



# Index

## A

AccuRev, 181–183  
ACME-Assembling Configuration Management  
  Environments (for software), Web site  
  for, 166  
Active codelines, new work on, 154  
Active development line, 151, 160  
  merging task branch into, 162  
  and shipping issues, 154  
*Active Development Line* pattern, 42, 57, 59–66,  
  68, 118, 124, 142, 148, 154, 158, 173  
Activities, with ClearCase UCM, 185  
Agile Manifesto, 16  
Agile software development, role of software con-  
  figuration management in, 7–8  
*Agile Software Development Ecosystems* (High-  
  smith), 17, 66  
Alexander, Christopher, 35, 36, 38–39  
*Antipatterns and Patterns in Software Configura-  
  tion Management* (Brown et al.), xxxiv  
ANT tool, 86, 122  
Architecture, 25–28, 30  
Architecture/module structure, 27  
*Are Your Lights On?* (Gause and Weinberg), 66  
*Art of Software Testing, The* (Myers), 133, 139  
Association for Configuration and Data Manage-  
  ment, Web site for, 168  
Atomic change transaction model, with  
  Perforce, 178  
Atomicity, 112  
Automated regression testing, 139  
Autonomous work, 22

## B

Babich, Wayne A., xxxiii  
Bad ideas, rolling back, 144  
Bar, Moshe, xxxiv, 193  
Baselines, UCM, 186  
Bays, Michael, xxxii, xxxiii, 57, 151, 161  
Beck, Kent, 131, 133  
*Becoming a Technical Leader* (Weinberg), 17  
Berliner, Brian, 110  
Best practices, for software configuration man-  
  agement, 37  
Bismarck, Otto von, 15  
BitKeeper, 122, 179–181  
Booch, Grady, 17, 132  
Branches, 110  
  creating when shipping, 149  
  of files, xxix  
  for isolation, 160–162  
  reasons for, 55  
  staircase of dependent, 150  
  uses for, 50–52  
Branching, 30, 41, 50  
  and code freeze, 5  
  of codelines into release codelines,  
    154–155  
  consequences with, 53–54  
  diagrams, xxxi  
  entire codeline, xxxi  
  fear of, 52–54  
  lazy,” 161  
  before releases, 155  
  releases off from mainline, 151





## 208 Index

---

- for special situations, 55–56
- staircase, 52, 53
- third-party code, 151
- Branching models
  - company handling of, 7
  - simplifying, 54–56
- Branch spec, in Perforce, 178
- Brown, William J., xxxiv
- Bugs, 30, 83
  - and branching, 55
  - fixing, 56, 126, 148, 149, 150, 151, 154
- Build environment, 84
- Build errors, 95, 98, 102
  - fixing, 90
  - identifying, 101
- Build-everything approach, 93
- Buildings, pattern language for building of, 35, 45
- Build machine, 100
- Build management, xxvii, 14
- Builds, 20
  - and integration of changes, 89
  - nightly, 88, 90, 91, 93, 100, 139
  - periodic, 21–22, 25
  - preliminary, 61
  - smoke tests for, 125–126
  - unpredictable, 6
- Build scripts
  - and *Repository* pattern, 85
  - for workspace, 81
- Build tools, customized versions of, 109
- Build tree, and version control tree, 86
- Build workspace, 76
- C**
- C++, 27
- Cabrera, Ralph, xx, xxiv
- Cascade, 52
- Centralized builds, 99, 100–102
- Change documents, in PVCS Dimensions, 189
- Change packages, in MKS Source Integrity, 192
- Changes, 66, 83, 136
  - batching, 114
  - centralized integration build and compatibility of, 88
  - to codeline structure, 62–63
  - combining at once, 69
  - integrating as it happens, 70
  - integrating into task branch, 161
  - isolating work and controlling, 72–76
  - in maintenance line, 151
  - managing parallel flows of, 68
  - merging, 50, 95
  - and private versions, 142–144
  - and release engineering branch, 156
  - tests for, 138–139
  - tracking, 85, 112
- Change-sets, in BitKeeper, 179, 180
- Change tasks, reasonable, 114
- Chaos theory, xiv
- Check-in process, 64, 65
- Check-ins, 20, 23, 112, 113, 114
- Check-in Task, in CM Synergy, 187
- Checkout process, 75
- Checkpointing changes, and developer's workspaces, 144, 145
- Checkpointing projects, in MKS Source Integrity, 193
- Checkpoints, in AccuRev, 182
- CLASSPATH(s), 91, 109
- Clean builds, 93–94, 99
- ClearCase, 10, 57, 77, 122
  - Base functionality (non-UCM), 183–185
  - Unified Change Management (UCM), 185–186
- Client spec, in Perforce, 177
- Client view, in Perforce, 177
- Client workspace, in Perforce, 177
- CM Crossroads-Online Community and Resource Center for CM Professionals Web site, 165–166
- CM Synergy, 187–188
- cmntalk mailing list, 165
- CM Today-Daily Configuration Management News Web site, 166
- CM Today* newsletter, 165
- CM Yellow Pages, 165, 166
- Cockburn, Alistair, 31



- Code
    - branching, 151
    - dead-end, 151
    - software as sum of, 22
    - testing, 65, 124
    - third-party, 105, 106
  - Code base, and reliable builds, 98–100
  - Code Complete* (McConnell), 31, 77, 139
  - Code freezes, 5, 6, 114, 154, 155
  - Codeline,
    - purposes of, 118
    - quality of, 130
    - regression tests for stability of, 138–139
    - rules for, 119
    - stabilizing for impending releases, 154
    - task branch integrated with, 162
    - for third-party code, 106–110
  - Codeline diagrams, xxxi
    - notation, xxxii
    - symbols, xxxii–xxxiii
  - Codeline Policy* pattern, 43, 66, 112, 117–122
  - Codeline-related patterns, 42
  - COM, 27, 74, 109, 110
  - Commit
    - in BitKeeper, 181
    - with CVS, 175
  - Communication, 21, 83, 94, 98
    - and distance, 28–29
    - effective, 12
    - team, 13
  - Complexity, reducing, 150
  - Complexity theory, xiv
  - Components, UCM, 186
  - “Concepts in CM” (Dart), 167
  - Concurrency, 27
  - Concurrent work, 26
  - Configuration control, xxvii, 13
  - Configuration files
    - and *Repository* pattern, 85
    - for workspace, 80
  - Configuration identification, xxvii, 13
  - Configuration Management: The Missing Link in Web Engineering* (Dart), 194
  - Configuration management environment, 23
  - “Configuration Management Models in Commercial Environments” (Feiler), 167
  - Configuration management patterns, 37–39
  - “Configuration Management Patterns” (Berczuk), xxi
  - Configuration management process, and team interactions, 14–15
  - Configuration Management Yellow Pages Web site, 165
  - Conflicts, about software configuration management, 7
  - Congruent behavior, 9
  - Context, of pattern, 39
  - Continuous integration, 65, 69, 70
  - Conway’s Law, 24
  - Coordination, 30
  - Coplien, James O., 102, 164
  - Copy-modify-merge model, with CVS, 175
  - CORBA IDL files, 27
  - Corporate politics, 23
  - Course-grained tasks, 113
  - CppUnit, 131, 132
  - Culture, and distance, 29
  - Customer releases, branching for, 56
  - CVS, 53, 57, 90, 122
  - CVS-Concurrent Versions System, 175–177
  - “CVS II: Parallelizing Software Development” (Berliner), 110
- D**
- Daemons, with Perforce, 178
  - Daily Build and Smoke Test* pattern, 102, 126, 164
  - Dart, Susan, 167, 194
  - Databases, CM Synergy, 187
  - Data files, for workspace, 81
  - Data migration, 148
  - Day-to-day software development, 8
  - Dead-end code, 151
  - Deadlock, 60
  - Debugging, 74, 89, 90, 92, 104, 138
  - Decisions, and solutions, 143
  - Decoupling, 27



## 210 Index

---

- Defects
    - fixing, 136
    - reports, 114
    - and smoke tests, 125
  - Deintegration build, 100
  - Delayed integration, 69
  - Delays, 61
  - Deliver, with ClearCase UCM, 186
  - DeMarco, Tom, 69
  - Dependencies
    - building, 74
    - and private system build, 91
  - Dependent branches, staircase of, 150
  - Deployment view, 26
  - Depot, in Perforce, 177
  - Derived objects, 108
  - Design, object-oriented, 36
  - Design Patterns* (Gamma et al.), xix, 36, 46
  - Design view, 27
  - Developer branches, 161
  - Developer builds, 89
  - Developers
    - and codelines, 118–120
    - and private versioning, 145
    - regression tests run by, 139
    - smoke tests run by, 126
  - Developer's workspaces, 76
    - and checkpointing changes, 144, 145
    - creating, 82, 109
  - Development, glacial, 5–6
  - Development codeline, policy for, 121
  - Development environment
    - integration between version control system and, 115
    - patterns in, 34
  - Development paths, in SI, 192
  - Development support, with software configuration management, 13
  - Development team, and integration issues, 83
  - Development workspace, 25
  - Dikel, David, 31, 64
  - Directories
    - source code partitioned into, 27
    - versioning with ClearCase, 184
  - Distance, and the organization, 28–29
  - Downey, Grace, 167
  - Dynamically loaded third-party components, installing, 109–110
  - Dynamic view, with ClearCase, 185
- E**
- Easterbrook, Steve, Configuration Management Resource Guide by, 167
  - Eaton, Dave, 167
  - Edit policy, 115
  - Embedded interpreters, 110
  - Empty workspaces, 94, 95
  - Enhancements, 149, 150, 151
    - and release engineering branch, 156
    - requests for, 148
  - Errors, 6, 64, 77, 125
    - build, 90, 95, 98, 101, 102
    - fixing, 151
  - Essential SourceSafe* (Roche and Whipple), 193
  - Executables, 38
  - Exhaustive testing, 136–137
  - Extreme Programming, xvii, 21
    - and continuous integration, 52, 70
    - unit testing and, 133
- F**
- Failure modes, 137, 138
  - Feiler, Peter, 167
  - Files
    - branches of, xxix
    - branching and merging with trunk, xxx
  - Fisher, Roger, 17
  - Flinders University, SEWEB Software Configuration Management Resources at, 168
  - Flow, defining, 69
  - Fogel, Karl Franz, xxxiv, 193
  - Folder/file hierarchies, with StarTeam, 188
  - Fowler, Martin, 65, 131
  - Freezing, branching instead of, 155
  - Full builds, 93, 95
  - Fully replicated peer-to-peer model, in BitKeeper, 179
  - Functionality, verifying, 125–126
  - Future tasks, 159



**G**

Gamma, Erich, xix  
 Garlan, David, 17  
 gcc, 109  
 “Generative Development Process Pattern Language, A” (Coplien), 164  
 get operation, in VSS, 174, 175  
*Getting Past No* (Fisher), 17  
*Getting to Yes* (Fisher), 17  
 Glacial development, 5–6  
 Global changes, 145  
 Global history, 142  
 Global stability, 158  
 Goals, defining, 63–66  
 Goetze, Christian, 193  
 Goldfedder, Brandon, 46  
 Google, 169  
 Granularity, and checkpointing changes, 144, 145  
 “Green fields” development project, 25  
 Grinter, Rebecca, 26  
*Guide to Software Configuration Management* (Leon), 194

**H**

Half Private Office pattern, 36  
 Header files, 27, 108  
 Helm, Richard, xix  
 “High-Level Best Practices in Software Configuration Management” (Wingerd and Seiwald), xxxiii  
 Highsmith, Jim, 7, 17, 66  
 Hillside Group, xix, 46  
 Hoek, Andre van der, 165  
 Hunt, Andrew, 31, 84, 91  
 IIIDE. *See* Integrated development environment  
 Identification, 30  
 Implementation view, 26  
 Import/export relationships, and codeline, 121  
 Improvements, 136  
 Include/exclude approach, 93  
 Incremental builds, 77, 93, 95  
 Installation kits, 102, 106  
 Installation scripts, for workspaces, 81  
 Institute for Configuration Management, 167  
 Institute for Information Technology, 168

Integrated development environment, xxviii, 80, 92, 173

**Integration**

branches, 56  
 difficulty with, 99  
 policy, 30  
 tests, 61, 130  
 workspace, 76  
 Integration build, 88, 94, 100  
 intent of, 102  
 pieces assembled by, 101  
 reports, 112  
 scripts, 92

*Integration Build* pattern, 45, 77, 80, 91, 94, 95, 97–102, 112, 124

Interface definition files, 27

Interface files, 108

Interpreted languages, 110

“Introducing Patterns into Organizations” (Manns and Rising), 31

Isolation, branches used for, 160–162

**J**

jar files, 80, 93

*Java Tools for Extreme Programming: Mastering Open Source Tools Including Ant, JUnit, and Cactus*, 86

Jeffries, Ron, 133

Johnson, Ralph, xix

*Joy of Patterns, The* (Goldfedder), 46

J2EE software architecture, xiii

JUnit, 132, 133

**K**

Kane, David, 31, 64

Kernighan, Brian W., 31

**L**

Labels, in VSS, 174

Labels, xxix, xxviii, xxxi, 30  
 for releases, 109

“Lazy branching,” 161

Leon, Alexis, 194

Library files, xxviii, 84

Linear development, 150



## 212 Index

---

- Lister, Timothy R., 69
- Local revision control area, 144
- Local traceability, maintaining, 142
- Long-lived parallel efforts, branching for, 56
- Long running tests, mixed value with, 61
- Long-term tasks, handling, 158–162
  
- M**
- Mainline, 148
  - branching releases off from, 151
  - doing all your work on, 149
  - and single product releases, 54
- Mainline* pattern, xiv, 41, 42, 49–57, 60, 74, 75, 173
- Mainline codeline, policy for, 122
- Mainline development, 55, 57
  - advantages with, 150
  - coding done for, 73
  - creating, 56
- Maintenance releases, 156
- “make” tool, 86
- Management software development, 8
- Management support, with software configuration management, 13
- Manager Pool, The: Patterns for Radical Leadership* (Olson and Stimmel), 46
- Manns, Mary Lynn, 31
- Master projects, in VSS, 174
- Master repository, in BitKeeper, 179
- McConnell, Steve, 25, 31, 95, 127, 139
- Members, in SI, 192
- Merant, 189
- “Merge ancestry,” 177
- Merge technology, in version control tools, 177
- Merging/merges, 41, 50, 108
  - automating, xxxi
  - and conflict reconciliation, 76
  - for integrating changes from branch to trunk, xxx
  - messy, 51
  - and releases, 149
  - of task branch into active development line, 162
- Metadata, versioning with AccuRev, 182
- Microsoft Visual Source Safe, 10
- Mikkelsen, Tim, 193
- Modularity, 27
  
- Module architecture, changing, 63
- Modules, 90, 109
  - changes to, 112
  - structure of, 30
  - testing, 130–131
- Molli, Pascal, Web site of, 193
  - CM Bubbles” SCM Resources Page Web site, 168
- Myers, Glen, 133, 139
- Mythical Man-Month* (Brooks), 127
  
- N**
- Named Stable Bases* pattern, 64, 65, 74, 126, 164
- labeling, 65
- Names, codeline, 118, 121
- Nightly builds, 88, 90, 91, 93, 100, 139
- Noncongruent behavior, 9
  
- O**
- Object-oriented design, 36
- Object-oriented systems, 37
- Object Solutions* (Booch), 132
- Open Source Development with CVS* (Fogel and Bar), xxxiv, 57, 193
- Oregon Experiment, The* (Alexander), 45
- Orenstein, Robert, xx
- Organization, 23
  - influence of, 28–29
  - and software development, 28–30
- Organizational structure, 22
  - and architecture, 27
  - and communication paths, 29
  - and version control, 12
- Orthogonalization, 108
- Oshry, Barry, 17
- Outdated code, avoiding, 75
  
- P**
- Parallel change, managing, 69
- Parallel efforts, 50
- Parser-Builder pattern, 26
- “Past, Present and Future of CM, The” (Dart), 167
- “Patches,” 156
- PATH(s), 91, 109
- Pattern Almanac 2000*, 164



- Pattern Language*, A (Alexander), 45
- Pattern languages, 34–36, 40–41
  - SCM, 42
- Pattern Languages of Programs (PLoP) Conferences, xix, xv, xx, 45
- Pattern-Oriented Software Architecture* series (Schmidt et al.; Buschmann et al.), 17, 46
- Patterns, xix, 16
  - Active Development Line*, 59–66
  - Codeline Policy*, 117–122
  - codeline-related, 44
  - configuration management, 37–39
  - Daily Build and Smoke Test*, 164
  - definition of, 34
  - Integration Build*, 97–102
  - Mainline*, 49–57
  - Named Stable Bases*, 164
  - parts of, 39
  - Private System Build*, 87–95
  - Private Versions*, 141–145
  - Private Workspace*, 67–77
  - Regression Test, 135–139
  - Release Line*, 147–151
  - Release-Prep Code Line*, 153–156
  - Repository*, 79–86
  - Smoke Test*, 123–127
  - in software, 36–37
  - Task Branch*, 157–162
- Task Level Commit*, 111–115
- Third Party Codeline*, 103–110
  - tool support for SCM, 171–193
- Unit Test*, 129–133
  - workspace-related, 42, 44, 45
- Patterns Almanac, The* (Rising), 46
- Patterns for Effective Use Cases* (Cockburn), 31
- Patterns Languages of Program Design* series, 46
- Peopleware* (DeMarco and Lister), 69
- Perforce, 122, 177–178
- Periodic builds, 21–22, 25
- Perl, 109, 110
- Persistence mechanism, changing, 159
- Pherigo, Suzanne, 193
- Physical location, and distance, 28
- Picture, for pattern, 39
- Pike, Rob, 31
- Pin operation, 174
- Pipes-and-filters architecture, 26
- Policies, for codelines, xxix, 118, 119, 120–122
- Politics, 22, 23
- Poole, Damon, 193
- Practical Software Configuration Management* (Mikkelsen and Pherigo), 193
- Practice of Programming, The* (Kernighan and Pike), 31
- Pragmatic Programmer, The: From Journeyman to Master* (Hunt and Thomas), 31, 91, 95
- Precheck-in
  - build, 88
  - policy, 114
  - process, 65
  - testing, 112, 127
  - verification, 124
- Pressman, Roger, Web site of SCM links for software engineering by, 168
- Private branches, and ClearCase GUI on Windows platforms, 184
- Private repositories, check-ins redirected to, 145
- Private system build
  - attributes of, 91
  - components of, 92
  - and product build, 92, 93
- Private System Build* pattern, 45, 66, 73, 74, 75, 76, 77, 87–95, 124, 126
- Private Versions* pattern, 43, 76, 141–145
- Private Workspace* pattern, 25, 40, 45, 66–77, 80, 84, 88, 98, 104, 145, 172
- Private workspaces, 72–73, 77
- Problem areas, and regression tests, 138
- Problem reports, 113, 114
- Problems, patterns and solving of, 39
- Processes, 38
- Process management, xxvii, 14
- Process view, 27
- Product architecture, 16, 23
  - and organizational forces, 29
  - and version control, 12
- Product build, and private system build, 92, 93
- Product code, versions of vendor code coordinated with, 104–105

## 214 Index

- Product releases, xxxi
  - Products, and PVCS Dimensions, 189
  - Products module structure, 27
  - Progress, balancing stability and, 4–6
  - Project access, with StarTeam, 188
  - Project checkpoints, in MKS Source Integrity, 192
  - Project database, PVCS, 191
  - Project object, in CM Synergy, 187
  - “Project-oriented” operations, VSS support for, 174
  - Project repositories, stable code sets checked into, 144
  - Project rhythm, 64
  - Projects
    - and PVCS Dimensions, 189
    - ClearCase UCM, 185
    - MKS Source Integrity, 192
  - Pull, in BitKeeper, 179, 181
  - Push, in BitKeeper, 179
  - PyUnit, 132
- Q**
- QA team, and integration issues, 83
  - Quality, 4, 9
    - of codeline, 130
    - and smoke tests, 126
- R**
- Rapid Development* (McConnell), 31, 77, 95, 102, 127
  - Real versions, in AccuRev, 182
  - Rebase, with ClearCase UCM, 186
  - Reconfigure, in CM Synergy, 187
  - Recoverability, 158
  - Redundant efforts, reducing, 150
  - Refactoring, 113, 132, 150, 159
  - Regression, 137
  - Regression Test* pattern, 45, 102, 127, 135–139
  - Regression tests, 64, 115, 132, 138–139
  - Release builds, 89
  - Release codeline, policy for, 121–122
  - Release cycles, 55, 104
  - Release dates, and code freeze, 5
  - Released versions
    - maintenance on, 148–150
    - on release line, 151
  - Release engineering, 89, 90
  - Release engineering branch, 155
  - Release histories, 108
  - Release, in CM Synergy, 187
  - Release Line* pattern, 43, 65, 118, 147–151, 156, 187
  - Release lines, 57, 118, 151, 160
    - branching, 126
    - early creation of, 160
  - Release-Prep Code Line* pattern, 43, 66, 153–156, 173
  - Releases, 20
    - automated regression testing tied to, 139
    - branching off from mainline, 151
    - code freezes instituted before, 155
    - and glacial development, 5–6
    - labeling, 109
    - management of, 104
    - and organizational forces, 29
    - stabilizing work on, 154
  - Release workspaces, 74
  - Repository, and CM Synergy, 187
  - Repository* pattern, 45, 77, 79–86, 104, 172
  - Responsibilities, and organizational structure, 28
  - Review, xxvii, 14
  - Revision control system, 114
  - Revisions, xxviii, xxx
  - Rhythm, 64
  - Rising, Linda, 31
  - Risk, reducing, 160
  - Roche, Ted, 193
- S**
- “Sandboxes,” 77, 192
  - SCM. *See* Software configuration management
  - SCM pattern language, 42
  - SCM tools, functions of, 77
  - Scripting languages, 110
  - Scripts, xxviii, 85
  - Seeing Systems. Unlocking the Mysteries of Organizational Life* (Oshry), 17
  - SEG. *See* Software Engineering Group
  - Seiwald, Christopher, xxxiii, 56, 91
  - Self-scoring smoke tests, 126
  - Semaphores, 62
  - SEWEB Software Configuration Management Resources
    - at Flinders University, Web site for, 168
  - Share operation, in VSS, 174
  - Shaw, Mary, 17

- SI. *See* Source Integrity
- Small-grained check-ins, 115, 124
- Small-grained tasks, one commit per, 113–114
- Smoke Test* pattern, 45, 66, 73, 75, 94, 95, 102, 123–127, 130, 136
- Smoke tests, 115, 125, 127, 130, 132, 164
- Snapshots
  - of codeline, xxix, xxviii, xxxi
  - coding tasks performed against, 72
- Software, patterns in, 36–37
- Software architecture, 25
  - Software Architecture: Organizational Principles and Patterns* (Dikel and Kane), 31
- Software configuration management, xiii, xvii
  - in context, 8–10
  - description of, 13–15
  - key concepts and terminology for, xxvii–xxxi
  - role in agile software development, 7–8
  - within software development, 4
  - as team support discipline, 11–13
- Software Configuration Management: Coordination for Team Productivity* (Babich), xxxiii
- Software Configuration Management FAQ Web site, 167–168
  - Practical Introduction\_\_ (White), 57, 77, 193
- Software configuration resources, on the Web, 165–169
- Software development
  - organizations, 9
  - and synchronization, 60
  - in team environment, 69
- Software Engineering Group, 168
- Software Engineering Institute, SCM publications by, 167
- Software Engineering Resource List for Software Configuration Management, Web site for, 168
- Software engineers, xvii
- Software environment, 20–31
  - general principles, 20–22
  - interactions between elements of, 24
  - structures within, 23
- “Software Reconstruction: Patterns for Reproducing the Build” (Cabrera et al.), xxi
- Software Release Methodology* (Bays), xxxii, xxxiii, 57, 151
- Software systems
  - big picture for, 30–31
  - building, 22–23
- Solutions
  - and decisions, 143
  - and pattern languages, 34
  - and patterns, 39
- Source code, xxviii
  - modules, 80
  - for workspace, 80
- Source control, 11, 104
  - repositories, 145
  - tools, 30
- Source control structure, and rhythm, 64
- Source control system, 61, 101
  - and clean builds, 94
  - and errors in build, 102
- Source files, and *Repository* pattern, 85
- Source Integrity, 192
- Source trees, updating, 73
- “Spectrum of Functionality in CM Systems, The” (Dart), 167
- Speed, 4, 112, 127
- SQL Software, 189
- Stability, 112
  - balancing progress and, 4–6
  - codeline requirements for, 118
  - in software development, 130
- Staged daily builds, 102
- Staircase branching (or a cascade), 52, 53
- Staircase codeline structure, 51
- Staircase of dependent branches, 150
- StarTeam, 188–189
- Status accounting, xxvii
- Status accounting audit, within software configuration management, 13
- “Streamed Lines: Branching Patterns for Parallel Software Development” (Appleton et al.), xxi, xxxii, 151
- Streams
  - Backing streams, in AccuRev, 182
  - Base stream, in AccuRev, 182
  - Development streams, with ClearCase UCM, 185
  - Dynamic streams, in AccuRev, 182
  - Integration streams, with ClearCase UCM, 185
  - Release stream, in CM Synergy, 187
  - Static streams, in AccuRev, 182
  - Workspace streams, in AccuRev, 182
- Subversion, Web site for, 193



## 216 Index

---

- Survival rules, 9
- Symbols, codeline diagram notation, xxxii–xxxiii
- Synchronization points, 60
- System build, and compatibility of changes, 88
- “System for Version Control, A” (Tichy), xxxiii
- System interfaces, stabilizing, 164
- System tests, 131
- T**
- tag command, in CVS, 175, 176
- Task branches, 126, 161, 162
- Task Branch* pattern, 43, 56, 66, 75, 114, 156, 157–162, 172
- Task Level Commit* pattern, 45, 76, 111–115, 172
- Tasks
  - in CM Synergy, 187
  - in StarTeam, 189
- Task workspace, 76
- Tcl, 109
- Teams, xxvii, 23, 40, 69
  - and communication, 21
  - and distance, 28, 29
  - and software configuration management, 14–15
  - structure of, 27
- Testing, 127
  - for changes, 138–139
  - code, 124
  - exhaustive, 136–137
  - modules, 130–131
  - treadmill, 62
- Test suites, 62
- Third-party code, 105, 106
  - branching, 151
  - codeline for, 106–110
- Third party codeline, 108
- Third Party Codeline* pattern, 43, 74, 77, 86, 103–110, 173, 176
- Third-party components, 25
  - and *Repository* pattern, 85
  - tracking, 104
  - for workspace, 80, 84
- Thomas, David, 31, 84, 91
- Tichy, Walter F., xxxiii
- Timeless Way of Building, The* (Alexander), 45
- “TimeSafe Property, The—A Formal Statement of Immutability in CM” (Poole), 193
- Time safety, with AccuRev, 182
- Tip, of codeline, xxviii
- Title, of pattern, 39
- Tools, 22
  - role of, 15–16
  - and version control, 12
- Topics, in StarTeam, 188
- Towns, pattern language for building of, 35, 45
- Traceability, 158
- “Transaction-Oriented CM: A Case Study” (Feiler and Downey), 167
- Triggers, 101, 122
  - in BitKeeper, 179
  - in ClearCase, 184
  - in Perforce, 178
- U**
- “Ubiquitous Automation,” 84
- UCM. *See* Unified Change Management
- UCM Central-Unified Configuration Management Web site, 166
- UML. *See* Unified Modeling Language
- Unified Change Management, 183, 185–186
- Unified Modeling Language, xxxi, 26
- Unified Modeling Language User Guide, The* (Booch et al.), 17
- Unit Test* pattern, 45, 66, 73, 127, 129–133
- Unit tests, 64, 115, 126, 131, 138, 139
  - developing/running, 131–132
  - properties of, 131
- Unresolved issues, patterns and addressing of, 39, 40
- Update
  - in AccuRev, 183
  - in CVS, 175, 176
- Update members, in CM Synergy, 187
- Updates, 82
- Use case view, 26
- Usenet group, 167
- User problem report, 132



**V**

Validated builds, xxxi  
Vance, Stephen, 120  
Variant projects, with PVCS, 191  
Vendor branch, 110, 176  
Vendor code, 106  
    accepting, 107  
    product code versions coordinated with,  
        104–105  
Vendor releases  
    cycles, 104, 105  
    tracking changes in, 108  
Ventimiglia, Bob, 171  
Version control, 4, 6, 7, 10, 11, 12, 17, 20, 26, 93  
    and check-ins, 127  
    and code freeze, 5  
    and communication, 21  
    and identification, 30  
    organization's influence on, 27  
    policies, 24  
    and tools, 15–16  
Version control system, 38, 61, 80, 82, 83, 85, 102,  
    104, 154, 158, 159  
    archiving with, 106  
    integration between development  
        environment and, 115  
    and private versions, 142–144  
    tools with interfaces to, 86  
    traceability in, 109  
    work to do between submissions to, 112  
Version control tools, 50, 75, 84, 155, 160  
    and branching, 53  
    and codeline policy, 121, 122  
    commonly used, 171–172  
    revision history in, 112  
Version control tree, 86, 93  
Version labels, with PVCS, 191  
Versions  
    of codeline, xxix, xxviii  
    and workspace update operations, 76  
Version tree, for workspace, 85  
View profiles, and ClearCase GUI on Windows  
    platforms, 184  
Views, 26, 77, 188

Virtual private repositories, 145  
Virtual versions, in AccuRev, 182  
Visitor pattern, 26  
VisualAge for Java, xiv  
Visual C++, 173  
Vlissides, John, xix  
VSS, branching in, 53  
VSS-Visual Source Safe, 173–175

**W**

Web. *See* World Wide Web  
WebSphere Studio (IBM), xiv  
Weinberg, Gerald, 8, 9, 17  
Whipple, Larry C., 193  
White, Brian, 77, 193  
Wingerd, Laura, xxxiii, 56, 91  
Work  
    autonomous, 22  
    policies influencing, 23–25  
Work areas, in CM Synergy, 187  
Workfile location, and PVCS, 191  
Working directory, in VSS, 174  
Working files, with StarTeam, 188  
Working folders, with StarTeam, 188  
Working projects, in CM Synergy, 187  
Work packages, in PVCS Dimensions, 189  
Worksets, and PVCS Dimensions, 189  
Workspace-related patterns, 42, 44, 45  
Workspaces, 23, 25  
    AccuRev, 182  
    BitKeeper, 179, 180  
    ClearCase support for, 77  
    CM Synergy, 187  
    defined, xxvii–xxviii  
    empty, 94, 95  
    items used in creating, 81  
    management of, 30  
    multiple, 74  
    Perforce, 178  
    populating from repository, 84  
    population of, from different codelines,  
        xxx  
    private, 72–73, 77  
    PVCS, 191



## 218 Index

---

rebuilding in, 93  
right versions of right components into, 80–82  
sharing components between, 71  
and smoke tests, 126  
structure of, 27  
updating, 75, 76  
version tree for, 85

Work styles, 10  
World Wide Web, SCM resources on, 165–169  
Writing test cases, 132

### **X**

XP. *See* Extreme Programming

