 459
 - in Reuse/Release Equivalence Principle, 452
- AffiliationTransactions class
 - class allocation in, 458
 - metrics for, 459





- Aggregation in class diagrams
 - associations, 252
 - composition, 253–254
 - multiplicity, 254–255
 - Agile Alliance, 4–5
 - Agile design, 103–104
 - Copy program, 108–113
 - and rotting software, 104–107
 - Alarm method
 - Turnstile, 341–342, 589
 - TurnstileFSM, 592
 - AllInfosCollected method, 643, 648
 - AllInformationIsCollected method, 647
 - Animal class, 178
 - Anticipation in Open/Closed Principle, 128–129
 - Application class, 312–314
 - class allocation in, 458, 464
 - metrics for, 459, 465
 - in Reuse/Release Equivalence Principle, 452
 - Application interface, 320
 - ApplicationRunner class, 319–320
 - Architecture, serendipitous, 37–38
 - AsDouble method, 655
 - AsInt method, 655
 - Assemblies, 416
 - Assembly class, 553–554, 571–572
 - Assembly method
 - BOMReportTest, 557
 - BomXmlTest, 567
 - AssemblyOfAssemblies method
 - BOMReportTest, 557
 - BomXmlTest, 567
 - AssemblyWithPartsCSV method, 570
 - AssemblyWithPartsXML method, 569
 - AssertSinkEquals method
 - ClockDriverTest, 481
 - ObserverTest, 488
 - Associations
 - in class diagrams, 181, 245
 - aggregation, 252
 - classes, 256–257
 - horizontal, 247
 - qualifiers, 257
 - stereotypes, 255–256
 - in CoffeeMaker, 267
 - Asynchronous messages in sequence diagrams, 235–239
 - AsynchronousLogger class, 235–241
 - ATM system
 - class diagrams for, 247–249
 - user interface example, 169–174
 - Aurelius, Marcus, 349
 - Author support, 417
 - Automated acceptance tests, 36
 - Axes of change in Single-Responsibility Principle, 118
- ## B
- Back-end documentation, 192
 - BadExtension method, 570
 - BadPartExtension class, 575
 - Behaviors in diagrams, 194–196
 - Bills of materials (BOM), 552–553
 - Binary units, 416
 - BinaryTreeNode class, 254
 - Blah class, 607–610
 - Boiler class, 263–264
 - BoilerEmptiesWhilePotRemoved method, 291
 - BoilerEmptyPotNotEmpty method, 291
 - BOM (bills of materials), 552–553
 - BOMReportTest class, 556–559
 - BomXmlTest, 567–570
 - Booch, Grady, 154–155, 250
 - Bottom-up design vs. top-down, 424–426
 - BoundedSet set, 144
 - Bowling game, 56–98
 - Brew button, 268, 271–272
 - Brewer, E. Cobham, 115
 - BRIDGE pattern, 503–505
 - Bubble sort, 316–319, 321–322
 - BubbleSorter class, 190–191, 316–319, 321–322





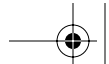
Budgets, developer, 27
 Buildability of applications, 425
 BuildProductQueryCommand method, 515
 Burke, Edmund, 579
 Burn-down charts, 28–29
 Business decisions, 25
 Business rules
 in persistence, 119
 in user interface, 642
 Businesspeople, collaboration with, 9
 Button class
 associations with, 245
 for cellular phones, 194–198
 for CoffeeMaker, 263–264
 for Dependency-Inversion Principle,
 157–159
 ButtonDialerAdapter class, 198–199
 ButtonListener interface, 197–198
 buttonPressed method, 197–199
 Buttons in CoffeeMaker, 263–264, 268,
 271–272
 ButtonServer interface, 159

C

C# programmers, UML for, 177–185
 Ca (afferent coupling), 427–428, 456
 CalcMaxPrimeFactor method, 48
 CalculateDeductions method, 404, 406–407
 CalculatePay method, 393
 defining, 395
 in HourlyClassification, 399, 401, 403
 CalculatePayForTimeCard method, 402
 Calculating bowling scores, 56–98
 Calling hierarchy in collaboration diagrams,
 184
 Cancelable superstate, 206
 CASE tools, 201
 CashDispenser class, 247
 CCP (Common Closure Principle), 419
 applying, 450–451
 DECORATOR for, 561
 in dependency cycles, 425
 for stability, 426

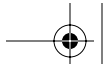
Ce (efferent coupling), 427–428, 456
 Cellular phones
 code for, 198–199
 collaboration diagrams for, 194–196
 diagram evolution for, 199–200
 state diagrams for, 194
 Celsius/Fahrenheit conversions, 313–315,
 320–321
 Change method
 ChangeClassificationTransaction, 387
 ChangeEmployeeTransaction,
 383–384, 391
 ChangeNameTransaction, 384
 ChangeAddressTransaction class, 381,
 383–385
 ChangeAffiliationTransaction class, 381,
 389, 391
 ChangeClassificationTransaction class, 381,
 384, 386–387, 449–450
 ChangeCommissionedTransaction class,
 386, 388
 ChangeDirectTransaction class, 388
 ChangeEmployeeTransaction class,
 381–384
 ChangeHoldTransaction class, 389
 ChangeHourlyTransaction class, 385–388,
 460
 ChangeMailTransaction class, 389
 ChangeMemberTransaction class, 390–392
 ChangeMethodTransaction class, 381–382,
 388
 ChangeNameTransaction class, 381–384
 Changes
 Copy program handling of, 109–111
 employee, 356–358, 381–393
 in Open/Closed Principle, 129–130
 requirements, 9
 responding to, 7–8
 and rigidity and fragility in design, 105
 rotting software from, 107
 as software module method, 42
 ChangeSalariedTransaction class, 385, 388
 ChangeUnaffiliatedTransaction class,
 390–393



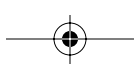


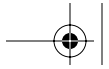
- ChangeUnionMember method, 390–391
- Characters, Copy program for, 108–113
- Charts. *See also* Class diagrams; Diagrams
 - burn-down, 28–29
 - velocity charts, 28
- CheckButton method, 272, 275, 277
- CheckMessagesFlowToLog method, 237
- CheckQueuedAndLogged method, 237
- CheckSavedPaymentMethodCode method, 613
- CheckSavedScheduleCode method, 612
- CheckSubmitEnabled method, 644
- CheckThatProductsMatch method, 537–538
- CheckWriter class, 33–34
- ChgEmp transaction, 366
- Child method, 568
- ChildText method, 568
- Circle class, 131, 137, 438, 467–468
- Circle structure, 125–127
- Clarity in design, 107
- Class diagrams, 180–181, 243–244
 - associations in, 181, 245
 - aggregation, 252
 - classes, 256–257
 - horizontal, 247
 - qualifiers, 257
 - stereotypes, 255–256
 - for ATM system, 247–249
 - for cellular phones, 196–198
 - classes in, 243–244
 - abstract, 250–251
 - association, 256–257
 - nested, 256
 - composition in, 253–254
 - inheritance in, 246–248
 - multiplicity in, 254–255
 - properties in, 251–252
 - stereotypes in, 249–250
- Class-form adapters, 498–499
- Class interfaces vs. object interfaces in ISP, 166
- Class utilities, 250
- Classes
 - abstract, 250–251, 430
 - Common Closure Principle for, 419
 - Common Reuse Principle for, 418–419
 - container, 144–147
 - degenerate, 548
 - dependencies in, 416
 - god, 266
 - Reuse/Release Equivalence Principle for, 417–418
 - vapor, 265
- Classification, changing, 384–393
- Classification property
 - in ChangeHourlyTransaction, 387–388
 - in Employee, 407
- Classifications class, 448–449
 - in Common Closure Principle, 450
 - couplings in, 454
 - dependence on, 460
 - merging into PayrollImplementation, 462
 - metrics for, 457, 459
 - in Reuse/Release Equivalence Principle, 452
- ClassificationTransaction class
 - dependence on, 460
 - metrics for, 457, 459
- Cleanup method
 - FtoCStrategy, 321
 - FtoCTemplateMethod, 314–315
- Clear method
 - ItemData, 525
 - TransactionContainer, 669
 - TreeMap, 228
- ClearEmployeeTable method, 612
- Client class, 123
- ClientInterface class, 123–124
- Clients
 - in Interface Segregation Principle, 165–166
 - in Open/Closed Principle, 123
- Clock interface, 473–474, 483, 485–487
- Clock project, 472–491
- ClockDriver class, 473–474, 477–480
- ClockDriverTest class, 473–476, 479, 481
- ClockObserver interface, 477–481





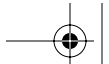
- ClockSink class, 474
- Close method
 - AbstractDbGatewayTest, 536
 - Db, 515
 - SocketServer, 214
- ClutchOnCommand method, 301
- Cockburn, Alistair, 221
- CoffeeMaker class, 259–260
 - abstraction in, 265–266, 270–279
 - hideous solution for, 263–265
 - improved solution for, 266–267
 - object-oriented design for, 279–291
 - specifications for, 260–263
 - user interface for, 267–270
- CoffeeMakerAPI interface, 261–265, 270–274
- CoffeeMakerStub class, 286–288
- Cohesion
 - of components, 420
 - Common Closure Principle, 419
 - Common Reuse Principle, 418–419
 - Reuse/Release Equivalence Principle, 417–418
 - considerations for, 461–463
 - metrics for, 455–457
 - in SRP. *See* Single-Responsibility Principle (SRP)
- Coin method
 - Locked, 596
 - LockedTurnstileState, 587
 - State, 595
 - Turnstile, 341, 588, 595
 - Unlocked, 595
 - UnlockedTurnstileState, 588
- Coincidence, programming by, 70
- CoinInLockedState method
 - SMCTurnstileTest, 597
 - TurnstileTest, 582–583
- CoinInUnlockedState method
 - SMCTurnstileTest, 597–598
 - TurnstileTest, 582–583
- Coins property, 341
- Collaboration
 - with businesspeople, 9
 - vs. negotiation, 6–7
- Collaboration diagrams
 - for cellular phones, 194–196
 - for relationships, 183–184
- Collective ownership, 17
- Command interface, 300, 469
- COMMAND pattern, 299–300
 - for decoupling, 304
 - for DeleteEmployeeTransaction, 372
 - for simple commands, 300–302
 - for transactions, 302–304, 366
 - Undo variation, 304–305
- Command property, 631, 633
- Commission property, 647
- CommissionedClassification class, 356, 403–404
- CommissionedClassification table, 605
- commissionRadioButton_CheckedChanged method, 654
- CommissionSalary property, 647
- commissionSalaryTextBox control, 656
- commissionSalaryTextBox_TextChanged method, 655
- commissionTextBox control, 656
- commissionTextBox_TextChanged method, 655
- Common Closure Principle (CCP), 419
 - applying, 450–451
 - DECORATOR for, 561
 - in dependency cycles, 425
 - for stability, 426
- Common Reuse Principle (CRP), 418–419
- Communication
 - diagrams for, 189–191
 - as software module method, 42
- Company class, 256–257
- Compare method, 132
- CompareAndSwap method, 190, 316
- CompareTo method, 130–131
- Compartments in class diagrams, 244
- Compile-time safety, 441





- Complete builds, 423
- Complete method, 280
- CompleteCycle method, 281
- Complexity in design, 106
- Component dependency graphs, 420
- Component-dependency structures, 425
- Component diagrams, 421–422
- Components, 416
 - cohesion of, 420
 - Common Closure Principle, 419
 - Common Reuse Principle, 418–419
 - Reuse/Release Equivalence Principle, 417–418
 - stability principles, 420
 - Acyclic Dependencies Principle, 420–426
 - Stable Abstractions Principle, 431–435
 - Stable-Dependencies Principle, 426–431
 - structure and notation for, 448–450
- COMPOSITE pattern, 467–468
 - for association stereotypes, 256
 - commands for, 469
 - multiplicity in, 470
- CompositeCommand class, 469
- CompositeShape class, 467–468
- Composition in class diagrams, 253–254
- Comprehensive documentation, 6
- ComputationalGeometryApplication class, 117
- Conceptual-level diagrams, 178–179
- Concrete classes in Dependency-Inversion Principle, 157
- Conditions in sequence diagrams, 232–233
- Configuration data structures, 559
- configureForUnix method, 544
- Constraints, 4
- Construction in sequence diagrams, 183
- Construction method, 668
- Container classes, 144–147
- ContainerAvailable method, 284
- ContainerUnavailable method, 284
- ContainmentVessel class, 266–270, 273, 283–284
- Context class, 590
- Contingencies, preparing for, 106
- Continuous delivery of software, 8
- Continuous integration, 17–18
- Contract negotiation, 6–7
- Contracts in Liskov Substitution Principle, 143–144
- Controls, events on, 656
- Conveniently callable software, 32
- Conventions in Liskov Substitution Principle, 150–151
- Conversions
 - Fahrenheit to Celsius, 313–315, 320–321
 - in MONOSTATE, 338
- Cooley, Mason, 325
- Coolidge, Calvin, 437
- Copier class, 109–112
- Copy method, 109–112
- Copy program, 108–113
- Core model in payroll system, 358
- Cost property
 - in ExplosiveCostExplorer, 555
 - in PiecePart, 555, 571
- Couplings
 - components. *See* Stability
 - factories for, 460–463
 - metrics for, 427–428, 455–457
 - in package analysis, 454–455
 - separating, 119
 - types of, 427–428
- CreateAssembly method
 - BOMReportTest, 557
 - BomXmlTest, 567
- CreateDirectDepositMethod method, 633
- CreateDirectDepositMethodFromRow method, 630–631
- CreateEmployee method, 626
- CreateHayes method, 563
- CreateHoldMethod method, 633





CreateInsertDirectDepositCommand
 method, 621

CreateInsertMailMethodCommand method,
 621

CreateMailMethod method, 634

CreateMailMethodFromRow method, 634

CreatePart method
 BOMReportTest, 557
 BomXmlTest, 567

CreatePaychecks method, 38

CreatePaymentMethod method, 631–632

CreateTransaction method, 648

CreatingTransaction method, 644, 650

Creation method
 AddEmployeePresenterTest, 642–643,
 648
 PayrollPresenterTest, 659

Cross-platform support, SINGLETON for,
 334

Crossed wires, 267

CrossOutMultiples method, 47–48, 51

CrossOutputMultiplesOf method, 48, 52

CRP (Common Reuse Principle), 418–419

CsvAssemblyExtension class, 575

CsvPartExtension interface, 574

CsvPiecePartExtension class, 574

CsvText property
 CsvAssemblyExtension, 575
 CsvPiecePartExtension, 574

CurrentFrame property, 69–73, 77, 87,
 91–92

CustomerId property
 Order, 533
 OrderImp, 521
 OrderProxy, 522

Customers
 collaboration with, 6–7
 in extreme programming, 14

Cut and paste, repetition from, 106

Cycles
 dependency, 421–426
 in extreme programming, 15

D

D metrics for main sequence, 434–435, 457

DAGs (directed acyclic graphs), 422

Dashes (-) in class diagrams, 244

Data access object (DAO), 528

Data-driven approach in Open/Closed
 Principle, 131–132

Data structures, configuration, 559

Data tokens in sequence diagrams, 183, 226

Database property, 661

DatabaseFacade class, 540

Databases
 as implementation detail, 350–351
 patterns with, 539–540
 for payroll. *See* PayrollDatabase class
 third-party, 526–528

DataSet class, 626

Date method, 376

DateUtil class, 406–407

Db class, 325–326, 346–347, 514–515

DB gateways, 535–539

DBC (design by contract), 142

DbGateways class, 535

DbOrderGateway class, 528–529, 536

DbOrderGatewayTest class, 538–539

DbProductGateway class, 531–532, 536

DbProductGatewayTest class, 536–538

DbTest class, 513, 520

DeclareComplete method, 283

DeclareDone method, 281

DECORATOR pattern
 for association stereotypes, 256
 with databases, 539–540
 for modem problem, 560–565

Decoupling
 physical and temporal, 304
 serendipitous, 36

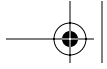
DedicatedHayesModem class, 503–504

DedicatedModem class, 500–502

DedicatedModemAdapter class, 502

DedModemController class, 504



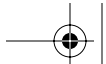


Index

707

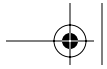
- DedUsers class, 500–502
- Degenerate classes, 548
- DelayAndRepeat method, 309–310
- DelayedTyper class, 308–310
- Delegation
 - separation through, 167–168
 - STRATEGY for. *See* STRATEGY pattern
 - TEMPLATE METHOD for. *See* TEMPLATE METHOD pattern
- Delete method, 145
- DeleteEmployeeTransaction class, 353, 372–373
- DeleteProductData method, 515
- Deleting employees, 353, 372–373
- Delivery of software, 8
- DeMarco, Tom, 116, 455
- Dependencies
 - in agile design, 113
 - class diagrams for, 243–244, 246
 - in classes, 416
 - in CoffeeMaker, 277–278
 - in FACTORY, 440–441
 - in modem problem, 501–502
 - from static typing, 441
- Dependency cycles, eliminating, 421–426
- Dependency-Inversion Principle (DIP), 153–154, 424
 - abstractions in, 156–159
 - in CoffeeMaker, 271
 - in Copy program, 113
 - in furnace example, 160–161
 - layering in, 154–155
 - in modem problem, 499
 - in OBSERVER, 493
 - ownership inversion, 155–156
 - simple example, 157–158
 - in table lamp, 496
 - violations of, 437–438
- Dependency management metrics, 416
- Dependent components, 428
- Deposit method, 342
- DepositTransaction class, 169–171
- DepositUI interface, 170–173
- Derivability in MONOSTATE, 337
- Deriving vs. factoring, 148–150
- Description property
 - Assembly, 554, 572
 - PiecePart, 555, 571
- Design
 - agile, 103–104
 - Copy program, 108–113
 - and rotting software, 104–107
 - attention to, 10
 - before coding, 188–189
 - in extreme programming, 19–20
 - large systems, 416
 - top-down vs. bottom-up, 424–426
- Design by contract (DBC), 142
- Desktop applications, 638
- Details
 - in Common Closure Principle, 451
 - in payroll system, 356–358
- DetermineIterationLimit method, 52
- Developers
 - budgets for, 27
 - and business decisions, 395–396
 - in extreme programming, 14
- Diagrams, 177–180
 - appropriateness of, 200–201
 - as back-end documentation, 192
 - for bowling game, 57
 - class. *See* Class diagrams
 - collaboration
 - for cellular phones, 194–196
 - for relationships, 183–184
 - for communication, 189–191
 - discarding, 192–194
 - effective use of, 189
 - iterative process for, 194–200
 - object, 182, 211
 - active objects in, 213–217
 - purpose of, 211–212
 - purpose of, 187–189
 - as road maps, 191–192
 - sequence. *See* Sequence diagrams
 - state. *See* State diagrams





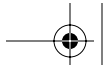
- Diagrams (*Continued*)
 - use cases, 222
 - Dial method
 - DedModemController, 504
 - DialModemController, 504
 - HayesModem, 562
 - LoudDialModem, 563, 565
 - Modem, 118, 561
 - ModemDecorator, 565
 - Dialer class
 - for cellular phones, 194–198
 - class diagram for, 244–245
 - DialForReal method, 561
 - dialImp class, 504
 - DialModem class, 503–504
 - DialModemController class, 504
 - Diderot, Denis, 299
 - Digital clock project, 472–491
 - DigitalClock interface, 473–474, 492–493
 - DIP. *See* Dependency-Inversion Principle (DIP)
 - Direct dependency, 155
 - DirectDepositAccount table, 605, 613, 615–616
 - DirectDepositMethod, 613, 618, 628, 630
 - DirectDepositMethodGetsSaved method, 614
 - Directed acyclic graphs (DAGs), 422
 - Directed edges, 421
 - Directed graphs, 421
 - Direction of dependencies, 113
 - DirectMethod class, 357
 - DisplayTime method, 472
 - Dispose method
 - AddEmployeeWindow, 653–654
 - PayrollWindow, 666–667
 - Distance from main sequence, 434–435, 457
 - Distributed processing, 601–602
 - DLLs, components in, 419
 - Do method, 469
 - Documentation
 - acceptance tests as, 36
 - comprehensiveness of, 6
 - diagrams as, 192
 - size of, 202
 - source code listings as, 103
 - Dog class, 178
 - “Done”, defining, 26
 - Done method
 - Application, 314
 - ContainmentVessel, 283
 - HotWaterSource, 281
 - M4UserInterface, 281
 - UserInterface, 280
 - Door interface, 163–165, 168
 - DoorTimeOut method, 167
 - DoorTimerAdapter, 167–168
 - DoPayroll method, 230
 - DoSort method, 317
 - Dot structure, 184
 - DoubleBubbleSorter class, 318–319
 - Draw method
 - Circle, 137
 - CompositeShape, 467
 - Shape, 127, 467
 - Square, 137
 - DrawAllShapes method, 125–128, 130, 132
 - DrawCircleCommand class, 304–305
 - Drawing rectangles, 600
 - DrawShape method, 137
 - DrawSquare method, 125
 - Dual dispatch technique, 544, 548
 - Duplication of code, 20
 - Dynamic diagrams, 179
 - Dynamic typing vs. static, 441–442
- E**
- Early delivery of software, 8
 - Efferent coupling (Ce), 427–428, 456
 - Efficiency
 - of MONOSTATE, 338
 - of SINGLETON, 334
 - Eiffel language, 143
 - EmpId property, 645
 - empIdTextBox_TextChanged method, 654





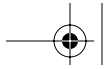
- Employee class, 33–35, 357–359
 - for affiliations, 391
 - associations with, 257
 - for ChangeMethodTransaction, 388
 - for changing employees, 382
 - in Common Closure Principle, 450
 - Hashtable for, 368
 - implementation, 407–408
 - inheritance from, 246
 - with NULL pattern, 346–348
 - for persistence, 119
 - in Reuse/Release Equivalence Principle, 452–453
 - in sequence diagrams, 227
- Employee table, 604–605
- EmployeeAffiliation table, 605
- EmployeeDatabase class, 33–34, 448
- EmployeeDB class, 227
- EmployeeFactory class, 442–443
- EmployeeImplementation class, 346
- EmployeeObserver class, 492
- Employees in payroll system
 - adding, 302–304, 352–353, 366–371
 - changing, 356–358, 381–393
 - database for. *See* PayrollDatabase class
 - deleting, 353, 372–373
 - paying, 393–408
- EmployeesText method, 666
- EmployeesText property
 - MockPayrollView, 662
 - PayrollWindow, 667
- EmployeeTest class, 347
- EmptyPotReturnedAfter method, 291
- EnablingAddEmployeeButton method, 652, 657
- EnablingCommissionFields method, 652, 656
- EnablingHourlyFields method, 652, 656
- EnablingSalaryFields method, 652, 656
- Encapsulation, 454–455
- enterSub action, 206
- enterSuper action, 206
- Entry events in state transition diagrams, 205–207
- Environment, viscosity of, 105–106
- Equals method
 - ItemData, 523
 - ProductData, 514
- EraseMembership method, 393
- ErnieForUnix method, 547, 552
- ErnieModem class, 546, 550
- ErnieModemVisitor interface, 550
- ErniesModem class, 499, 503–504
- EventHandler method, 656
- Events
 - on controls, 656
 - in state diagrams, 184, 205–207
 - use cases for, 220
- Evolving patterns, 471–491
- Exclusion zones, 432–433
- Execute method, 304–305
 - AddEmployeeTransaction, 369, 371
 - ChangeEmployeeTransaction, 382–384
 - Command, 300–301, 306
 - DeleteEmployeeTransaction, 372–373
 - DepositTransaction, 172
 - LoadPaymentMethodOperation, 630–632, 635
 - PaydayTransaction, 397, 403–404
 - SaveEmployeeOperation, 623
 - ServiceChargeTransaction, 380
 - SleepCommand, 308
 - StopCommand, 309–310
 - TestSleepCommand, 306
 - TimeCardTransaction, 375–377
 - Transaction, 171, 366, 448
- ExecuteCommand method, 620
- ExecuteSql method, 536
- exit events in state transition diagrams, 205–207
- exitSub2 action, 206
- exitSuper action, 206
- ExplodedCost method, 559
- ExplodedCost property, 553
- ExplodedCostVisitor class, 555
- EXTENSION OBJECT pattern, 539, 565–576
- Extensions in Open/Closed Principle, 122





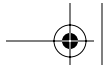
- ExtractItemDataFromResultSet method, 524
- Extreme programming, 13
 - acceptance tests in, 15–16
 - collective ownership in, 17
 - continuous integration in, 17–18
 - metaphors in, 21–22
 - open workspace in, 18
 - pair programming in, 16
 - planning in, 19
 - refactoring in, 17, 20–21
 - short cycles in, 15
 - simple design in, 19–20
 - sustainable pace in, 18
 - test-driven development in, 16
 - user stories in, 14
 - whole teams in, 14
- F**
- FACADE pattern, 325–326
 - with databases, 539–540
 - PayrollDatabase class, 368
 - for refactoring, 119
 - vs. SINGLETON, 334
 - with TABLE DATA GATEWAY, 528
- Face-to-face conversations, 9
- factoring in Liskov Substitution Principle, 148–150
- FACTORY pattern and factories, 437–438
 - for couplings, 460–463
 - dependencies in, 440–441
 - for fixture tests, 443–444
 - importance of, 444–445
 - initializing, 461
 - for payroll window, 658
 - static vs. dynamic typing, 441–442
 - substitutable, 442–443
- Factory Method pattern, 369
- Fahrenheit/Celsius conversions, 313–315, 320–321
- FanSwitch class, 496–497
- Fat interfaces. *See* Interface Segregation Principle (ISP)
- Feedback
 - customer, 7
 - of velocity, 26
- Final pseudostates in state transition diagrams, 207–208
- Find method
 - DbOrderGateway, 530
 - DbOrderGatewayTest, 538–539
 - DbProductGateway, 532
 - DbProductGatewayTest, 537
 - InMemoryOrderGateway, 531
 - InMemoryProductGateway, 532
 - TreeNode, 180
- FindSubNodeForKey method, 180
- Finite state machines, 579–580
 - for CoffeeMaker, 279
 - diagrams for, 208–209
 - high-level application policies for GUIs, 598–600
 - Monostate implementation of, 338–343
 - nested switch/case statements for, 580–584
 - state diagrams for. *See* State diagrams
 - UML notation for, 184
- First Law of Documentation, 6
- FitNesse tool, 37
- Fixtures, testing, 443–444
- Floor plans, 212
- Forms, 639–640
- Fowler, Martin, 41, 177
- Fragility in design, 105
- Frame class, 58–62, 64
- FrameTest class, 58–59
- Friendly, Fred W., 467
- FSMError class, 596–597
- FSMs. *See* Finite state machines
- ftoc program, 313
- FtoCraw class, 313
- FtoCStrategy class, 319–321
- FtoCTemplateMethod class, 314–315
- Function matrices, 548, 552
- Functional decompositions, 424
- Furnace example, 160–161



**G**

- Game class, 61–62, 64–65, 73–74, 87–88, 92–94, 116
- GameTest class, 61–63, 66–73, 95–98
- Gamma, Erich, 312
- Gateways
 - example, 528–535
 - with Facade, 326
 - testing, 535–539
- GdatabaseFactory class, 444
- Generality metric, 456
- Generalization, 246
- GeneralTransactions class
 - class allocation in, 458
 - metrics for, 459
- GeneratePrimeNumbers method, 43–45, 47–48, 51
- GeneratePrimes program
 - final version, 49–52
 - refactoring, 45–49
 - testing, 52–53
 - unit testing, 44
 - version 1, 42–44
 - version 2, 45–46
 - version 3, 47
 - version 4, 48–49
 - version 5, 49
- GeneratePrimesTest class, 44, 52–53
- GeometricRectangle class, 117
- Get method, 179–180
- GetBoilerStatus method
 - CoffeeMakerAPI, 262
 - CoffeeMakerStub, 287
- GetBrewButtonStatus method
 - CoffeeMakerAPI, 262, 271–272, 275
 - CoffeeMakerStub, 287
- GetClassification message, 384
- GetCountForPart method, 556
- GetCurrentState method, 594
- GetCurrentStateName method, 594
- GetEmployee method
 - DB, 346–347
 - PayrollDatabase, 369
 - in sequence diagrams, 227
- GetExtension method, 566
 - Part, 570
 - PiecePart, 576
- GetHashCode method, 514
- GetHours method
 - MockTimeSink, 476, 480, 490
 - MockTimeSource, 489
- GetItemsForOrder method, 524
- GetItsLockedState method, 594
- GetItsUnlockedState method, 594
- GetMinutes method
 - MockTimeSink, 476, 480, 490
 - MockTimeSource, 489
- GetOrderData method, 524
- GetPayPeriodStartDate method, 408
- GetProductData method, 514–515
- GetScore method, 61
- GetSeconds method
 - MockTimeSink, 476, 480, 490
 - MockTimeSource, 490
- GetVersion method, 594
- GetWarmerPlateStatus method
 - CoffeeMakerAPI, 262
 - CoffeeMakerStub, 287
- God classes, 266
- GoodStart method, 289
- Granularity principles, 417–420
- GraphicalApplication class, 117
- Graphs
 - burn-down charts, 28–29
 - diagrams. *See* Diagrams
 - directed, 421–422
- Grenning, James, 226
- Guards in sequence diagrams, 183
- Guillemet characters («») in class diagrams, 249
- GUIs
 - desktop applications, 638
 - high-level application policies for, 598–600
 - interaction controllers, 600–601





GUITransactionSource class, 409

H

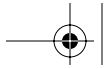
H (relational cohesion) metric, 456
 HandleBrewingEvent method, 285
 HandleEvent method, 580–581, 584–585
 HandleIncompleteEvent method, 285
 HandleSecondThrow method, 80–85
 hangImp class, 504
 hangup method
 DedModemController, 504
 DialModemController, 504
 Modem, 118
 Hashes (#) in class diagrams, 244
 Hashtable class, 368
 HayesForUnix method, 547, 552
 HayesModem class, 499, 503–504, 546, 550, 562–563
 HayesModemVisitor interface, 550
 Heater interface, 160–161, 263, 265
 Height property, 138–141
 Heine, Heinrich, 3
 Helm, Richard, 312
 Heuristics
 in CoffeeMaker. *See* CoffeeMaker class
 in Liskov Substitution Principle, 150–151
 hideLoginScreen action, 204
 High-level application policies, 598–600
 High-level design placement, 429–431
 High-level modules, 154
 HoldMethod class, 357
 HoldMethod method, 622, 630
 Hollywood principle, 155
 Hooks in Open/Closed Principle, 129–130
 Horizontal associations in class diagrams, 247
 HotWaterSource class, 266–270, 273–274, 281–282
 Hourly employee payments, 398–402
 HourlyClassification class, 356, 374–375, 386–387, 403–404, 460

HourlyClassification table, 605
 hourlyRadioButton_CheckedChanged method, 654
 HourlyRate property, 646
 hourlyRateTextBox control, 656
 hourlyRateTextBox_TextChanged method, 655
 HourlyUnionMemberServiceCharge method, 405
 Hours method, 376
 HttpRequest class, 189–190
 HttpResponse class, 189–190
 Human attention, refactoring in, 41–42
 example. *See* GeneratePrimes program
 in extreme programming, 17, 20–21
 Hunt, Andy, 70
 Hunt the Wumpus program, 32–33

I

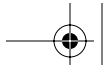
I (instability) metric, 427–428, 456
 IComparable interface, 130, 180
 Id property, 533
 IDataReaderExecuteQueryStatement method, 515
 Idle method
 FtoCStrategy, 321
 FtoCTemplateMethod, 314–315
 Imaginary abstraction, 265–266
 Immobility in design, 105
 Implementation-level diagrams, 178–179
 In-memory TDGs, 535–539
 Independent components, 427–428, 450–451
 Individuals in agile development, 5
 InformCashDispenserEmpty method, 247
 Inheritance, 497
 in class diagrams, 246–248
 separation through, 168
 with SINGLETON, 334
 STRATEGY for. *See* STRATEGY pattern
 TEMPLATE METHOD for. *See* TEMPLATE METHOD pattern





- Inheritance relationships, 178
- Init method
 - ContainmentVessel, 283
 - Db, 514
 - FtoCStrategy, 321
 - FtoCTemplateMethod, 314
 - HotWaterSource, 281
 - M4CoffeeMaker, 276–277
 - UserInterface, 280
- Initial pseudostates in state transition diagrams, 204, 207–208
- InitialConditions method
 - SMCTurnstileTest, 597
 - TestCoffeeMaker, 288
 - TurnstileTest, 582
- Initialization programs, 301
- InitializeArrayOfBooleans method, 50
- InitializeArrayOfIntegers method, 46–49
- InitializeSieve method, 46
- Initializing factories, 461
- InMemoryGateway class, 535
- InMemoryOrderGateway class, 531, 535
- InMemoryPayrollDatabase class, 606, 669
- InMemoryProductGateway class, 532
- Insert method
 - DbOrderGatewayTest, 539
 - DbProductGateway, 532
 - DbProductGatewayTest, 537
 - InMemoryOrderGateway, 531
 - InMemoryProductGateway, 532
 - OrderGateway, 528–529
- InsertItems method, 530
- insertPaymentMethodCommand variable, 621–622
- Instability (I) metric, 427–428, 456
- Instances, 331–332
 - Monostate, 336–343
 - Singleton, 332–336
- Instantiating proxies, 442–443
- Insulation layers, 526–527
- IntBubbleSorter class, 318–319
- Integration in extreme programming, 17–18
- Integration penalty, 421
- Intentional programming, 33
- Interaction controllers, 600–601
- Interactions in agile development, 5
- Interface pollution, 163–165
- Interface Segregation Principle (ISP), 163
 - ATM user interface example, 169–174
 - class interfaces vs. object interfaces in, 166
 - for interface pollution, 163–165
 - modem problem, 500
 - in Observer, 493
 - separate clients in, 165–166
 - for separation, 167–168
- Interfaces
 - in class diagrams, 246–247, 249–250
 - for CoffeeMaker, 267–273
 - names of, 302
 - in sequence diagrams, 240–241
- Internal scope state variables, 583
- IntSortHandler class, 322–323
- InvalidOperationException class, 376
- Inverse principle. *See* Dependency-Inversion Principle (DIP)
- Inversion, ownership, 155–156
- Irresponsible components, 428, 450
- IS-A relationships, 138, 142
- “Is connected to” relationships, 245
- isBrewing method, 275
- IsCommission property, 646
- IsHourly property, 646
- IsInPayPeriod method
 - DateUtil, 406
 - HourlyClassification, 401, 403
 - PaymentClassification, 404
- IsLastDayOfMonth method, 397
- IsLines method, 254
- IsLocked method, 589
- IsMember method, 145
- Isolation in testing, 33–35
- IsOn method, 148–150
- ISP. *See* Interface Segregation Principle (ISP)





IsPayDate method
 Employee, 408
 MonthlySchedule, 397
 WeeklySchedule, 402
 IsReady method
 ContainmentVessel, 284
 HotWaterSource, 281–282
 M4ContainmentVessel, 274–275, 284
 M4HotWaterSource, 273–274, 282
 IsSalary property, 646
 IsTimeToPay method, 346–348
 IsUnlocked method, 589
 Item class, 534–535
 Item property, 511
 ItemCount property, 533–534
 ItemData class, 521–523
 Iteration planning, 15, 25–27
 Iterative process for diagrams, 194–200

J

Jacobson, Ivar, 122
 Jeffries, Ron, 18
 Johnson, Ralph, 312
 Just-in-time requirements, 220

K

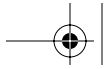
Kelly, Kevin, 41
 Kelvin, Lord, 23
 Keyboard, Copy program for, 108–113
 KeyboardReader class, 112–113
 Khrushchev, Nikita, 495
 Kitchen instance, 212
 Koss, Bob, 55

L

Lamp, 496
 Abstract Server for, 496–497
 abstraction in, 158–159
 Adapter for, 498–499
 in Dependency-Inversion Principle,
 157–158

Lamp class, 157–159
 Large systems, component design in, 416
 LastBallInFrame method, 93
 Layering in Dependency-Inversion Principle, 154–155
 Length method, 322
 Lifelines in sequence diagrams, 226
 Light class, 263–264, 496–499
 Line class, 148–150
 LinearObject class, 149
 LineSegment class, 148–150
 Links in collaboration diagrams, 183
 Liskov, Barbara, 136
 Liskov Substitution Principle (LSP),
 135–136
 factoring in, 148–150
 heuristics and conventions in, 150–151
 in modem problem, 499
 real-world example, 143–147
 violations of, 136–143
 LoadAddEmployeeView method
 MockViewLoader, 663
 ViewLoader, 658
 WindowViewLoader, 664–665
 WindowViewLoaderTest, 664
 LoadData method, 633
 LoadDirectDepositMethodCommand method,
 630
 LoadEmployee method, 623–624
 LoadEmployeeCommand method, 625
 LoadEmployeeData method, 626
 LoadEmployeeOperation class, 624–628,
 630
 LoadEmployeeOperationTest class, 624–628
 LoadEmployeeTable method, 612
 LoadHoldMethod, 629
 Loading employees in database, 623–635
 LoadingEmployeeDataCommand method,
 625
 LoadingHoldMethod method, 628
 LoadingSchedules method, 626–627
 LoadItems method, 530
 LoadMailMethodCommand method,
 631–634





LoadPaymentClassificationOperation class,
634

LoadPaymentMethod class, 628

LoadPaymentMethodOperation class,
628–635

LoadPaymentMethodOperationTest class,
628–634

LoadPayrollView method
MockViewLoader, 663
ViewLoader, 658
WindowViewLoader, 664, 669
WindowViewLoaderTest, 663

LoadPrimes method, 45–46

LoadTable method, 616–617

LoadView method, 665

Lock method
Turnstile, 341, 589
TurnstileFSM, 592

Locked class, 341–343, 596

Locked state
in state diagrams, 184, 208–209
in Turnstile, 338

LockedTurnstileState class, 587–588

Logger interface, 240–241

Login events, 204

Login method in sequence diagrams,
226–227

login.sm file, 599–600

Login state machines, 204–205

LoginPage class, 189–190

LoginTransaction class, 257

LogMessage method, 235, 238, 240

LogOneMessage method, 238

LogoutExitModem decorator, 564

LogQueuedMessages method, 238

Loops in sequence diagrams, 228–229,
232–233

LoudDialModem class, 561–565

Low-level modules in Dependency-
Inversion Principle, 154

LSP. *See* Liskov Substitution Principle
(LSP)

lunchRoom instance, object diagram for, 212

M

M4CoffeeMaker class, 276–277, 280–283,
285–286

M4ContainmentVessel class, 273–275,
284–285

M4HotWaterSource class, 274

M4UserInterface class, 272–275, 277,
280–281

MailMethod class, 357, 613, 616, 631

MailMethodGetsSaved method, 617

Main loop structure in Application, 312–313

Main method in M4CoffeeMaker, 276–277,
286

Main program for payroll, 408–409

Main sequence
in abstraction, 432–434
distance from, 434–435, 457

MainLoggerLoop method, 237–238

Maintainability vs. reusability, 419

Make method, 440–441

MakeCircle method, 438–439

MakeClassification method
AddEmployeeTransaction, 369
AddSalariedEmployee, 371

MakeSchedule method
AddEmployeeTransaction, 369–370
AddSalariedEmployee, 371

MakeSquare method
ShapeFactory, 227, 438
ShapeFactoryImplementation, 439

Managers in extreme programming, 14

Manifesto of the Agile Alliance, 4–5

ManyMessages method, 236

Mark IV Special Coffee Maker. *See* Coffee-
Maker class

Martin, Bob, 55

Martin’s First Law of Documentation, 6

Math class, 250

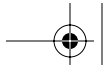
Matrices of functions, 548, 552

maxPrimeFactor method, 50

McBreen, Pete, 13

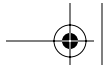
Measures of progress, 9





- Mechanism layer, 155
- MechanismLayer layer, 155–156
- Mediator pattern, 327–329
- Messages in sequence diagrams, 226, 233–241
- MessagesInQueue method, 238
- MessagesLogged method, 238
- Metaphors in extreme programming, 21–22
- Method property
 - ChangeMethodTransaction, 388
 - Employee, 408
 - LoadPaymentMethodOperation, 633
- Methods class
 - in Common Closure Principle, 450
 - merging into PayrollImplementation, 462
 - metrics for, 457, 459
 - in Reuse/Release Equivalence Principle, 452–453
- Methods in class diagrams, 244
- MethodTransactions class
 - metrics for, 459
 - in Reuse/Release Equivalence Principle, 453
- Metrics
 - for abstraction, 432, 456
 - in package analysis, 455–457
 - for payroll application, 457–463
 - for stability, 427–428
- Meyer, Bertrand, 122, 142–143
- Middleware
 - with Singleton, 334
 - third-party, 526–528
- MockAddEmployeeView class, 642, 649–650
- MockGame class, 61
- Mocking, 443
- Mockobject pattern, 34
- MockPayrollView class, 662
- MockTimeSink class, 475–477, 479–480, 490
- MockTimeSource class, 475–476, 478, 482–487, 489
- MockViewLoader class, 658, 662–663
- Model View Presenter pattern, 637
 - for employee transactions, 641–642
 - for windows
 - building, 650–657
 - payroll, 657–669
- Models
 - for Observer pattern, 492–493
 - purpose of, 187–188
- Modem interface, 117, 499–500, 503–504, 543–545, 549, 561–563
- Modem problem
 - Acyclic Visitor for, 548–552
 - Adapter for, 499–503
 - bridges in, 503–505
 - Decorator for, 560–565
 - Visitor for, 544–548
- Modem property, 565
- ModemConnectionController class, 504
- ModemDecorator class, 564–565
- ModemDecoratorTest class, 563–564
- ModemImplementation class, 119, 504
- ModemVisitor interface, 548–549
- ModemVisitorTest class, 547, 551–552
- Modifications in Open/Closed Principle, 122
- Modules in Dependency-Inversion Principle, 154
- Monostate class, 337
- MONOSTATE pattern, 331–332, 336–337
 - benefits and costs of, 337–338
 - for DeleteEmployeeTransaction, 373
 - example, 338–343
- MonthlySchedule class, 397
- Morning-after syndrome, 420
- Motivated individuals, 9
- Motor class, 158
- MotorOffCommand, 300
- Multiple inheritance, 168
- Multiple threads in sequence diagrams, 239–240
- Multiplicity
 - in class diagrams, 254–255
 - in Composite, 470



**N**

Name-only dependencies, 440
Name property
 AddEmployeePresenter, 645
 Employee, 407
 Product, 533
 ProductImpl, 517
 ProductProxy, 517
Name/value pairs for properties, 251
Names of interfaces, 302
nameTextBox_TextChanged method, 654
Natural structure in Open/Closed Principle, 128–129
Negotiation vs. collaboration, 6–7
Nested classes in class diagrams, 256
Nested switch/case statements, 580–584
NewElement method, 572
NewOrder method, 520
NewTextElement method, 572
NextBall method, 84
NextBallForSpare method, 87, 89, 95
NextTwoBalls method, 83
NextTwoBallsForStrike method, 87, 89, 95
NoAffiliation class, 393
 Null Object for, 357
 for payment methods, 362
 for service charges, 379–380
Nodes on directed graphs, 421
NoItems method, 523
Nonblocking source control, 17–18
NormalBrew method, 291
NormalFill method, 290
NormalStart method, 289
Norman, Donald A., 447
Nosek, J. T., 16
Notation for components, 448–450
NotCrossed method, 47, 49, 52
Notifications in Reuse/Release Equivalence Principle, 417
Notify method
 SalaryObserver, 492
 TimeSource, 483

TimeSourceImplementation, 485

NotifyObservers method, 489

Null Object pattern
 description, 345–348

 for service charges, 380

NullEmployee class, 346–348

NumberOfFridaysInPayPeriod method, 404

NumberOfUncrossedIntegers method,
 49, 51

O

Object diagrams, 182, 211
 active objects in, 213–217
 purpose of, 211–212
Object-form adapters, 498
Object interfaces in Interface Segregation Principle, 166
Object-oriented database management systems (OODBMS), 369, 410
Object-oriented design
 for CoffeeMaker, 279–291
 limitations of, 98
Objects in sequence diagrams, 226–227
ObservableClock class, 483–484, 486
Observer interface, 488
Observer pattern
 evolving, 471–491
 models for, 492–493
 for OOD principles, 493
ObserverTest class, 488
OCP. *See* Open/Closed Principle (OCP)
Off method, 264
On method, 264
One-to-many association, 245
OneMessage method, 236
OODBMS (object-oriented database management systems), 369, 410
Opacity in design, 107
Open/Closed Principle (OCP)
 abstraction in, 123–124, 128–131, 430





Open/Closed Principle (OCP) (*Continued*)
 for adding employees, 366
 anticipation and natural structure in,
 128–129
 for components, 419
 conforming to, 127–128
 in Copy program, 112
 data-driven approach to, 131–132
 description, 122–124
 in modem problem, 499
 in Observer, 493
 in payroll system, 360–361
 for stability, 430
 violations of, 124–127
 Open workspace in extreme programming,
 18
 OpenConnection method, 536
 Order class, 508, 510, 533–534
 Order interface, 519–520
 OrderData class, 519–520
 OrderGateway interface, 528–529
 OrderId property, 522
 OrderImp class, 521
 OrderKeyGeneration method, 520
 OrderProxy class, 519, 522
 OutOfOrder method
 BubbleSorter, 317
 DoubleBubbleSorter, 319
 IntBubbleSorter, 318
 IntSortHandler, 323
 Ownership
 in extreme programming, 17
 inversion of, 155–156

P

Pace in extreme programming, 18
 Packaging, 415–416, 447–448
 Common Closure Principle in, 450–451
 components in. *See* Components
 coupling and encapsulation in, 454–455
 final structure, 463–466
 metrics in, 455–463

Reuse/Release Equivalence Principle
 in, 452–454
 Page class, 189–190
 Page-Jones, Meilir, 116
 Pain, zone of, 433
 Pair programming
 Bowling game, 56–98
 in extreme programming, 16
 Part class, 553, 570
 Part interface, 553
 PartCount method, 559
 PartExtension interface, 571
 Partitionings, errors in, 263
 PartNumber property
 Assembly, 554, 572
 PiecePart, 555, 571
 PartNumberCount property, 556
 Parts property, 554, 572
 PartVisitor interface, 555
 Pass method
 Locked, 596
 State, 595
 Turnstile, 341–342, 588–589, 594
 Unlocked, 595
 UnlockedTurnstileState, 588
 PassInLockedState method
 SMCTurnstileTest, 598
 TurnstileTest, 582–583
 PassInUnlockedState method
 SMCTurnstileTest, 598
 TurnstileTest, 582–583
 Patterns
 with databases, 539–540
 evolving, 471–491
 Pause method
 M4HotWaterSource, 282
 TestLog, 237
 Pause transitions in state transition dia-
 grams, 206
 pay method, 346
 PaycheckAddress table, 605, 613, 616
 PayClassification class, 302, 304
 Payday method, 395, 398, 408

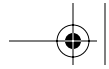




Index

- PaydayTransaction class, 393–397, 403–404
- payEmployee message in sequence diagrams, 232
- PayGasBillTransaction class, 169
- PayingSingleHourlyEmployeeNoTimeCards method, 398
- PaymentClassification class
 - for abstraction, 360, 384, 386
 - for adding employees, 369
 - benefits of, 356
 - in Common Closure Principle, 450
 - for load operations, 634
 - for pay periods, 403–404
 - for payments, 358–359, 393
 - in Reuse/Release Equivalence Principle, 452
 - for time cards, 373–375
- PaymentClassification table, 605
- PaymentMethod class, 357, 393
 - abstraction in, 362, 388
 - in Common Closure Principle, 450
 - in Reuse/Release Equivalence Principle, 452
- PaymentMethod table, 605
- PaymentMethodCode method, 613
- PaymentMethodGetsSaved method, 613
- Payments
 - abstraction in, 360
 - methods, 362
 - process, 358–359, 393–396
 - hourly employees, 398–402
 - pay periods for, 402–408
 - salaried employees, 396–398
 - schedules for, 360–361, 366–367
- PaymentSchedule class, 369, 386, 393
 - in Common Closure Principle, 450
 - for pay periods, 403
 - in Reuse/Release Equivalence Principle, 452
- PaymentSchedule table, 605
- Payroll class, 33–37, 230
 - abstractions in, 360–363
 - testing, 443–444
- Payroll system, 349–350, 365
 - affiliations in, 362–363
 - classification changes in, 384–393
 - Command pattern for, 302–304
 - database for. *See* PayrollDatabase class
 - employees in
 - adding, 302–304, 352–353, 366–371
 - changing, 356–358, 381–393
 - deleting, 353, 372–373
 - paying, 393–408
 - Factory for, 442–444
 - main program, 408–409
 - metrics for, 457–463
 - Null Object for, 345–348, 380
 - packaging. *See* Components; Packaging
 - payments in. *See* Payments
 - sales receipts in, 354–355, 377
 - service charges in, 355, 378–380
 - specification for, 350–351
 - time cards in, 354, 373–377
 - transactions in, 302–304, 366
 - user interface for. *See* User interfaces
- Payroll window, 657–669
- PayrollApplication class, 408–409, 448–450
 - class allocation in, 458, 464
 - in Common Closure Principle, 450
 - metrics for, 457, 459, 465
 - in Reuse/Release Equivalence Principle, 452, 454
- PayrollDatabase class, 603–604
 - for affiliations, 391–393
 - building, 604–605
 - class allocation in, 458, 464
 - for DeleteEmployeeTransaction, 373
 - design flaw in, 605–607
 - employees in
 - adding, 368–369, 607–617
 - changing, 382
 - loading, 623–635
 - metrics for, 457, 459, 465





- PayrollDatabase class (*Continued*)
 - objects in, 409–410
 - in Reuse/Release Equivalence Principle, 452
 - for service charges, 379
 - transactions for, 618–623
- PayrollDatabase interface, 607
- PayrollDatabaseImplementation class
 - class allocation in, 458, 464
 - metrics for, 459, 465
- PayrollDomain class
 - class allocation in, 458, 464
 - in Common Closure Principle, 450
 - metrics for, 457, 459, 465
 - in Reuse/Release Equivalence Principle, 452–454
- PayrollFactory class
 - class allocation in, 458, 464
 - metrics for, 465
- PayrollImplementation class
 - class allocation in, 458, 464
 - for cohesion, 462
 - metrics for, 465
- PayrollMain class, 669
- PayrollPresenter class, 657, 660–661
- PayrollPresenterTest class, 658–660
- PayrollTest class, 34, 443–444, 606
- PayrollView interface, 661–662
- PayrollWindow class, 666–667
- PayrollWindowTest class, 665–666
- PaySingleHourlyEmployeeOneTimeCard method, 399
- PaySingleHourlyEmployeeOnWrongDate method, 400
- PaySingleHourlyEmployeeOvertimeOneTimeCard method, 399
- PaySingleHourlyEmployeeTwoTimeCards method, 400
- PaySingleSalariedEmployee method, 396
- PaySingleSalariedEmployeeOnWrongDate method, 396–397
- PDImplementation class, 452
- People in agile development, 5
- PerformClick method, 656
- Persistence in Single-Responsibility Principle, 119
- PersistentObject class, 145–147
- PersistentSet set, 145–147
- Phone calls, timing of, 233–234
- Phone class, associations with, 245
- PhoneNumber property
 - HayesModem, 563
 - LoudDialModem, 563
 - ModemDecorator, 565
- Phones, cellular
 - code for, 198–199
 - collaboration diagrams for, 194–196
 - diagram evolution for, 199–200
 - state diagrams for, 194
- Photocopier commands, 300–302
- Physical bonds, 497
- Physical decoupling, 304
- Physical diagrams, 179
- PieceCount property, 553, 556
- PiecePart class, 554–555, 571
- PiecePart1toCSV method, 569
- PiecePart1XML method, 568
- PiecePart2toCSV method, 569
- PiecePart2XML method, 568
- Planning, 23–24
 - in extreme programming, 19
 - flexibility of, 7–8
 - initial exploration in, 24–25
 - iteration, 15, 25–27
 - release, 25
 - task, 26–27
 - tracking, 28–29
- Platforms in Monostate, 338
- Plus signs (+) in class diagrams, 244
- Poe, Edgar Allen, 543
- Point-of-sale systems, 220–221
- Point structure, 136–137
- Policy class, 124
- Policy layer, 155
- PolicyLayer layer, 155–156
- PolicyServiceInterface interface, 156



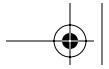


Index

721

- Political issues, 417
- Poll method, 157–158
 - M4ContainmentVessel, 284–285
 - M4HotWaterSource, 282
 - M4UserInterface, 277, 280
 - Pollable, 275–276, 286
 - TestCoffeeMaker, 288
- Pollable interface, 275–276, 285–286
- Polyadic form of methods, 173–174
- Polymorphism
 - in Monostate, 337
 - in Open/Closed Principle, 135
 - in Shape, 137
- Positional stability of components, 427
- Postconditions, 143
- Posting in payroll system
 - payments, 395
 - sales receipts, 354–355
 - service charges, 355
 - time cards, 354
- PotRemovedAndReplacedWhileEmpty method, 289–290
- PotRemovedWhileNotEmptyAndReplaced-Empty method, 290
- PotRemovedWhileNotEmptyAndReplaced-NotEmpty method, 290
- Powell, Colin, 603
- Preconditions, 143
- Prepare method, 632–634
- PrepareToSaveEmployee method, 620
- PrepareToSavePaymentMethod method, 616–617, 620–621
- Presence of MONOSTATE, 338
- Presenter property
 - AddEmployeeWindow, 654
 - MockPayrollView, 662
 - PayrollWindow, 667
- PresenterValuesAreSet method, 651, 656
- Price property
 - Product, 511, 533
 - ProductImpl, 517
 - ProductProxy, 517
- Primary course in use cases, 220
- Prime numbers. *See* GeneratePrimes program
- PrimeGenerator class, 45–52
- Principles in agile development, 8–10
- Printers, Copy program for, 108–113
- PrinterWriter class, 113
- PrintSet method, 145
- PriorityFor method, 132
- Private classes in class diagrams, 244
- Processes in agile development, 5
- Product class, 510–512, 533, 535
- Product interface, 516
- ProductData class, 326, 513–514
- ProductDataExtractProductDataFromReader method, 515
- ProductDBProxy class, 511
- ProductGateway interface, 531
- ProductImpl class, 517–518
- ProductImplementation class, 511–512
- ProductProxy class, 517–518
- ProductProxy method, 516
- Programming
 - by coincidence, 70
 - by difference, 312
 - extreme. *See* Extreme programming
- Programming practices example, 56–98
- Progress measures, 9
- PromptForAccount method, 247
- Properties
 - in class diagrams, 251–252
 - virtual, 140
- Protected classes in class diagrams, 244
- PROXY pattern
 - for association stereotypes, 256
 - factories with, 442–443, 445
 - implementing, 512–525
 - for refactoring, 119
 - for shopping cart, 508–512
 - for third-party APIs, 527–528
- ProxyTest class, 516, 519
- Pseudostates in state transition diagrams, 204, 207–208
- Public classes in class diagrams, 244





Pull model observers, 487, 492
 Push model observers, 487
 PutUncrossedIntegersIntoResult method,
 49, 51

Q

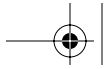
Quantity class, 511
 Quantity property, 535
 QuantityOf property, 534
 QuickBubbleSorter class, 323–324
 QuickEntryMediator class, 327–329

R

Race conditions in sequence diagrams, 234
 Raskin, Jef, 637
 Rationales, 6
 Ray class, 150
 RDBMS (relational database management
 systems), 369, 410
 Read Keyboard module, 108
 Read method, 112
 Readability of software, 42, 49
 Reader interface, 112
 ReadUser method, 335
 “Realizes” relationships, 246–247
 receive method, 504
 receiveImp class, 504
 RecordMembership method
 Affiliation, 393
 ChangeAffiliationTransaction, 391
 ChangeMemberTransaction, 392
 ChangeUnaffiliatedTransaction, 392
 Rectangle class, 116–117, 138–141
 Rectangles, drawing, 600
 rcv method, 118
 Redesign problems, 104
 Reeves, Jack, 103
 Refactoring, 41–42
 example. *See* GeneratePrimes program
 in extreme programming, 17, 20–21
 Reflexive transitions in state transition
 diagrams, 205

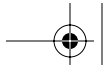
Refund method, 342
 Refunds property, 341
 Register method, 164–166
 RegisterObserver method, 481
 MockTimeSource, 482, 484, 486
 Subject, 489
 TimeSource, 482–483, 485
 TimeSourceImplementation, 485
 Regulate method, 160–161
 Relational cohesion (H) metric, 456
 Relational database management systems
 (RDBMS), 369, 410
 Relationships
 in class diagrams, 181
 collaboration diagrams for, 183–184
 RelayOnCommand method, 300
 Release planning, 15, 25
 RemoteTransactionSource class, 409
 REP (Reuse/Release Equivalence Principle)
 applying, 452–454
 description, 417–418
 Repetition in design, 106–107
 Report generation
 Extension Object for, 565–576
 Visitor for, 552–559
 RequestDepositAmount method, 169, 171
 RequestTransferAmount method, 169, 172
 RequestWithdrawalAmount method, 172
 Requirements change
 attitudes toward, 9
 rotting software from, 107
 Requirements documents, acceptance tests
 as, 36
 Requirements in extreme programming, 14
 reset method, 341
 ReSharper refactoring add-in, 45
 Responsibility. *See* Single-Responsibility
 Principle (SRP)
 Responsible components, 427–428, 450–451
 Resume method, 283
 Reuse
 Common Reuse Principle for, 418–419
 Reuse/Release Equivalence Principle
 for, 417–418





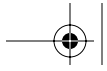
- Reuse/Release Equivalence Principle (REP)
 - applying, 452–454
 - description, 417–418
 - Rigidity in design, 105
 - Road maps, diagrams as, 191–192
 - Rotting software, 104–107
 - RTC (run-to-completion tasks), 308
 - Rules of bowling, 99–100
 - Run method
 - ActiveObjectEngine, 305–306
 - Application, 314
 - ApplicationRunner, 320
 - SocketServer, 215
 - Run-to-completion tasks (RTC), 308
 - runButton_Click method, 667
 - RunTransactions method
 - PayrollPresenter, 661
 - PayrollPresenterTest, 659–660
 - PayrollWindowTest, 666
- S**
- Salaries employee payments, 396–397
 - SalariesClassification class, 356
 - SalariesClassification table, 605
 - SalariesEmployee class, 246
 - SalariesUnionMemberDues method, 402–403
 - Salary property, 646
 - SalaryObserver class, 492
 - salaryRadioButton_CheckedChanged method, 654
 - salaryTextBox control, 656
 - salaryTextBox_TextChanged method, 655
 - Sales receipts
 - implementing, 377
 - posting, 354–355
 - SalesReceipt class
 - in Common Closure Principle, 450
 - couplings in, 454–455
 - posting in, 354–356
 - SalesReceipt table, 605
 - SalesReceiptTransaction class
 - couplings in, 454–455
 - static model of, 377
 - Salient documentation, 6
 - SAP (Stable Abstractions Principle), 431–435
 - SaveEmployeeOperation class, 622–624
 - SaveEnabled property, 649
 - SaveIsTransactional method, 618
 - SaveMailMethodThenHoldMethod method, 621–622
 - SavePaymentMethod method, 615
 - Scenarios, sequence diagrams for, 228–231
 - Schedule property
 - ChangeHourlyTransaction, 387–388
 - Employee, 407–408
 - ScheduleCode method, 611–612
 - ScheduleGetsSaved method, 610, 612
 - Schedules class
 - in Common Closure Principle, 450
 - merging into PayrollImplementation, 462
 - metrics for, 457, 459
 - in Reuse/Release Equivalence Principle, 452
 - Schedules for payments, 360–361, 366–367
 - Score method, 73
 - Score property, 59–62, 64, 68, 74, 76, 87, 91–93
 - ScoreForFrame method, 64–65, 67–68, 72–73, 75, 77–89, 94
 - Scorer class, 65, 80, 87–89, 94–95, 116
 - Screen class, 194
 - SDP (Stable-Dependencies Principle), 426–431
 - SelectSubNode method, 180
 - Self-organizing teams, 10
 - Self-Shunt pattern, 443
 - send method
 - DedModemController, 504
 - Modem, 118
 - sendImp class, 504
 - Sending Password Failed state in login state machine, 205





- Sending Password Succeeded state in login state machine, 205
- sendPassword method, 205
- Seneca, 31
- Sense method, 265
- Sensor class, 263
 - abstraction in, 265
 - for composites, 469–470
 - function of, 301
- Separate clients in Interface Segregation Principle, 165–166
- Separation
 - of coupled responsibilities, 119
 - through delegation, 167–168
 - through multiple inheritance, 168
- Sequence diagrams, 182–183, 225–226
 - active objects in, 240
 - for cases and scenarios, 228–231
 - conditions in, 232–233
 - creating and destroying, 227–228
 - loops in, 228–229, 232–233
 - messages in, 226, 233–239
 - multiple threads in, 239–240
 - objects in, 226–227
 - sending messages to interfaces in, 240–241
- Sequence numbers in collaboration diagrams, 184
- Serendipitous architecture, 37–38
- Serendipitous decoupling, 36
- Serve method, 214
- Server class, 123
- Server method, 214
- Service charges
 - implementing, 379–380
 - pay periods for, 402–403
 - posting, 355
- ServiceCharge class, 379–380, 449
- ServiceChargesSpanningMultiplePayPeriods method, 405–406
- ServiceChargeTransaction class, 378–380, 449
- ServiceRunner method, 215
- Set interface, 145
- SetArray method, 322
- SetBoilerState method
 - CoffeeMakerAPI, 262
 - CoffeeMakerStub, 287
- SetDone method, 314
- SetDriver method
 - ClockDriver, 474
 - MockTimeSource, 476
 - TimeSource, 475
- SetIndicatorState method
 - CoffeeMakerAPI, 262, 264
 - CoffeeMakerStub, 287–288
- SetLocked method, 589
- SetObserver method, 481
 - MockTimeSource, 478–479
 - TimeSource, 478
- SetReliefValveState method
 - CoffeeMakerAPI, 263
 - CoffeeMakerStub, 288
- SetState method, 594
- SetTime method
 - MockTimeSink, 477
 - MockTimeSource, 476, 478, 482–483, 485, 489
 - ObservableClock, 484
 - TimeSink, 475
- SetUnlocked method, 589
- SetUp method
 - AddEmployeePresenterTest, 642, 648
 - AddEmployeeWindowTest, 650–651, 656
 - Blah, 607, 609
 - BOMReportTest, 556–557
 - BomXmlTest, 567
 - ClockDriverTest, 481
 - DbOrderGatewayTest, 538
 - DbProductGatewayTest, 537
 - DBTest, 513
 - GameTest, 66, 95
 - LoadEmployeeOperationTest, 625
 - LoadPaymentMethodOperationTest, 629





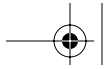
- ModemVisitorTest, 547, 551
- ObserverTest, 488
- PayrollPresenterTest, 659
- PayrollWindowTest, 665
- ProxyTest, 516
- SMCTurnstileTest, 597
- SqlPayrollDatabaseTest, 609, 611–612
- TestCoffeeMaker, 288
- TestLog, 236
- TransactionContainerTest, 668
- TurnstileTest, 339, 582
- WindowViewLoaderTest, 663
- SetUpReportDatabase method, 558–559
- SetWarmerState method
 - CoffeeMakerAPI, 262
 - CoffeeMakerStub, 287
- Shape application, 124–126
- Shape class, 136–137
- Shape interface, 130, 438, 467–468
- Shape structure, 125–126
- ShapeComparer class, 131–132
- ShapeFactory class, 227–228
- ShapeFactory interface, 438–441
- ShapeFactoryImplementation class, 438–441
- ShapeType enumeration, 125–126, 136
- Shopping cart
 - implementing, 512–525
 - object model, 508
 - Proxy for, 508–512
 - relational data model, 509
- Short cycles in extreme programming, 15
- Show method, 656
- showLoginScreen method, 204–205
- ShuntRow method, 627, 630
- Sieve method, 46
- SillyAddAction method, 668
- SimpleAssemblyCSV method, 570
- SimpleAssemblyXML method, 568–569
- Simplicity
 - in extreme programming, 19–20
 - importance of, 10
 - for use cases, 219–220
- Single-Responsibility Principle (SRP), 115–117
 - for adding employees, 366
 - for CoffeeMaker, 273
 - for components, 419
 - defining responsibilities in, 117–118
 - in dependency cycles, 425
 - in payroll system, 361
 - persistence in, 119
 - for report generation, 553
 - separating coupled responsibilities in, 119
 - in table lamp, 498
- Singleton class, 332–333
- SINGLETON pattern, 331–333
 - benefits and costs of, 334
 - for DeleteEmployeeTransaction, 373
 - example, 334–336
- Sku property
 - Product, 533
 - ProductImpl, 517
 - ProductProxy, 517–518
- SleepCommand class, 306–308, 310
- SleepTillMoreMessagesQueued method, 238
- SMC (State Machine Compiler), 591–593
- SMCTurnstileTest class, 597–598
- Smells of rotting software, 104–107
- SocketServer class, 213–217
- SocketService interface, 214
- Software
 - early and continuous delivery of, 8
 - models for, 187–188
 - rotting, 104–107
- Software delivery in extreme programming, 15
- Software module functions, 42
- Sort method
 - BubbleSorter, 190, 316–317, 322
 - DoubleBubbleSorter, 318
 - IntBubbleSorter, 318
 - QuickBubbleSorter, 323
- SortHandler interface, 322





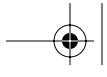
- Sorts
 - bubble sorts, 316–319, 321–322
 - QuickBubbleSorter, 323–324
- Source code dependencies, class diagrams
 - for, 243–244
- Source control, nonblocking, 17–18
- Spare method, 84, 89, 95
- Spares
 - in bowling, 99
 - testing, 66–73
- Sparse matrices, 552
- Speaker class, 194
- SpeakerVolume property
 - HayesModem, 562
 - LoudDialModem, 563
 - ModemDecorator, 565
- Special events in state transition diagrams, 205
- Specification-level diagrams, 178–179
- Splitting user stories, 24–25
- Spock, Mister, 507
- Spoofing, 443
- SQL Server, 604
- SqlCommandBuildInsertionCommand
 - method, 514
- SqlCommandBuildItemInserionStatement
 - method, 523–524
- SqlCommandBuildItemsForOrderQueryS-
 - tatement method, 524
- SqlCommandBuildProductDeleteState-
 - ment method, 515
- SqlPayrollDatabase class, 607–608, 614–623
- SqlPayrollDatabaseTest class, 607, 609–614, 621–624
- SqlTransaction class, 618
- Square class, 137–141, 438, 467–468
- Square structure, 125–127
- SRP. *See* Single-Responsibility Principle (SRP)
- Stability
 - and abstraction, 432
 - definition, 426–427
 - metrics for, 427–428, 455–457
 - principles, 420
 - Acyclic Dependencies Principle, 420–426
 - Stable Abstractions Principle, 431–435
 - Stable-Dependencies Principle, 426–431
 - variable, 429
- Stable Abstractions Principle (SAP), 431–435
- Stable-Dependencies Principle (SDP), 426–431
- Start method
 - ContainmentVessel, 283
 - HotWaterSource, 274, 281
 - M4ContainmentVessel, 275
- StartBrewing method, 272–273
 - M4HotWaterSource, 282
 - UserInterface, 280
- StartedPotNotEmpty method, 289
- StartingState method, 651, 656
- StartNoPot method, 288–289
- StartNoWater method, 289
- StartServiceThread method, 214
- State class, 595
- State diagrams, 184, 203
 - basics, 204–205
 - for cellular phones, 194
 - FSM, 208–209
 - state transition diagrams, 204–205, 600–601
 - pseudostates in, 207–208
 - special events in, 205
 - superstates in, 206–207
- State Machine Compiler (SMC), 591–593
- State machines
 - applications
 - distributed processing, 601–602
 - GUI interaction controllers, 600–601
 - high-level application policies for GUIs, 598–600
 - UML notation for, 184





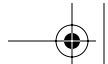
- STATE pattern
 - costs and benefits of, 590
 - vs. Strategy, 589–590
 - for turnstile, 586–589
 - State transition tables (STTs), 208
 - State variables, internal scope, 583
 - Stateless interfaces, 598
 - StateName method, 595
 - States, 579–580
 - in state diagrams, 184
 - State Machine Compiler for, 591–593
 - transition tables for, 584–586
 - Static diagrams, 179
 - Static typing vs. dynamic, 441–442
 - Statistical analysis of designs, 434–435
 - Stereotypes in class diagrams, 249–250, 255–256
 - Stop method, 238
 - StopCommand class, 309
 - Store method
 - Db, 514
 - ItemData, 523
 - StoreItem method, 523
 - StoreProduct method, 513
 - STRATEGY pattern, 113, 312
 - for Application problem, 319–324
 - for employee pay, 398
 - in Open/Closed Principle, 123
 - in payroll system, 361
 - vs. STATE, 589–590
 - vs. TEMPLATE METHOD, 323–324
 - Strike method, 83, 92–93, 95
 - Strikes in bowling, 99
 - Strong players, 5
 - Stroustrup, Bjarne, 55
 - Structure
 - class diagrams for, 196–198
 - of components, 448–450
 - Structure documents, 6
 - STTs (state transition tables), 208
 - Subject class, 487, 489
 - SubmitEnabled property
 - AddEmployeeWindow, 656–657
 - MockAddEmployeeView, 649
 - Substates in state transition diagrams, 206
 - Substitutable factories, 442–443
 - Substitution and subtypes. *See* Liskov Substitution Principle (LSP)
 - Subway turnstiles. *See* Turnstiles
 - Superstates in state transition diagrams, 205–207
 - Support efforts, 417
 - Sustainable pace
 - in agile development, 9
 - in extreme programming, 18
 - Swap method
 - BubbleSorter, 190, 316
 - DoubleBubbleSorter, 319
 - IntBubbleSorter, 318
 - IntSortHandler, 322
 - Switch/case statements, 580–584
 - Switch class, 496–497
 - Switchable interface, 496–499
 - SwitchableDevice, 159
 - Synchronous messages in sequence diagrams, 235
 - System boundary diagrams in use cases, 222
 - System.Data namespace, 326
- T**
- Table Data Gateway (TDG) pattern
 - example, 528–535
 - with Facade, 326
 - testing, 535–539
 - Table lamp, 496
 - Abstract Server for, 496–497
 - abstraction in, 158–159
 - Adapter for, 498–499
 - in Dependency-Inversion Principle, 157–158
 - Talfourd, Thomas Noon, 153
 - Tasks
 - in extreme programming, 15
 - planning, 26–27
 - TDD (test-driven development), 16, 32





- Teams, 5
 - in extreme programming, 14
 - self-organizing, 10
- TearDown method
 - Blah, 609
 - DbOrderGatewayTest, 538
 - DbProductGatewayTest, 537
 - DBTest, 513
 - PayrollWindowTest, 665
 - ProxyTest, 516
 - TestLog, 236
- Technical excellence, attention to, 10
- TEMPLATE METHOD pattern
 - abuse of, 315
 - for adding employees, 369
 - for Application problem, 311–325
 - for bubble sort, 316–319
 - for changing employees, 382–384, 386–387, 389, 391
 - for modem problem, 561
 - in Open/Closed Principle, 124
 - for similar functions, 20
 - vs. STRATEGY, 323–324
- Temporal decoupling, 304
- Tennyson, Alfred, 345
- Test cases in extreme programming, 18
- Test-driven development (TDD), 16, 32
- Test-first design, 32–33
- Testability of models, 188
- Testable software, 32
- TestAddOneThrow method, 59
- TestAddSalariedEmployee method, 367–368
- TestCancelAlarm method, 340
- TestChangeHourlyTransaction method, 386
- TestChangeNameTransaction method, 383
- TestCoffeeMaker class, 288–291
- TestCoin method, 339
- TestCoinAndPass method, 339
- TestCreateCircle method, 440
- TestCreateSingleton method, 333
- TestEndOfArray method, 77, 97
- TestExhaustive method, 50, 53
- TestFourThrowsNoMark method, 63, 66–67, 69, 71, 96
- TestGame class, 63
- TestHeartBreak method, 79, 97
- Testing, 31
 - acceptance tests, 15–16, 36–37
 - fixtures, 443–444
 - GeneratePrimes program, 44, 52–53
 - isolation in, 33–35
 - in Open/Closed Principle, 129–130
 - serendipitous architecture in, 37–38
 - serendipitous decoupling in, 36
 - test-driven development, 32
 - test-first design, 32–33
- TestingVisitor method, 557
- TestInit method, 339
- TestInstance method, 336
- TestInstancesBehaveAsOne method, 336
- TestLog class, 235–236
- TestMonostate method, 336
- testMove method, 32–33
- TestMultipleSinks method
 - ClockDriverTest, 481
 - ObserverTest, 488
- TestNoPublicConstructors method, 333
- TestNull method, 347
- TestOneThrow method, 61–62, 66, 69, 73
- TestOrderPrice method, 510
- TestPass method, 340
- TestPayroll method, 35
- TestPaySingleHourlyEmployeeWithTimeCardsSpanningTwoPayPeriods method, 401
- TestPerfectGame method, 76–77, 96–97
- TestPrimes method, 44, 52
- TestSampleGame method, 78, 97
- TestScoreNoThrows method, 58
- TestSimpleFrameAfterSpare method, 68, 72, 96
- TestSimpleSingleton class, 332–333
- TestSimpleSpare method, 66–67, 72, 96
- TestSimpleStrike method, 74, 96
- TestSleep method, 307

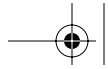


**Index**

729

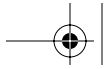
- TestSleepCommand class, 306–307
- TestTenthFrameSpare method, 79, 97–98
- TestTimeCardTransaction method, 375–376
- TestTimeChange method
 - ClockDriverTest, 475, 479, 481
 - ObserverTest, 488
- TestTurnstile class, 583
- TestTwoCoins method, 339–340
- TestTwoOperations method, 340
- TestTwoThrowsNoMark method, 63, 66, 69, 71, 95
- Text in FSM diagrams, 208
- Text interfaces, 638
- TextFieldChanged method, 328
- TextParser class, 450
- TextParserTransactionSource class, 409, 448
 - class allocation in, 458, 464
 - in Common Closure Principle, 450
 - couplings with, 460–461
 - metrics for, 459, 465
- Thankyou method, 184
 - Turnstile, 589
 - TurnstileFSM, 592
- Thermometer interface, 160–161
- Thermostat algorithm, 160–161
- Third-party APIs, 526–528
- Thomas, Dave, 70
- Thread class, 216–217
- Threads
 - in object diagrams, 213–217
 - in sequence diagrams, 239–240
- ThreadStart method, 215
- Throw class, 57–59
- ThrowTest class, 57–58
- tic method, 484
- Time, digital clock project for, 472–491
- Time cards
 - implementing, 354, 373–377
 - posting, 354
- TimeCard class, 356, 373–376, 449
 - in Common Closure Principle, 450
 - couplings in, 454–455
 - implementing, 373–376
 - for postings, 354
- TimeCard table, 605
- TimeCardTransaction class, 375–377, 449
 - couplings in, 454–455
 - structure of, 374
- TimedDoor class, 164–168
- TimeOut method, 164–168
- Timer class, 164–167
- TimerClient class, 164–168
- TimeSink interface, 473–475, 479–481, 487
- TimeSource interface, 473–475, 477–479, 482–487, 489
- TimeSourceImplementation class, 482–486
- Timing in sequence diagrams, 233–234
- Tools in agile development, 5
- Top-down design vs. bottom-up, 424–426
- Total property
 - Order, 510, 534
 - OrderImp, 521
 - OrderProxy, 522
- Tracking planning, 28–29
- Transaction class, 448, 606
 - in ATM system, 169
 - in Common Closure Principle, 450
 - in Reuse/Release Equivalence Principle, 453–454
- Transaction interface, 171, 366
- TransactionAdded method, 660
- TransactionApplication class
 - class allocation in, 458, 464
 - metrics for, 459, 465
 - in Reuse/Release Equivalence Principle, 454
- TransactionContainer class, 668–669
- transactionContainer field, 657
- TransactionContainer property
 - AddEmployeePresenter, 647
 - PayrollPresenter, 660
- TransactionContainerTest, 667–668
- TransactionFactory class, 461
 - class allocation in, 458, 464
 - metrics for, 465





- TransactionImplementation class
 - class allocation in, 458, 464
 - for cohesion, 462
 - metrics for, 465
 - object factory, 461
 - Transactions
 - Command pattern for, 302–304
 - forms for, 639–640
 - for payroll database, 615–616, 618–623
 - Transactions class, 448–449, 452
 - Transactions method, 669
 - TransactionSource class, 448
 - in Common Closure Principle, 450
 - in Reuse/Release Equivalence Principle, 454
 - TransactionSource interface, 408–409
 - TransactionsText method, 665
 - TransactionsText property
 - MockPayrollView, 662
 - PayrollWindow, 667
 - TransferTransaction class, 169, 172
 - TransferUI interface, 170, 172–173
 - Transition class, 586
 - Transition tables, 584–586
 - Transitions in state diagrams, 184. *See also* State diagrams
 - Transitive dependency, 155
 - Transparency
 - MONOSTATE, 337
 - SINGLETON, 334
 - TreeMap class, 179–180
 - class diagram for, 180–181
 - collaboration diagram for, 183–184
 - object diagram for, 182
 - sequence diagram for, 182–183, 228
 - TreeNode class, 179–181
 - Triangle class, 126–127
 - TurnOff method, 157–158, 265
 - TurnOn method, 157–158, 265
 - Turnstile class, 340–343, 580–581, 583–589, 593–595
 - TurnstileActions class, 591–592
 - TurnstileController class, 598
 - TurnstileController interface, 581, 583–584
 - TurnstileControllerSpooF class, 581, 597
 - TurnstileFSM class, 592–593
 - TurnstileLockedState class, 587
 - Turnstiles
 - finite state machines for, 580
 - Monostate pattern for, 338–343
 - nested switch/case statements for, 580–584
 - state diagrams for, 184, 208
 - State Machine Compiler for, 591–598
 - State pattern for, 586–589
 - transition tables for, 584–586
 - TurnstileState interface, 587
 - TurnstileTest class, 339–340, 581–582
 - TurnstileUnlockedState class, 587
 - TwoBallsInFrame method, 84, 89, 95
 - TwoConsecutiveMessages method, 236
 - Typing, static vs. dynamic, 441–442
- ## U
- UI class, 249
 - UI interface, 169–170
 - UIGlobals class, 170, 173
 - UIs. *See* User interfaces
 - UML. *See* Unified Modeling Language (UML)
 - UnboundedSet set, 144
 - UncrossIntegersUpTo method, 50–51
 - Undo method, 304–305
 - Unified Modeling Language (UML), 177–180
 - for back-end documentation, 192
 - CASE tools, 201
 - for communication, 189–191
 - diagrams. *See* Diagrams
 - for road maps, 191–192
 - Union dues and service charges
 - implementing, 379–380
 - pay periods for, 402–403
 - posting, 355





Index

731

- UnionAffiliation class, 357, 390, 392–393
 - for service charges, 379–380
 - for union dues, 403–406
 - Unit tests
 - for GeneratePrimes, 44
 - limitations of, 36
 - UnixModemConfigurator class, 544, 546–547, 551
 - Unlock method
 - Turnstile, 587, 589
 - TurnstileFSM, 592
 - Unlocked class, 342–343, 595–596
 - Unlocked state
 - in FSM diagrams, 208–209, 338
 - in state diagrams, 184
 - UnlockedTurnstileState class, 588
 - UPC codes, 221
 - Update method
 - ClockDriver, 476, 478
 - ClockObserver, 478
 - MockTimeSink, 480, 490
 - Observer, 488
 - SalaryObserver, 492
 - UpdateEmployeesTextBox method, 661
 - UpdateTransactionsTextBox method, 660–661
 - UpdateView method, 647, 649
 - Use cases, 219–220
 - alternate courses in, 221–222
 - for CoffeeMaker, 267–273
 - diagramming, 222
 - for payroll system, 351–359
 - sequence diagrams for, 228–231
 - writing, 220–221
 - Uselessness, zone of, 433
 - User interfaces, 637–639
 - for ATM system, 169–174
 - for CoffeeMaker, 267–273
 - designing, 639–640
 - implementing, 640–650
 - unveiling, 669–670
 - windows for
 - building, 650–657
 - payroll, 657–669
 - User stories
 - in extreme programming, 14
 - in planning, 24
 - splitting, 24–25
 - velocity of, 25
 - UserDatabase class, 189–190
 - UserDatabase interface, 334–335
 - UserDatabaseSource class, 335–336
 - UserInterface class, 268–270, 279
 - using statements, 428
 - USRModem class, 503–504
 - USRoboticsModem class, 499
 - Utility layer, 155
 - UtilityLayer layer, 155–156
- ## V
- Validate method, 302
 - ValidateHourlyPaycheck, 398–399
 - validateUser method, 204–205
 - Validating user state, 204
 - Validity in Liskov Substitution Principle, 142
 - Vapor classes, 265
 - Variable component stability, 429
 - Variables in class diagrams, 244
 - Velocity
 - feedback of, 26
 - of user stories, 25
 - Velocity charts, 28
 - VerifyPrime method, 53
 - VerifyPrimeList method, 53
 - Vertical inheritance, 247
 - View property, 660
 - ViewGetsUpdated method, 643, 649
 - ViewLoader interface, 662
 - Virtual properties, 140
 - Viscosity in design, 105–106
 - Visible events, use cases for, 220
 - Visit method
 - BOMReportTest, 557
 - ExplodedCostVisitor, 555





Modem, 548
 PartCountVisitor, 556
 UnixModemConfigurator, 546–547, 551
 VISITOR pattern, 543–544
 for databases, 539–540
 for modems, 544–548
 for report generation, 552–559
 VisitorCoverage method, 558
 Vlissides, John, 312

W

WaitForServiceThreads method, 215
 WakeLoggerThread method, 238–239
 wakeup command, 307
 WakeUpCommand class, 306
 WarmerPlate class, 263–265
 Web interfaces, 638
 Weekly builds, 421
 WeeklySchedule class, 386–387
 White box tests, 36
 Whole/part relationships in class diagrams, 252
 Whole teams in extreme programming, 14
 Width property, 138–141, 143
 Wiener, Lauren, 150
 Wilkerson, Brian, 150
 Williams, Laurie, 16
 Windows
 building, 650–657
 payroll, 657–669
 WindowViewLoader class, 658, 664–665, 669

WindowViewLoaderTest class, 663–664
 Wirfs-Brock, Rebecca, 150
 Wiring of commands, 301
 WithdrawalTransaction class, 169–170, 247
 WithdrawalUI interface, 170, 172–173, 247
 Write Printer module, 108
 WriteUser method, 335
 WumpusGame, 32–33

X

XML representation for reports, 566
 XmlAssemblyExtension class, 573–574
 XmlElement property
 XmlAssemblyExtension, 573–574
 XmlPiecePartExtension, 573
 XmlPartExtension class, 572
 XmlPiecePartExtension class, 573
 XP. *See* Extreme programming

Y

YIntercept property, 148

Z

Zones
 of exclusion, 432–433
 of pain, 433
 of uselessness, 433
 ZoomForUnix method, 547, 552
 ZoomModem class, 546, 551
 ZoomModemVisitor interface, 550

