

Index

A

- Acceptance tests, creation and execution of, 119
- Access rules to protect data, use of, 249
- Active Reviews for Intermediate Designs (ARID), 47–48
- Advanced metadata architecture, 239–241
- Agile Alliance Manifesto, 150–151
- Agile architecture. *See also* Agile modeling
 - Agile Alliance Manifesto, 150–151
 - communication modes, 153–154
 - complications, potential, 163–164
 - focus on the whole picture, 157
 - goals of, 158
 - implementation, 158–161
 - introduction to the organization, 161–162
 - iterative and incremental methods, 155–156
 - making your architecture attractive to your customers, 158
 - Model-Driven Architecture, in, 162
 - overview, 150–151
 - people, focus on, 153–154
 - principles of, 151, 153–158
 - project teams, coordination with, 156–157
 - simplicity, assume, 155
 - testing, 157
 - traditional approaches, potential complications with, 152–153
 - Unified Modeling Language, in, 162
 - Zachman Framework, in, 163
- Agile CIO, role of the, 262–263, 265–266
- Agile database techniques, 242–255
- Agile documents, 177
- Agile enterprise architecture, seven habits of, 275–276
- Agile modeling
 - approach to, 165
 - architects, implications for, 177–178
 - communication, 167
 - documents, agile, 177
 - feedback, 167
 - goals of, 166
 - practices of, 172–174
 - principles of, 168–171
 - traits of, 175–177
 - values of, 166–167
- Agility, 52
- Antipatterns, 87–89
- Application adaptor components, 186
- Application development costs, reduction of, 64–65
- Application layer, 18
- Applied software architecture viewpoints, 57–59
- Architects, roles of, 177–178, 260
- Architectural styles and patterns, 44, 60–62
- Architectural viewpoints, 56–59
- Architecturally significant use cases, 43

Architecture baseline, 43
 Architecture description languages, 48–49
 Architecture models, 278
 Architecture Trade-off Analysis Method (ATAM), 47–48
 ARID. *See* Active Reviews for Intermediate Designs (ARID)
 Asset responsible structure, 105
 Asynchronous, 16
 ATM, 18, 19
 Attacks by employees, 28
 Availability, 50, 53

B

Baseline data architecture, 227–229
 BDUF method. *See* Big design up front (BDUF) method
 Best practices, pursuit of, 261–262
 Big design up front (BDUF) method, 155
 Binding, 67, 75
 Biometric devices, 29
 Buildability, 51, 55
 Business architecture (conceptual architecture), 231
 Business components, 101
 Business data, 231–232
 Business domain, representation of, 71–72
 Business intelligence, 22–23
 Business logic tier, 181
 Business models, 277
 Business needs and presentation requirements, 179–180
 Business object modeling, 231
 Business process execution language (BPEL), 72
 Business technology integration, 278
 Business tier components, 187
 Business units, 105

C

Capability Maturity Model (CMM)
 advantages of, 125–126
 defined level, 123–124
 development, 120
 disadvantages of, 125–126
 framework, 122
 initial level, 122–123
 managed level, 124
 optimizing level of maturity, 124–125
 repeatable level, 123
 standard process for software development, 123

variations, 121
 Centralization of shared and current data, 248–249
 Character-Based User Interface (CUI), 209
 CIBMs. *See* Computation Independent Business Models (CIBMs)
 Class, Responsibility and Collaboration (CRC) card, 118
 Client/server architecture, 10–13
 CMM. *See* Capability Maturity Model (CMM)
 CMMi. *See* Integrated Capability Maturity Model (CMMi)
 Co-operative evaluation, 221
 COBOL copy books, 15
 Code smells, 46
 Code view, 59
 Coding the unit, 118–119
 COM. *See* Common Object Model (COM) components
 Commercial off-the-shelf systems (COTS), 102
 Common Object Model (COM) components, 12
 Common Object Request Broker Architecture (CORBA), 16, 77
 Common Warehouse Metamodel (CWM), 131
 Communication between hosts, 18
 Composites, 74
 Computation Independent Business Models (CIBMs), 130
 Conceptual architecture view, 58
 Conceptual components, 58
 Conceptual integrity, 51, 54–55
 Conformance, ensuring, 48
 Consultants, role of, 264–265
 Contextual inquiry, 218
 Continuous planning, 116–117
 Continuous coding, 116, 118–119
 Continuous design, 116, 117–118
 Continuous testing, 116, 119–120
 Contract mechanism, 70
 Contract specifications, 68
 Contracts, 15
 CORBA. *See* Common Object Request Broker Architecture (CORBA)
 Core assets, 91, 96, 98
 Core competencies reference architecture, 261
 Cost-benefit analysis, 217
 COTS. *See* Commercial off-the-shelf systems (COTS)
 Credit card information, protection of, 254
 CUI. *See* Character-Based User Interface (CUI)
 Custom controls, 185
 Custom screens, 185
 CWM. *See* Common Warehouse Metamodel (CWM)

D

Damage or compromise of business data, 30
 Data access logic components, accessing, 201–202
 Data architecture
 access rules to protect data, use of, 249
 advanced metadata architecture, 239–241
 agile approach, 225, 233
 agile database techniques, 242–255
 allowances for rapid changes in business
 requirements and database technologies, 248
 architectural process, 232–233
 baseline data architecture, 227–229
 business architecture (conceptual
 architecture), 231
 business data, 231–232
 business object modeling, 231
 business problem, 226–227
 centralization of shared and current data,
 248–249
 credit card information, protection of, 254
 data integrity, ensuring, 252
 data security, 241–242
 data validation, 250
 database iterations, 245
 development timeline, 244
 encryption methods, 254–255
 federated metadata approach, 235–236
 frameworks, 230–234
 layered data architecture, 230
 master lineage, 246
 metadata, 234–239
 monolithic database designs, 249
 normalization, 247–248
 OAGIS virtual business object model, 238
 online transaction processing (OLTP)
 databases, 247
 open standards to proprietary extensions,
 preference for, 253–254
 organizational context, 226–227
 replication of data, 250–251
 scripts, working with, 246–247
 star schemas, 246–248
 total performance *versus* local performance, 253
 transparency of location of data, 251–252
 validation/final review, 233–234
 Data integrity, ensuring, 252
 Data mining, 8
 Data security, 241–242
 Data tier, 181, 187
 Data validation, 250
 Database iterations, 245

Database management, 181
 Decomposition, 99
 Defined level, 123–124
 Denial-of-service (DOS) attacks, 30–31
 Development department, 104
 Development environment, setting up a, 44
 Development timeline, 244
 DHTML (Dynamic Hyper Text Markup Language),
 192, 193
 Diary methods, 218
 Direct attached storage (DRS), 24, 27
 Disaster recovery plan (DRP), 3, 31–34
 Discovery, concept of, 75
 Document-based user interfaces, 196–197
 DOM (Domain Object Model), 192
 Domain engineering unit, 106
 DOS. *See* Denial-of-service (DOS) attacks
 DRP. *See* Disaster recovery plan (DRP)
 DRS. *See* Direct attached storage (DRS)

E

EbXML, 77
 Empathic modeling, 220
 Encryption methods, 254–255
 Enhancing system value, role in, 15–17
 Enterprise application integration (EAI), 7
 Enterprise architectural model, 1
 Enterprise management discipline, 146–147
 Enterprise Unified Process (EUP)
 additions to the RUP life cycle, 142–143
 adoption of, 147
 advantages of, 142
 enterprise management discipline, 146–147
 life cycle of, 143
 operations and support discipline, 145
 production phase, 144
 retirement phase, 144–145
 Error handling, 69
 Ethernet, 18, 20
 Ethnographic approach, 218
 EUP. *See* Enterprise Unified Process (EUP)
 Expected inputs and outputs, 69
 EXtended IDE (XIDE), 103–104
 EXtreme programming (XP)
 acceptance tests, creation and execution of, 119
 advantages of, 120
 approach, 114–115
 Class, Responsibility and Collaboration (CRC)
 card, 118
 coding the unit, 118–119
 communication, 115

continuous planning, 116–117
 continuous coding, 116, 118–119
 continuous design, 116, 117–118
 continuous testing, 116, 119–120
 disadvantages of, 120
 execution on spike solutions, 118
 feedback, 115
 integration methods, 119
 iterative XP process, 116
 pair programming, 118
 principles of, 115
 refactoring, role of, 118
 roles, 115–116

F

Facades, 74
 Fat-client, 182–183, 194–196
 FDDI, 18, 24
 Federated metadata approach, 235–236
 Flow labeling of packets, 20
 Focus groups, 218
 Frame Relay, 18
 FTP, 77, 78
 Functionality, 50, 53, 63, 66, 68
 Functionality matrix, 218

G

Generic native controls, 185
 GNU, role of, 263–264
 Group discussions/future workshops, 217–218
 Grouping product lines, 108–109

H

Hacker attacks, 30
 Hierarchical domain engineering unit model, 106–109
 Hosts, communication between, 18
 HTTP, 77
 Human-computer interaction (HCI) principles, 209–215

I

IDE. *See* Integrated development environment (IDE)
 IDL. *See* Interface Definition Language (IDL)
 Information models, 277
 Initial level, 122–123
 Integrability, 50, 54

Integrated Capability Maturity Model (CMMi), 121
 Integrated development environment (IDE), 44
 Integrated Product Development Capability Maturity Model (IPD-CMM), 121
 Interface Definition Language (IDL), 68
 Internal services, development of, 80–82
 Internally developed systems, 102
 Internationalization, 82
 Internet address mapping, 18
 Internet browser, 182
 Internet browser user interfaces, 191–193
 Internet, protection of the, 20
 Interviews, 218
 IPD-CMM. *See* Integrated Product Development Capability Maturity Model (IPD-CMM)
 IPv4, 20
 IPv6, 20, 21–22
 Iterative XP process, 116
 IUSR Project, 222

J

JAD (Joint Application Design) Workshops, 220
 Java, 16
 Java server pages, 14
 Java servlets, 14
 Java Swing application, 12
 Jxxxx Dxxxx Bxxxx Cxxxx (JDBC), 13

L

Laboratory-based observation, 220
 LAN technologies, 18
 Layered data architecture, 230
 Layers, 60–61
 Legacy applications, 7–9
 Local performance, total performance *versus*, 253
 Loosely coupled services, 73–75
 Low-fidelity prototype, 219

M

Maintenance costs, reduction of, 65
 Managed level, 124
 Master lineage, 246
 MDA. *See* Model-Driven Architecture (MDA)
 Message Oriented Middleware (MOM) technology, 16
 Messaging protocols, 77
 Messaging technology, 16
 Meta Object Facility (MOF), 131

Metadata, 234–239
 Metaphors, 60–62
 MFC. *See* Microsoft Foundation Classes (MFC)
 Microsoft Foundation Classes (MFC), 12
 Mixed responsibility model, 105
 Mobile devices, 183, 193–194
 Model-Driven Architecture (MDA), 130–133, 162
 Model-View-Controller (MVC) pattern, 60, 183–184
 Models, use of, 277–279
 Modifiability, 50
 Modular design, 72–73
 Module view, 58
 MOF. *See* Meta Object Facility (MOF)
 MOM technology. *See* Message Oriented
 Middleware (MOM) technology
 Monolithic database designs, 249
 Multicasting support, 18
 Multiprotocol Label Switching (MPLS), 20
 MVC. *See* Model-View-Controller (MVC) pattern

N

Native screens, 185
 Naturalistic observation, 217
 Network, 25
 Network address translation, 21–22
 Network attached storage, 25, 26, 27
 Network connectivity, 202–203
 Network protocols, 17–22
 Next-minute challenges, 266
 Nonfunctional requirements, 55–56
 Normalization, 247–248

O

OAGIS virtual business object model, 238
 Object Management Group (OMG), 130
 Object wrapper, 15, 16
 ODBC. *See* Open Database Connectivity (ODBC)
 Offline applications, 202–203
 OLTP. *See* Online transaction processing (OLTP)
 databases
 OMG. *See* Object Management Group (OMG)
 Online transaction processing (OLTP) databases,
 247
 OOBE. *See* Open Database Connectivity (ODBC)
 Open Database Connectivity (ODBC), 13
 Open source, role of, 263–264
 Open standards to proprietary extensions,
 preference for, 253–254
 Operation models, 277

Operations and support discipline, 145
 Optimizing level of maturity, 124–125
 Organization models, 278
 Organizational matrix, 257–258
 Out-Of-the-Box Experience (OOBE), 222–223
 Outsourcing, 258–259

P

P-CMM. *See* People Capability Maturity Model
 (P-CMM)
 Pair programming, 118
 Paper prototyping, 219
 Parallel design, 221
 Password policies, 29
 People Capability Maturity Model (P-CMM), 121
 Performance, 50, 52–53
 Physical data model, 170
 Physical view, 57
 PICMs. *See* Platform Independent Component
 Models (PICMs)
 PIMs. *See* Platform Independent Models (PIMs)
 Pipes and Filters, 60
 Platform independence, 76
 Platform Independent Component Models
 (PICMs), 130
 Platform Independent Models (PIMs), 130
 Platform Specific Models (PSMs), 130
 Portability, 50, 53
 Practical considerations, 273–274
 Presentation tier architecture
 application adaptor components, 186
 approach, 179
 business logic tier, 181
 business needs and presentation
 requirements, 179–180
 business tier components, 187
 components, 181–187
 custom controls, 185
 custom screens, 185
 data access logic components, accessing,
 201–202
 data tier, 181
 data tier components, 187
 database management, 181
 design recommendations, 187–188
 document-based user interfaces, 196–197
 fat-client, 182–183, 194–196
 generic native controls, 185
 internet browser, 182
 internet browser user interfaces, 191–193

- mobile devices, 183, 193–194
- model-view-controller (MVC) pattern, 183–184
- native screens, 185
- network connectivity, 202–203
- offline applications, 202–203
- presentation tier, 181, 188–189
- primary presentation tier, 181, 182–183
- process management, 181
- proprietary native controls, 185
- protocol gateway tier, 181
- resource components, 186
- rule-based pages, 192
- secondary presentation tier, 181, 183–186
- template-based pages, 192
- thin-client, 182
- user interface components, 184–185, 189–191
- user interface process components, 185–186, 197–201
- user system interface, 181
- widgets, 185
- Primary presentation tier, 181, 182–183
- Process management, 181
- Process view, 57
- Product development, 96
- Product lines. *See* Software product lines
- Production phase, 144
- Program office, 123
- Project initiation, 216–217
- Proprietary native controls, 185
- Protocol gateway tier, 181
- Pseudosynchronous, 16
- PSMs. *See* Platform Specific Models (PSMs)
- Publish-Subscribe, 60

Q

- Quality attributes, 49–56
- Quality of service agreements and SLAs, 69

R

- RAD (Rapid Application Development)
 - Workshops, 220
- Rapid prototyping, 220
- Rational Unified Process (RUP), 48, 133, 134–138, 141–142, 215
- Refactoring, role of, 118
- Registry, 69–71
- Registry protocols, 77
- Reliability, 50, 53
- Repeatable level, 123
- Replication of data, 250–251

- Resource components, 186
- Retirement phase, 144–145
- Reusability, 50, 54
- Rollout, 221
- Rule-based pages, 192
- RUP. *See* Rational Unified Process (RUP)

S

- SA-CMM. *See* Software Acquisition Capability Maturity Model (SA-CMM)
- SAAM. *See* Software Architecture Analysis Method (SAAM)
- SANs. *See* Storage area networks (SANs)
- Scenario building, 219
- Scripts, working with, 246–247
- SE-CMM. *See* Systems Engineering Capability Maturity Model (SE-CMM)
- SEC Order 4-460, 8
- Secondary presentation tier, 181, 183–186
- Security, 27–31, 50
- SEPG. *See* Software Engineering Process Group (SEPG)
- Serial Lines, 18
- Service contract, 63
- Service level agreements (SLAs), 23–24
- Service-oriented architecture
 - advantages of, 82–84
 - agility of the corporation, impact on, 66
 - antipatterns, 87–89
 - application development costs, reduction of, 64–65
 - benefits of, 63–66
 - best practices, 87
 - binding, 67, 75
 - business domain, representation of, 71–72
 - characteristics of, 67–77
 - contract mechanism, 70
 - contract specifications, 68
 - disadvantages of, 82–84
 - discovery, concept of, 75
 - error handling, 69
 - expected inputs and outputs, 69
 - functionality, integration of, 63, 66, 68
 - implementation issues, 82–84
 - interactions, 67
 - internal services, development of, 80–82
 - internationalization, 82
 - loosely coupled services, 73–75
 - maintenance costs, reduction of, 65
 - management considerations, 84–87
 - modular design, 72–73

- platform independence, 76
- pre- and post-conditions, 69
- quality of service agreements and SLAs, 69
- registry, 69–71
- service contract, 63
- support introspection, 75
- transparency of service location, 76
- transport binding, 76
- transport mechanism independence, 76
- web services, and, 77–79, 82
- well-defined interfaces and polices, 67–71
- Shared architecture, 98–101
- Shared components, 101–102
- Shared technology and tools, 102–103
- Sharing the usability test reports, 222
- Simple Object Access Protocol (SOAP), 77, 78
- Skeleton, 1567
- SLAs. *See* Service level agreements (SLAs)
- Smart card readers, 29
- SMTP, 77, 78
- SOAP. *See* Simple Object Access Protocol (SOAP)
- Sockets, 18
- Software Acquisition Capability Maturity Model (SA-CMM), 121
- Software architecture
 - 4+1 view model, 56–57
 - Active Reviews for Intermediate Designs (ARID), 47–48
 - agile methodologies, relationship with, 35–36
 - agility, 52
 - analyzing and evaluating the architecture, 47–48
 - applied software architecture viewpoints, 57–59
 - approach methods, 38–39
 - architectural styles, 44
 - architectural styles and patterns, 60–62
 - architectural viewpoints, 56–59
 - architecturally significant use cases, 43
 - architecture baseline, 43
 - architecture description languages, 48–49
 - Architecture Trade-off Analysis Method (ATAM), 47–48
 - availability, 50, 53
 - buildability, 51, 55
 - business case, 42
 - code view, 59
 - conceptual architecture view, 58
 - conceptual integrity, 51, 54–55
 - conformance, ensuring, 48
 - creating or selecting the architecture, 43
 - definition of, 36–37
 - demonstration, 41–48
 - design checklist, 45–46
 - designing the architecture, 44
 - development environment, setting up a, 44
 - development view, 57
 - execution view, 58–59
 - eXtreme Programming (XP), 35
 - functionality, 50, 53
 - impact on enterprise architecture, 35
 - implementing the design, 44
 - integrability, 50, 54
 - layers, 60–61
 - logical view, 56
 - metaphors, 60–62
 - Model-View-Controller (MVC) architecture
 - pattern, 60
 - modifiability, 50
 - module view, 58
 - nonfunctional requirements, 55–56
 - performance, 50, 52–53
 - physical view, 57
 - Pipes and Filters, 60
 - portability, 50, 53
 - process view, 57
 - Publish-Subscribe, 60
 - purpose of, 37–39
 - qualities, identify important, 44
 - quality attributes, 49–56
 - Rational Unified Process (RUP), 48
 - reliability, 50, 53
 - representing and communicating the
 - architecture, 46–47
 - requirements, understanding the, 42–43
 - reusability, 50, 54
 - role of a software architect, 37
 - security, 50
 - Software Architecture Analysis Method (SAAM), 47–48
 - stakeholders, 39–41
 - subsetability, 51, 54
 - testability, 50–51, 54
 - UML, 48–49
 - usability, 50, 53
 - use cases, selection of, 43
 - variability, 51
 - volatility, 46
- Software Architecture Analysis Method (SAAM), 47–48
- Software Capability Maturity Model (SW-CMM), 121, 122
- Software Development Life Cycle, 112–114
- Software Engineering Institute (SEI). *See* Capability Maturity Model (CMM)
- Software Engineering Process Group (SEPG), 123

- Software product lines
 - asset responsible structure, 105
 - benefits of, 96–97
 - business benefits, related, 97–98
 - business components, 101
 - business units, 105
 - commercial off-the-shelf systems (COTS), 102
 - components of, 91–92
 - core assets, 91, 96, 98
 - decomposition, 99
 - definition of, 95–96
 - development department, 104
 - domain engineering unit, 106
 - eXtended IDE (XIDE), 103–104
 - goals of, 95
 - grouping product lines, 108–109
 - hierarchical domain engineering unit model, 106–109
 - historical development of, 94–95
 - internally developed systems, 102
 - management, 96
 - mixed responsibility model, 105
 - product development, 96
 - shared architecture, 98–101
 - shared components, 101–102
 - shared technology and tools, 102–103
 - source code reuse, 98
 - systems, 102
 - technical components, 101
 - unconstrained model, 105
 - Source code reuse, 98
 - Spike solutions, 118, 157
 - Stakeholders, 5, 39–41
 - Star schemas, 246–248
 - Storage area networks (SANs), 24–25, 26, 27
 - Storage of data, and, 24–27
 - Storyboarding/presentation scenarios, 219
 - Stovepipe architecture, 6–7
 - Subnets, 17
 - Suboptimal architectures, 6
 - Subsetability, 51, 54
 - Supply-chain management, 7
 - Support introspection, 75
 - Surveys, 217
 - SW-CMM. *See* Software Capability Maturity Model (SW-CMM)
 - Systems, 102
 - Systems architecture
 - approach to infrastructure, 4–5
 - ATM, 19
 - attacks by employees, 28
 - biometric devices, 29
 - business intelligence, and, 22–23
 - client/server architecture, 10–13
 - components, 1
 - considerations, 5–6
 - context, 5–6
 - damage or compromise of business data, 30
 - definition of, 1
 - denial-of-service (DOS) attacks, 30–31
 - disaster recovery planning, and, 31–34
 - enhancing system value, role in, 15–17
 - hacker attacks, 30
 - legacy applications, 7–9
 - Multiprotocol Label Switching (MPLS), 20
 - network address translation, 21–22
 - network protocols, 17–22
 - password policies, 29
 - purpose of, 2
 - security considerations, 27–31
 - service level agreements (SLAs), 23–24
 - smart card readers, 29
 - stakeholders, 5
 - storage of data, and, 24–27
 - stovepipe architecture, 6–7
 - subnets, 17
 - suboptimal architectures, 6
 - thin-client architecture, 13–15
 - Transmission Control Protocol/Internet Protocol (TCP/IP), 17–19
 - UDP, 19
 - viruses and worms, protection against, 28
 - Systems Engineering Capability Maturity Model (SE-CMM), 121
- ## T
- Task allocation charts, 218–219
 - Task analysis, 219
 - TCO. *See* Total cost of ownership (TCO)
 - TCP stack, 18
 - TCP/IP. *See* Transmission Control Protocol/Internet Protocol (TCP/IP)
 - Technical components, 101
 - Technical leadership, role of strong, 259
 - Technical prototype, 157
 - Template-based pages, 192
 - Testability, 50–51, 54
 - Text-Based User Interface (TUI), 209
 - Thick-Client Graphical User Interface (GUI), 209
 - Thin-client, 12, 13–15, 182
 - Thought leadership
 - agile CIO, role of the, 262–263, 265–266
 - approach, 257

- architects, role of, 260
 - best practices, pursuit of, 261–262
 - consultants, role of, 264–265
 - core competencies reference architecture, 261
 - GNU, role of, 263–264
 - next-minute challenges, 266
 - open source, role of, 263–264
 - organizational matrix, 257–258
 - outsourcing, 258–259
 - technical leadership, role of strong, 259
 - Token Ring, 18
 - Total cost of ownership (TCO), 7
 - Total performance *versus* local performance, 253
 - Transmission Control Protocol/Internet Protocol (TCP/IP), 17–19
 - Transparency of location of data, 251–252
 - Transparency of service location, 76
 - Transport binding, 76
 - Transport layer, 18
 - Transport mechanism independence, 76
 - Transport protocols, 77
 - TUI. *See* Text-Based User Interface (TUI)
- U**
- UDDI, 77
 - UDP. *See* User Datagram Protocol (UDP)
 - UML. *See* Unified Modeling Language (UML)
 - Unconstrained model, 105
 - Unified Modeling Language (UML), 38, 48–49, 131, 134–135, 162, 176
 - Unit testing framework (XUnit), 44
 - Usability and user experience
 - Character-Based User Interface (CUI), 209
 - co-operative evaluation, 221
 - contextual inquiry, 218
 - cost-benefit analysis, 217
 - definition, scope of, 207–208
 - deployment and ongoing requirements, 221
 - design process, 215–216
 - design, development, and testing stage, 220–221
 - design, prototype, and evaluate requirements, 218–219
 - diary methods, 218
 - empathic modeling, 220
 - ethnographic approach, 218
 - focus groups, 218
 - functionality matrix, 218
 - gathering and analyzing requirements, 217–218
 - group discussions/future workshops, 217–218
 - human-computer interaction (HCI) principles, 209–215
 - implementation challenges, 206–207
 - interpretations of, 205
 - interviews, 218
 - JAD (Joint Application Design) Workshops, 220
 - laboratory-based observation, 220
 - low-fidelity prototype, 219
 - naturalistic observation, 217
 - Out-Of-the-Box Experience (OOBE), 222–223
 - paper prototyping, 219
 - parallel design, 221
 - project initiation, 216–217
 - RAD (Rapid Application Development) Workshops, 220
 - rapid prototyping, 220
 - requirements stage, 216–220
 - rollout, 221
 - scenario building, 219
 - sharing the usability test reports, 222
 - standardized techniques, 216–221
 - storyboarding/presentation scenarios, 219
 - surveys, 217
 - task allocation charts, 218–219
 - task analysis, 219
 - Text-Based User Interface (TUI), 209
 - Thick-Client Graphical User Interface (GUI), 209
 - usability context analysis, 217
 - usability design criterion, 218
 - usability planning, 217
 - usage metrics, 221
 - user experience components, 208–209
 - Voice Response User Interface (VRUI), 209
 - walk-through, 221
 - Web-Based User Interface (WUI), 208–209
 - Wizard of Oz prototyping, 219
 - Usage metrics, 221
 - Use cases, selection of, 43
 - User Datagram Protocol (UDP), 18, 19
 - User experience. *See* Usability and user experience
 - User interface components, 184–185, 189–191
 - User interface process components, 185–186, 197–201
 - User stories, 47
 - User system interface, 181
- V**
- Validation/final review, 233–234
 - Variability, 51
 - Viruses and worms, protection against, 28
 - Visual Basic, 12
 - Voice Response User Interface (VRUI), 209
 - Volatility, 46

W

- Walk-through, 221
- WAN technologies, 18
- Web Service Description Language (WSDL), 76, 77, 78
- Web-Based User Interface (WUI), 208–209
- Well-defined interfaces and policies, 67–71
- Widgets, 185
- Wizard of Oz prototyping, 219

X

- XIDE. *See* eXtended IDE (XIDE)
- XML Metadata Interchange (XMI), 130
- XML/XSLT (Extensible Markup Language/Extensible Stylesheet Language Transformation), 192
- XP. *See* EXtreme programming (XP)
- XUnit. *See* Unit testing framework (XUnit)

Z

- Zachman Framework (ZF)
 - advantages of, 129–130
 - agile architecture, in, 163
 - development of, 126
 - disadvantages, 129–130
 - framework, 127–128
 - principles of, 126