

# INDEX

---

- 80/20 rule 6
- Acceptance plan 126
- Acceptance testing 126, 270–271
- Accessibility, Web 322–323
- Actions, in user interface 195–196
- Adjusted function points (AFP) 143
- Adoption obstacles 77–78
- ADP *See* Automated Defect Prevention (ADP)
- AEP *See* Automated Error Prevention (AEP)
- AFP *See* adjusted function points (AFP)
- Agile programming process 12, 13, 358–359
- AJAX *See* Asynchronous JavaScript and XML
- Algorithmic and processing defects 208
- Alpha testing 270–271
- Analyzing
  - cost 301
  - defects 299–301
  - schedule 301
  - source code growth 294–296
  - test results 296–299
- Application logic
  - defining 185
  - dividing into modules 374–375
  - Web services 373
- Architect, role in ADP 59–61, 76
- Architectural and detailed design
  - best practices 168
  - critical attributes 168
  - defining policies for 172
  - managing changes 188
  - modules 178
- Architectural attributes
  - correctness 169
  - efficiency and scalability 170
  - interoperability 171
  - maintainability and extensibility 171
  - modularity 169
  - portability 171
  - reusability 170
  - security 172
- Architectural design 125
- Architecture-first model 12
- Ariane 5 41–42
- Assertions 213–214

- Asynchronous JavaScript and XML (AJAX) 28, 391–393
- Audits Sarbanes-Oxley Act (SOX) compliance 325–327
- Automated build 10
  - measurements 240–241
  - policy for 237–241
  - tracking data 241
- Automated build system 27, 43, 55, 69, 71, 77
- Automated Defect Prevention (ADP)
  - See also* Defect prevention and requirements management 86
  - automation *See* Automation
  - benefits 3–8
  - case for 1–13
  - infrastructure *See* Infrastructure
  - introduced 1–13
  - management, impact on 7–8
  - management, support of 77
  - obstacles 77–78
  - people, impact on 3–4
  - policies *See* Policies
  - practices *See* Practices
  - principles *See* Principles, of ADP
  - process improvement, impact on 6–7
  - productivity, impact on 5–6
  - quality, impact on 4–5
  - team, support for 77
  - goals 3–8
  - in development process model 38–39
  - principles *See* Principles, of ADP
  - product, impact on 4–5
  - team member roles in 27. *See also* Group, roles
  - technology for 26–27
- Automated Error Prevention (AEP) 2
- Automated reporting system *See* Reporting system
- Automated testing system 27, 43–44, 74, 75
- Automatic code generator 34, 40, 74, 73
- Automatic programming (AP) 41
- Automation
  - impact on communication 34
  - impact on people 32–33
  - impact on product quality 33–34
  - introduced 11
  - limitations of 39–40
  - need for 32
  - of deployment 303–304
  - of measurement 36
  - of practice enforcement 34–35
  - of test generation 35–36
  - overview and benefits 32–36
  - role in ADP 11, 20
  - uses of 40–41
- Behavioral patterns 176
- Best practices *See* Practices
- Beta testing 271
- Black box testing
  - conducting 256–260, 264–265, 375, 378–380
  - Web services 375, 378–380
- Bug tracking system *See* Problem tracking system
- Bugs *See* Defects
- Build system *See* Automated build system
- Builds *See* Automated builds
- Business logic Web services 373
- CAL *See* Current Activity Log (CAL)
- Capability, process 47, 289
- Case study 341–352
- Cause and effect graph 22. *See also* Fishbone diagrams
- Change
  - detecting unexpected *See* Regression testing
  - resistance to 36–37
  - tracking *See* Requirements, changing management 132
- Check-ins 295, 380
- Checkpoint 264
- Client, Web services 371
  - deployment 386–387
  - implementation 386–387
  - testing 387
- CMM 212, 328–337
- CMMI
  - continuous representation 336–337
  - implementing 328–337
  - maturity levels 330–335
  - problems with 6
  - staged representation 330–335

- COCOMO 141–143
- COCOMO II 141, 143–145
- Code Base Size stabilization 294–296
- Code consistency verifying 228
- Code construction 207–247
  - best practices 209–229
- Code descriptions 217
- Code documentation defects 209
- Code freeze 275
- Code generator *See* Automatic code generator
- Code ownership 62
- Code review
  - automation of 40
  - best practices 229, 250–271
  - conducting 260–263
  - practices for 28
- Code source control system policy for 229–236
- Code structure standards
  - cross-code correlation rules 214
  - functional and data flow rules 213
  - structural complexity rules 213
  - usability and accessibility rules 215
  - use of assertions 213
- Coding defects 208
  - algorithmic and processing defects 208
  - code documentation defects 209
  - code reuse defects 208
  - configuration and version defects 209
  - control, logic, and sequence defects 208
  - data defects 208
  - data flow defects 208
  - error handling defects 208
  - external hardware and software
    - interface defects 209
  - initialization defects 208
  - service, module, and object interface defects 208
  - usability defects 209
- Coding standards 210–225, 232
  - about 75
  - accessibility 215
  - applying 210
  - applying to Web services 376
  - assertions 213–214
  - automating 221–225
  - code structure 213–215
  - code structure standards 213–215
  - cross-code correlations 214
  - customized policy 241
  - data flow 213
  - defining 39, 43, 44
  - documentation 217–219
  - language-specific 215–217
  - measuring 220
  - selecting 219–220
  - style, layout, and formatting 212
  - technology-specific 215–217
  - tracking 220–221
- Comments format 218
- Commercial Off-the-Shelf Components (COTS) 125–126, 145
- Communication problems 34
- Competency zone 4, 32
- Compiling code 69–71
- Complexity, system 2, 32
- Compliance, regulatory 7, 322–328
- Conceptual test cases *See* Test cases, conceptual
- Confidence Factor 47–48, 64, 304–307
- Configuration and version defects 209
- Configuration management system *See* Source control system
- Construction and testing phase 39
- Constructive Cost Model (COCOMO) 141–143
- Continuous testing 392–393
- Contracts, for outsourced projects 315–316, 321
- Control limits 47
- Control, logic, and sequence defects 208
- Cost
  - ADP impact on 5–6
  - analyzing 301
  - and requirements management 73
  - estimating 39, 135–146
  - evaluating 76
  - metrics 63–64, 65
  - of defects 4, 6, 23–24, 101–102
  - reduction 23–24
- Cost drivers (CD) 144
- COTS *See* Commercial Off-the-Shelf Components (COTS)
- Coverage, test 126, 253–256

- Creational patterns 176
- Critical paths 147
- Cross-code correlation rules 214
- Culture, group *See* Group, culture
- Current Activities Log (CAL) 64, 132, 137, 148, 263
- Customer training plan 127
- Cutoff date 38
- Cyclical referencing 383–384
- Cyclomatic complexity 242
  
- Data defects 208
- Data flow 208, 213
- Data flow defects 208
- Database
  - infrastructure for 155–156
  - Web services 373
- Database testing, Web services 386
- Decision making *See* Project management
- Defect analysis and prevention 271–273
- Defect casual analysis 361
- Defect detection 25, 281
- Defect prevention *See also* Automated Defect Prevention)
  - and risk management 129–131
  - attitudes towards 5
  - benefits 21–24
  - commitment to 77
  - concept 25, 30
  - conducting 271–273
  - cost reduction 23–24
  - defining the process 129
  - definition 21–24
  - extended view 22
  - history 24–26
  - need for 5
  - quality, impact on 21–22
  - requirements for 5
  - requirements-related defects 90
  - software industry resistance to 26
  - traditional view 21–22
  - user satisfaction, impact on 22–23
  - vs. error detection 25
  - in the automotive industry 24–26
- Defect tracking system *See* Problem tracking system
  
- Defects *See also* Coding defects, Design defects
  - analyzing 299–301
  - cost of *See* Cost, of defects
  - definition 21
  - requirements-related 86, 90
  - time to fix 102
- Deming, W. Edwards 21
  - auto industry, impact on 24–26
  - five step error prevention procedure 46
  - methodology 24–26, 30, 46, 59
- Deployment
  - best practices 301–309
  - phase 39
  - plan 127
  - readiness 64, 301–307
  - release 307
  - staging system 301–302, 307
  - Web services 384–386
- Deployment process
  - automation 303–304
  - nightly 373
- Design
  - automation of 40
  - complexity 130
  - patterns 28
  - Web services 374–375
- Design by Contract 214, 260
- Design defects
  - algorithmic 166
  - control, logic, sequence 166
  - data 167
  - defined 166
  - external hardware, software interface 167
  - service, module, object functional 166
  - service, module, object interface 166
  - usability 167
- Design documents, storage and inspection 187
- Design patterns 175
  - behavioral patterns 176
  - creational patterns 176
  - example 176
  - structural patterns 176
- Design policies 167

- Detailed design defined 166
- Developer, role in ADP 59–61, 76
- Development database 155–156
- Development process models *See*
  - Process models
- Documentation standards 217–219
  - code descriptions 217
  - comments format 218
  - naming conventions 217
- Effort
  - analogy 137, 138–141
  - estimating 135–146
- Embedded mode, COCOMO 142
- Encoded coding style, avoiding 384
- EQF *See* Estimate Quality Factor (EQF)
- Error detection 25
- Error handling defects 208
- Error prevention concept *See* Defect prevention concept
- Errors *See* Defects
- Estimate quality factor (EQF) 146
- Estimation
  - by effort analogy 138–141
  - by parametric models 141–145
  - by Wideband Delphi 137–138
  - iterative adjustments 145
  - metrics 149–150
  - of costs 120, 135–146
  - of effort 135–146
  - of schedules 120
  - quality factor 145
  - quality metrics 150
- EVM (Earned Value Management)
  - system 28
- Extended planning and design phase 39
- Extensibility 171–172
- External hardware and software
  - interface defects 209
- Extreme Programming 12, 13, 225, 358–359
- Fault density 33
- Feature requests, entering 108
- Features
  - implemented 64
  - tracking 71
- analyzing 290–294
  - implementation status 290
  - stabilization 291–294
- Feedback loop 12, 59, 96, 97, 106
- FERPA 7, 13
- Fishbone diagrams 21–22, 42, 111, 157
- FIT *See* Framework for Integrated Testing
- Flow, state of 4
- Focus on prevention 224
- Formal Requests for Proposals (RFPs) 315
- Fourth generation programming
  - languages 41
- FPA *See* Function Point Analysis (FPA)
- Framework for Integrated Testing (FIT) 86
- Function Point Analysis (FPA) 143
- Function points 143
- Functional and data flow rules 213
- Functional testing 266
  - for Web services 385–386
- Functional tests, from conceptual test cases 100
- Geographically-distributed development
  - See also* Outsourcing
  - group organization 56–58
  - infrastructure location 58
  - requirements management 106
- Gramm-Leach Bliley 7, 14
- Graphical User Interface (GUI)
  - architectural patterns 195
  - critical attributes 190–192
  - defining design policy 193–195
  - design best practices 189
  - prototyping 197
  - testing 197
- Group
  - culture 56, 60, 61–63, 68, 277, 320–321, 351
  - extension 156–157
  - formation and organization 56–58
  - infrastructure location 58
  - obstacles to ADP adoption 77–78
  - pilot 56–58
  - reorganization 156
  - roles 58–61, 156
  - size 56

- training 44–45, 61, 157
- work hours 62–63
- workflow *See* workflow
- Group-by-group implementation 36–37
- GUI *See* Graphical User Interface (GUI)
- HIPAA 7, 14
- Human intelligence vs. automation 32–33, 39–40
- Human nature, and ADP 2–3
- Hunt and fix mentality 24
- IDE *See* Integrated Development Environment
- Implementation
  - group-by-group 36–37
  - incremental 8, 20, 36–38, 78
  - practice-by-practice 37–38
- Implementing
  - modules 375–381
  - Web service client 386–387
  - Web services interfaces 376–377
  - WSDL document 381–384
  - group workflow (case study) 348
  - Java coding standards (case study) 341–352
- Incremental development *See* Process models, incremental
- Incremental implementation of ADP 8, 20, 36–38, 78
- Indicators 8, 31, 64–66
- Infrastructure
  - expanded technology 73–74
  - extending 150–157
  - for automation 44–45
  - for database-related development 155–156
  - for embedded systems development 151
  - for measurement 44–45
  - for outsourcing 317–318
  - for practice enforcement 44–46
  - for tracking 44–45
  - for web application development 151
  - for workflow support 44–46
  - initial 53–56
  - integrating people and technology 75–77
  - intermediate technology 72–73
  - legacy infrastructure components 76
  - minimum technology 27, 55, 68–72
  - overview 26–27, 53–54
  - people 26–27, 54–55, 56–63, 156–157
  - role in ADP 2, 20
  - role in software production line 26
  - technology 63–75, 151–156
  - usage 75–77
- Initial planning and requirements phase 39
- Initialization defects 208
- Inline schemas 383–384
- Integrated Development Environment (IDE) 55, 68, 70
- Integration testing 263–265
- Ishikawa (fishbone) diagrams *See* Fishbone diagrams
- Iterative process *See* Process models, iterative
- Language and technology specific standards 215–217
- Legacy code, documentation 106
- Lifecycle, software 14
- Load testing 44, 266–268
  - for Web services 389
- Maintainability 171–172
- Make files 77
- Management *See* Project management
- Manufacturing industry 24–26
- Mars Polar Lander (MPL) failure *See* MPL failure
- Measurement
  - automation of 36, 41
  - for estimation 149–150
  - overview 31
  - requirements 109–110
  - role in ADP 20
- Metrics 31, 41, 54
  - code 149
  - cost 63–64, 65
  - defect 149
  - estimate quality 149
  - requirements 149
  - test 149
- Milestones 147–148, 158–160

- Mindset, for defect prevention 10
- Mistakes, learning from 28–30, 104
- Mis-use cases 97
- Mock-up server 153–154
- Model, software development *See*
  - Process models
- Modularity 169
- Module interfaces 226
  - defining 375
- Modules
  - designing interfaces of 181
  - diagram 180
  - implementing 375–381
  - modeling 182
  - test-first approach 376–377
- Monitoring
  - non-intrusive 307–309
  - outsourced projects 321
  - Sarbanes-Oxley Act (SOX)
    - compliance 324–328
- Monitoring system *See* Reporting system
- MPL failure 361–366
  
- Naming conventions 29, 217
- National Institute of Standards and Testing (NIST) 4, 250
- NIST *See* National Institute of Standards and Testing
- Non-intrusive monitoring 307–309
  
- Objectives, defining 121–124
- Object-oriented unified process 12, 357–358
- Obstacles to ADP adoption 77–78
- Offshoring *See* Outsourcing
- Open Web Application Security Project (OWASP) 91
- Operation manuals 127
- Organic mode, COCOMO 142
- Outsourcers
  - contracts 315–316
  - finding 315
- Outsourcing *See also* Geographically-distributed development
  - best practices 312–321
  - completing projects 321
  - employee exchange 320
  - implementation 319–321
  - infrastructure 317–318
  - managing 7–8, 312–321
  - monitoring projects 321
  - planning 315–319
    - verifying contract fulfillment 321
- Outsourcing process establishing 313
- OWASP *See* Open Web Application Security Project (OWASP)
  
- Pair programming 28–29, 261
- Parallel development 230–231
- Parametric models 137, 141–146
- Pareto diagrams 21
- People *See also* Groups
  - ADP impact on 2–4
    - and software development plan 54–55
    - impact of automation on 32–33
  - infrastructure *See* Infrastructure, people
  - role in ADP 26–27
- Performance testing 266–268
  - for Web services 389
- Person months 135
- Person years 135
- Phased implementation *See* Incremental implementation
- Pilot group 37
- Plan, software development *See* Software Development Plan
- Planning
  - extended 119–160
  - initial 53–56
  - outsourcing 315–319
  - problems 120
    - Web services 369–375
- Polar Lander failure *See* MPL failure
- Policies
  - and requirements 102
  - defining 39
  - for code reuse 175
  - for memory management 173
  - for problem tracking system 273–278
  - for regression testing system 278–279
  - for reporting system 66–68
  - for requirements management system 107–109
  - for using inheritance in class definition 174
  - introduced 9–10

- role in ADP 9–10
  - security 91–92
- Portability 171
- Practice-by-practice implementation 37–38
- Practices
  - automation 34–35
  - code construction level 28–29
  - configuring infrastructure for 76
  - customized 9, 20, 29–30, 45–46, 110–112, 242
  - deployment and transition 301–309
  - enforcement 34–35
  - general 9, 20, 28–29
  - granularity levels 28–29
  - introduced 9
  - measurement 31
  - organization level 28
  - outsourcing 312–321
  - requirements 87–105
  - role in ADP 9, 20
  - SOA 369–389
  - standardization across team 44–46
  - tracking 32
  - Web services 369–389
- Preventive action plan 21, 44, 111–112
- Principles, of ADP 8–9, 19–48
- Problem tracking system 27, 55, 104
  - about 69
  - as idea repository 78–79
  - group culture 277
  - measurements of data 277
  - policy for 273–278
  - role of 71
  - tracking of data 277
- Process
  - and software development plan 55–56
  - capability 47, 289
  - control trends 288
  - stabilization 288–289
  - variations 288
- Process improvement
  - ADP impact on 6–7, 11
  - from defect prevention 25
- Process models
  - Agile 12, 13, 358–359
  - and defect prevention 11
  - architecture-first 12
  - construction and testing phase 39
  - deployment phase 39
  - extended planning and design phase 39
  - Extreme Programming 12, 13, 225, 358–359
  - incremental 11–12, 14, 355
  - initial planning and requirements phase 39
  - iterative 11–13, 87, 119, 126
  - Object-oriented unified process 12, 357–358
  - overview of 11–13, 353–359
  - phases of 12–13
  - RAD 12, 353–355
  - selecting 128–129
  - spiral 12–14, 355–357
  - waterfall 11–13
  - with ADP 38–39
- Process status indicators *See* Indicators
- Production line, software 26
- Production server 153, 307
- Productivity
  - ADP impact on 5–6
  - impact of automation on 35–36
- Project artifact repository *See* Shared project artifact repository
- Project management
  - ADP impact on 7–8
  - analyzing and responding to trends 31–32, 65–68, 76, 78–80, 287–301, 304–307
  - artifact organization/tracking 158
  - change management 132
  - defining artifacts and deliverables 124–128
  - defining objectives 121–124
  - estimating effort 135–146
  - milestone scheduling/tracking 158
  - outsourced projects 312–321
  - planning 119–160
  - risk management 129–131
  - scheduling 146–149, 157–158
  - Work Breakdown Structure (WBS) 132–135
- Project manager, role in ADP 58, 76
- Projects
  - existing 87, 105–106, 113, 123–124, 128, 129, 131, 135, 138, 139, 149

- new 87, 105–106, 112–113, 122–123, 127–128, 129, 131, 133–135, 138, 139, 148–149
- progress trends 290–301
- approval status report 325
- Prototype 12, 14, 136
  - creating 99–100
  - evolutionary 98–99, 101, 104
  - throwaway 98, 101
  - vertical 130
  - working 137
- Prototyping development 353–354
- Proxy server Web services 371
- QA, role in ADP 59–60
- QoS *See* Quality of Service (QoS)
- QSM *See* Quantitative Software Management (QSM)
- Quality
  - ADP impact on 4–5
  - attributes of 33
  - control 24
  - defect prevention impact on 22–23
  - metrics 33
  - initiatives 6
- Quality of Service (QoS) 89
- Quantitative Software Management (QSM) 136
- RAD *See* Rapid Application Development (RAD)
- Rapid Application Development (RAD) 12, 353–355
- Rapid prototyping *See* Rapid Application Development (RAD)
- Regression testing 268–270, 298, 374, 380, 384, 386, 387
- Regression testing system
  - about 72, 73
  - measurements related to 279, 298
  - policy for 278–279
  - tracking of data 279
- Regulatory compliance 7, 322–328
- Release
  - assessment of readiness 304–307
  - deployment 307
- Release readiness, assessing 304–307
- Reporting system
  - about 55, 63–64
  - and source control data 79–80
  - policies for 66–68
  - reviewing reports 126
  - role 65–66
- Requirements
  - and ADP 86
  - challenges 85
  - changing 103–105, 114–115
  - core 94
  - correlating to use cases 93
  - cross-reference to features 108
  - defect prevention 90
  - defects 86
  - defining 85–115
  - elicitation techniques 91, 98
  - functional priorities 93–94
  - gathering and organizing 89–93
  - implementation priorities 93–95
  - inspection 101–103
  - management strategies 85–115
  - managing 85–115
  - mapping to modules 178
  - measurements 109–110
  - non-core 94
  - prioritizing 93, 112–114
  - security 89, 91
  - severities 93
  - storing 92
  - tracking 72–73, 100, 110
  - types of 89
  - Web services 370–371
  - stability index 64
- Requirements management system 9, 27, 33, 34, 39, 45, 69, 71, 72–73, 92, 103
  - measurements 109–110
  - policies for 107–109
  - role of 73
  - tracking 72–73, 110
- Reuse, of code 106, 145, 170, 208
- Risk
  - common 131
  - management 129–131
  - of design complexity 130
  - plan 130
  - project-specific 131
  - sources 130
- Roles, in ADP 27, 55

- Root cause analysis and prevention 10, 71, 77, 110, 158
  - and defect prevention 21–24, 25
  - examples 41–43, 280–283
  - requirements for 60
  - role in ADP 20
- Rule of separation 314
- RUP *See* Unified Process
  
- Sandbox 229, 230, 233
- Sarbanes-Oxley (SOX)
  - reports 325–327
  - tracking/monitoring compliance 324–328
- Sarbanes-Oxley Act (SOX) 7, 14, 323–328
- scalability 170
- Scale drivers (SD) 144
- Scenario-based tests, creating 386
- Schedules
  - adjusting 148, 301
  - analyzing 301
  - preparing 146–149
- Scripts
  - build 77
  - deployment 127
  - test 40
- SDP *See* Software Development Plan (SDP)
- Section 508 7, 14, 322–323
- Security
  - defined 172
  - policy 91–92
  - project management concerns 7
  - requirements 91–92
  - testing 266–268
  - Web services 387–388
- SEI *See* CMM, CMMI
- Semidetached mode, COCOMO 142
- Server deployment, Web services 384–386
- Server stubs 387
- Server, Web services 373
- Service contracts 226
- Service Oriented Architecture (SOA)
  - 28, 165, 178, 369–389, *See also* Web services
  - functional requirements 370
  - non-functional requirements 371
- Shared Project Artifact Repository 27, 69, 72
- Six Sigma 47, 329
- SLIM 136
- SLOC *See* Source Lines of Code (SLOC)
- SOA *See* Service Oriented Architecture (SOA)
- Software architecture 165–166
- Software design
  - model-driven 166
  - pattern 166
- Software Development Plan (SDP) 39, 54–56, 120–121–124, 137
- Software lifecycle 14
- Software process 55–56
- Software Project Management Plans (SPMP) *See* Software Development Plan
- Software Requirements Specification (SRS) document 89, 96, 107–108, 121, 125
- Source control system 27, 70, 55
  - about 69,71
  - checking in code 380
  - checking in JUnit tests 380
  - checking in verified tests 380
  - measurements 234–236
  - policy for 229–236
  - tracking data 236
- Source Lines of Code (SLOC) 64, 99, 140–141, 143
- SOX *See* Sarbanes-Oxley Act (SOX)
- Spiral process *See* Process models, spiral
- SRS document *See* Software Requirements Specification (SRS) document
- Stability, process 47
- Staging server, for Web services 384
- Staging systems 127, 150–155, 301–302, 307
- Statement of Work (SOW) 315
- Static analysis 70, 221–225, 263
- Static analyzer 34, 73, 75
- Statistical control limits 31
- Statistical processes 31
- Statistical quality control 24
- Status indicators *See* Indicators

- Status assessment 20, 47–48, 64, 304–307
- Stop-test criteria 126
- Stress testing 266–268
- Structural complexity rules 213
- Structural patterns 176
- Stubs
  - server 387
  - unit testing 258
- Sub-module interfaces defining 375
- System Administrator, role in ADP 60
- System architecture 165
- System compliance, verifying
  - sustainability 327
- System security, verifying 327
- System testing 265–268
  - acceptance testing 270
  - functional testing 266
  - performance and load stress testing 266–268
  - performance and load testing 266
  - regression testing 268
  - security testing 266
- Target average values 31
- TDD (Test-Driven Development) *See* Test-Driven Development (TDD)
- Team *See* Group
- Team workflow *See* Workflow
- Technology index 33
- Test cases
  - conceptual 100–101
  - correlating to use cases 93
  - for implemented features 108–109
  - for requirements 108–109
  - from use cases 100
  - refining 186
  - status 64
- Test engineer, role in ADP 59, 61, 76–77
- Test generation, automation 35
- Test pass rate 126
- Test results
  - analyzing 296–299
  - reviewing 68
  - stabilization 296–299
- Test-Driven Development (TDD) 39, 225, 227, 293
- Test-first approach 225
  - modules and sub-modules 376–377
- Testing
  - acceptance 270–271
  - alpha 270–271
  - beta 271
  - client functionality 387
  - coding standards *See* Coding standards
  - conducting 249–271, 375–389
  - continuous 392–393
  - fixed defects 380–381
  - functional 266–268
  - integration 263–265
  - performance and load stress 266–268
  - regression 268–270
  - security 266–268
  - system 265–268
  - unit *See* Unit testing
  - weakness of 2, 24–25
  - Web services 375–389
  - Web services security 388
- Testing tools *See* Verification tools
- Tracking
  - for estimation 149–150
  - overview 31–32
  - requirements 110
  - role in ADP 20, 31
  - Sarbanes-Oxley Act (SOX) compliance 324–328
- Training
  - customers 127
  - team 44–45, 61, 157
- Transition, best practices 301–309
- Trend analysis 68, 287–301
- UML *See* Unified Modeling Language (UML)
- Unified Modeling Language (UML) 125, 182–185
- Unified process 12, 14
- Unit testing
  - about 75
  - automating 254–256, 260
  - black box 256–260
  - conducting 225–226, 227–228, 249, 251–260, 377–381
  - coverage 253–256
  - from conceptual test cases 100
  - measuring 254, 259
  - tracking 254, 259–260

- Web services 377–381
  - white box 251–256
- Usability and accessibility rules 215
- Usability defects 209
- Use cases
  - change in 88, 105
  - conceptual test cases 100
  - correlating to code 93
  - correlating to requirements 97
  - correlating to tests 93
  - developing 90, 95–97
  - developing 125
  - for estimation 136
  - measuring 88
  - per module 198–199
  - positive 97
  - system testing 266–267
  - tracking 93, 105
  - negative 97
- User interface design 167, 189–198
- User manuals 127
- Variable names 29
- Verification
  - compliance sustainability 327
  - conducting 249–271, 375–389
  - contract fulfillment for outsourced projects 321
  - fixed defects 380–381
  - security 327
  - Web services security 387–388
  - Web services performance 389
  - WSDL documents 384
- Verification tools
  - integrating into infrastructure 69
  - selecting 63
  - using during development 70
- Verified tests checking into source control 380
- Version control system *See* Source control system
- Vision and scope document 39, 96, 157
  - creating 54, 87–89
  - modifying and storing 124–125
  - policies for 107
- Waterfall process 11–13, 15
- WBS *See* Work Breakdown Structure (WBS)
- Web Service Description Language (WSDL) *See* WSDL
- Web services
  - application logic 373, 374–375
  - best practices 369–389
  - business logic 373
  - client 371, 386–387
  - coding standards for 376
  - constructing 375–389
  - databases 373, 386
  - deployment 384–387
  - functionality testing 385–386
  - infrastructure for 154–155
  - load testing 389
  - nightly deployment process 373
  - nightly test process 374, 380–381, 384, 386, 387
  - performance 389
  - planning and design 369–375
  - proxy server 371
  - security 387–388
  - security-enabled 388
  - server 373, 384–386
  - test execution 385–386
  - testing 375–389
  - verifying fixed defects 380–381
- Web Service Description Language (WSDL) 373
- White box testing
  - bottom-up approach 251
  - conducting 228, 251–256, 377–378
  - Web services 377–378
- Wideband Delphi 137–138
- Work Breakdown Structure (WBS) 132–135, 137
- Workflow
  - automation of 34
  - examples of 44–46
  - infrastructure for 44–46
  - role of reporting system in 67–68
  - using infrastructure components 70–72, 75–77
- WSDL documents 373
  - creating regression tests 384
  - deploying 381
  - implementing 381–384
- XML validation 383–384
- XP *See* Extreme Programming