Index

adaptation, 134
adaptive software development, 193
Adobe's Eve, 114
agility, 41, 211
capability maturity model and, 195
consistency vs., 170
continual refinement and, 134
design and, 110
development methods for, 193
Extreme Programming and, 183
manifesto on, 172
AND/OR thinking, 10
Apple
  Shark, 87
  war rooms, 169
Apple iPod, 24
architecture
  identifying battles over, 167
  rearchitecture and, 123
  record and playback, 84
  user visibility of, 116
barely sufficient documentation, 53
Boundschecker, 96
bug tracking tools, 75
bugzilla, 52
builds
  metrics for, 148
  nightly, 56
  release planning and, 139
time taken by, 57
timestamping, 59
bureaucracy, 30
burnout, 9
business models
disruptive, 19
business plans, 18
capability
  working harder vs. working smarter and, 3
capability maturity model (CMM), 195
ceremony, 100
discipline vs., 36
change, 12
  agents, 151
  aversion to, 13
  continual refinement and, 133
  cultural, 151
designing for, 107, 111
disruptive technologies and, 18
effecting, 152
enablers of, 158
examples of, 170
factors for, 156
jumping in place and, 16
need for, 212
phases of, 165
reactions to, 156
resistance to, 154
tolerance of, 39
transition plans and, 162
chemical manufacturing, sustainable
development in, 3, 5
coaching, 144
Cobol, 11
code
  copy and paste porting, 69
code (cont.)
documentation of source, 95
metrics for, 150
ownership of, 98
platform dependencies and, 69
reviews, 98
source code analyzers and, 93
code comments, 54
code-aware tests, 80
code-then-fix mentality, 2, 28, 211
defect prevention and, 73
working product and, 42
coding standards and guidelines, 65
collaboration, 38
as project control, 21
cross-functional, 137
design and, 110
in Extreme Programming, 186
in software design, 55
iterative development and, 137
positive reinforcement of, 161
with users, 168
Collins, Jim, 15, 40
command-control, 152
commodities, 67
communication, 119, 161
competition, 16, 18
compilers, defect prevention and, 92
complacency, 154
complementarity of talents/skills, 39
complexity, 12
componentized software, 130
configuration management, 57, 96
consistency, 170
constructor methods, 93
consultants, 160
continual improvement, 7, 9, 11
continuous integration, 56
copy and paste porting, 69, 93
cornerstone tests, 91
cost management, 20
CPD, 93
Creativity. See also innovation, 195
for retrospectives, 142
in automated testing, 81
culture, organizational, 23, 36
changing, 151, 212
defect detection oriented, 75
do business with, 169
of software development, 28
work environment in, 169
customers
collaboration with, 168
contact with, 147
defect reporting by, 44
disruptive technologies and, 18
feedback from, 29, 31
in Extreme Programming, 187
mindshare of, 16
on-site, 190
positive reinforcement of contact with, 161
understanding, 145
value to, 3
daily shipping, 43
data structures, 63
debugging
milestones and, 47
working product and, 42
decision making, 23
defects
avoiding backlogs of, 51
backlogs of, 75
build times and, 57
cost of fixing and time of discovery, 73
designing for reuse and, 128
detection of and cost of change, 13
detection vs. prevention of, 8
metrics for, 148, 167
milestones and, 47
no broken windows practice
and, 49
practices for preventing, 179
pragmatic practices for, 52
preventing, 73, 212
preventing from reaching
customers, 42
preventing vs. detecting, 31,
33, 73
prioritizing, 101
programming and code reviews
and, 98
quality assurance and, 73, 76
rearchitecture and, 123
root-cause analysis and, 100
ruthless testing for, 78
time spent fixing, 13
tools for, 92
tracking, 52
uncompromising attitude
towards, 50
won’t fix, 101
working product and, 42
Dell Computers, 19
design, 107, 213
bottom-up, 108
discussions, 122
emphasis on, 33
extreme, 108
Extreme Programming and, 192
for reuse, 128
for testability, 84
frequent meetings in, 122
guiding principles in, 112–113
patterns in, 119
practices for, 180
rearchitecture and, 123
refactoring and, 117
reviews, 122
simplicity in, 115
top-down, 108
vision in, 112
deterministic programs, 85
discipline, 36, 142
Extreme Programming and, 193
disruptive technologies, 18
diversity, 153
documentation
barely sufficient, 53
code reviews and, 100
dangers of excessive, 55
design and, 110
in plan-driven development, 22
source code, 95
updating, 44
doxygen, 55
dynamic systems development
method (DSDM), 194
Eclipse, 45
education, 154, 211
emergent design, 108. See also
design, 195
engineering approach to software
development, 27
engineering, guiding principles
for, 112–113
errors, defect prevention and, 93
Eve, 114
event logging, 95
evolutionary design, 108. See also
design, 195
exception handling, 93
execution profilers, 94
exit criteria, 53
expertise, 22
external dependencies, 18
Extreme Programming
design in, 192
misconceptions of, 192
practices for, 184
strengths of, 185
Extreme Programming (cont.)
values of, 183
weaknesses of, 187
Extreme Programming (Beck), 108
Extreme Programming (XP), 183

feature cards, 55, 136
feature-driven development, 193
features
AND/OR thinking about, 10
continual improvement and, 7, 9, 11
documenting, 54
metrics for, 148
non-feature work and, 8
planning development of, 44
simplicity and reliability and, 23
time spent developing, 13
vision and, 29
feedback
from users, 29, 31
in continual refinement, 142
iterative development and, 135
retrospectives and, 142
flexibility, 41
flywheel metaphor, 15
fun, 40
functionality freezes, 47
game engines, 85
global variables, 93
Good to Great (Collins), 40
guiding councils, 174
hardware, 63
Highsmith, Jim, 109, 172
hotkey managers, 116
Hunt, Andrew, 49
IBM, 67
improvement
continual, 8, 39

working harder vs. working smarter and, 6
innovation
bureaucracy and, 30
pace of, 1
integration
continuous, 100
tests for, 81, 85
integration, continuous, 56
interfaces
designing for reuse and, 128
external dependencies and, 18
integration testing, 81, 85
internationalization of products, 68
Internet
as disruptive technology, 19
iPod, 24
ISO-9000, 21
iterations
bug fix-only, 53
design reviews and, 122
iterative development, 135, 213
Java, PMD for, 93
JavaDoc, 55
JavaDocs, 95
jumping in place, 16
Kerievsky, Joshua, 109
keystone practices, 177. See also practices, 195
leadership, 23
change and, 156
culture of, 37
Extreme Programming and, 191
lean development, 193
Lean Software Development (Poppendieck), 43
learning, continuous, 29, 39
libraries, integration tests and, 81
Linux, 20
logging capability, 84
logging, event, 95
logic errors, 99
management
attitude leadership and, 8
change and, 152
parking lot managers in, 5
project, 171
support for change and, 158
manipulation, 152
market share, 16
markets, understanding, 146
meetings
daily standup, 141
design, 122
status report, 141
memory leaks, 63
methodology, 21
metrics, 148
for defects, 167
Microsoft
business model of, 20
daily shipping at, 46
Office97, 23
Windows, testing and, 88
milestones, artificial, 47
mindset
continual improvement and, 8
for sustainable development, 2
software development culture and, 27
mock objects, 81
model-view-controller pattern, 86
morale
working harder vs. working smarter and, 4
multi-threading, 124
nested statements, 93
nightly builds, 56
no broken windows, 49
nondeterministic programs, 85
Norton Anti-Virus, 46
object-oriented programming, 130
objects, mock, 81
Office97, 23
open source software, 20, 45
defect prevention and, 97
fixes to, 67
Osborne Computer, Inc., 16
Osborne effect, 16
outsourcing, 146
overdesigning, 2, 108, 129
pair programming, 189
Palm PDA, 24
parking lot managers, 5
performance, 62
metrics, 148
testing, 87
persistance, 158
pillars of design, 112–113
plan-driven development, 21
release planning in, 139
sloppy planning and, 134
tradition methods and, 144
transition plans in, 162
platform dependencies, 69
rearchitecture and, 124
PMD, 93
positive reinforcement, 160
practices, 28, 177
attention to performance, 62
barely sufficient
documentation, 53
coding standards/guidelines, 65
continual refinement of, 31
continuous integration, 56
practices (cont.)
defect prevention, 78
  for continual refinement, 135
  for Extreme Programming, 184
  internationalization of
    products, 68
  nightly builds, 56
  no broken windows, 49
  principles vs., 28
  prototyping, 59
  standards adoption, 67
  uncompromising attitude toward
defects, 50
  zero tolerance for
    memory/resource leaks, 63
Pragmatic Programmer (Hunt,
  Thomas), 49
principles, practices vs., 28
product
  continual refinement of, 32
  working, at all times, 31–32
product development, 29
product differentiation, 33
professional development, 148
project controls, 21
project management, 189
  training for, 171
project stresses, 17
prototyping, 59
  iterative development and, 139
Purify, 96
Qt, 45
quality assurance (QA)
  defect prevention vs. detection
    and, 33
  importance of prompt, 43
    in defect prevention, 73, 76
  reactive thinking, 134
  record and playback, 84
refactoring, 117
  group exercises in, 168
  supporting, 161
  refinement, continual, 133, 213
    coaching/team development
      in, 144
    daily standup meetings in, 141
    iterative development in, 135
    metrics in, 148
    practices for, 181
    release planning in, 139
    retrospectives in, 142
regression tests, 88
regressions, 51, 75
reinforcement, positive, 160
reinvention of companies, 3
releases
  death spiral and, 14
  defect backlogs and, 75
  iterative development and, 135
  planning, 139
  reliability, 23
  Renerware, 46
  Repenning, Nelson P., 4
  replaceability, 129
  research projects, iterative
dev development for, 137
  resource leaks, 63, 88
    detecting, 95
  resource-usage tests, 88
  responsibility, 37
  retrospectives, 142
  reuse, designing for, 128
  ripple effects, 44
  risk
    awareness of, 39
    prototyping and, 59
  roadmaps, 124
  root-cause analysis, 100
  rules, 28
  run-time monitoring, 87
scalability, 189
scheduling
  artificial milestones and, 47
self-organization, 191
Shark, 87
shipping
  at unpredictable times, 47
daily, 45
simplicity, 23
  in design, 115
SketchBook Pro, 55
sketching tools, 55
skills, complementary, 39
software development. See also
design, 195
  agile methods for, 193
capability maturity model
  and, 195
culture of, 28
death spiral in, 13
engineering approach to, 27
Extreme Programming in, 183
iterative, 135, 213
manifesto for agile, 172
methodology of, 21
plan-driven development, 21
waterfall approach to, 27
Software Engineering
  Institute, 195
software ICs, 130
source code analyzers, 93
standards
  adopting, 67
  for coding, 65
standup meetings, 141
Sterman, John D., 4
sustainable development, 212
  advantages of, 3
  applying principles of, 31
  as juggling, 34
  beginning, 163
capability maturity model
  and, 195
chemical manufacturing example
  of, 3, 5
Cobol and, 11
culture change and, 151
definition of, 1
design in, 107
example of, 10
feedback and, 31
for new projects, 166
mindset behind, 2
practices and, 29
principles of, 27
vision and, 29, 31
switch statements, 93
system tests, 83
talents, complementary, 39
teams
  developing/coaching, 144
  new projects and, 166
  proactive, 2, 212
  self-organization of, 191
technical debt, 15, 77
testing, 44. See also quality assurance
  (QA), 44, 195
  automated, 81–82, 87, 90
  cornerstone, 91
  cost of, 74
  equipment for, 169
  in defect prevention, 78
  integration, 81, 85
  performance, 87
  positive reinforcement of, 161
  rearchitecture and, 127
  record and playback, 84
  regression, 88
  resource-usage, 88
  resources for, 90
  system, 83
testing (cont.)
  test coverage analysis and, 94
  usability, 89
  user verification, 89
  visual validation, 86
Thomas, David, 49
throwaway prototypes, 61. See also
  prototyping, 59
timestamping builds, 59
Torque, 46
tracer bullets, 59, 62
training, 168
  change and, 159
  consultants and, 159
  project management, 171
transition plans, 162
UML diagrams, 112
underdesign, 2, 108
unit tests, 80
unsustainable development, 13–14,
  16, 18, 20, 22, 24, 212
  causes of, 17, 19, 21, 23
  flywheel metaphor of, 15
  jumping in place and, 16
  technical debt and, 15
  turning into sustainable, 163
urgency, shared sense of, 38, 157
  consultants and, 160
usability tests, 89
user needs, 29
  guiding design principles for, 113
user requirements, 17
user verification tests, 89
validation testing, 44
value-add, 67
velocity, 138, 149
vicious cycles
  working harder vs. working
    smarter, 3
virtual shipability, 41
virtuous cycles
  working harder vs. working
    smarter, 6
vision-oriented approach, 29
  design in, 112
  iterative development
    and, 139
  organizational culture and, 38
visual validation tests, 86
war rooms, 169
waterfall approach, 27, 211
whiteboards, 55, 112, 169
wins, continual, 160
work environment, 169
  40-hour week and, 190
working product, 31–32, 41, 212
  internationalization of, 68
  practices for, 178
  vs. working software, 41
working smarter vs. working
  harder, 3
Extreme Programming
  and, 190
XML, 68
zero tolerance, 63
Register Your Book at www.awprofessional.com/register

You may be eligible to receive:

- Advance notice of forthcoming editions of the book
- Related book recommendations
- Chapter excerpts and supplements of forthcoming titles
- Information about special contests and promotions throughout the year
- Notices and reminders about author appearances, tradeshows, and online chats with special guests

Contact us

If you are interested in writing a book or reviewing manuscripts prior to publication, please write to us at:

Editorial Department
Addison-Wesley Professional
75 Arlington Street, Suite 300
Boston, MA 02116 USA
Email: AWPro@aw.com

Visit us on the Web: http://www.awprofessional.com
Instill Collaboration in Your Projects with

**Collaboration Explained: Facilitation Skills for Collaborative Leaders**

by Jean Tabaka

0-321-26877-6

Designed for XP coaches, Scrum Masters, Crystal Clear practitioners, and all agile project managers, *Collaboration Explained* reviews the fundamentals of collaboration:

- Primary collaboration roles and events
- Collaboration techniques
- What to do for distributed, virtual teams
- Guidelines for conducting key collaboration activities in your projects

Jean Tabaka has an extensive background as a programmer, project manager, and most recently methodologist, specializing in agile and adaptive software development environments for internal IT departments, ISVs, and consulting organizations. She has created and implemented both rigorous and agile methodologies for Qwest Communications, Siebel Systems, Sybase, Inc., and several smaller firms. She is currently the lead Agile Mentor at Rally Software Development, guiding customers in adopting agile principles and practices and has been working in and writing about collaborative approaches in software projects for more than four years.

With this book, Tabaka presents the collaboration techniques she has developed as "how-to" steps and applies them across a variety of agile methodologies focusing on the application of practices for agile teams, customers, and testers.

Straightforward, practical, and simple, *Collaboration Explained* cuts through the fluff and gives you what you need to effectively put collaboration into practice.

For more information on this title please visit www.awprofessional.com/title/0321268776.
YOUR GUIDE TO IT REFERENCE

Articles
Keep your edge with thousands of free articles, in-depth features, interviews, and IT reference recommendations – all written by experts you know and trust.

Online Books
Answers in an instant from InformIT Online Book’s 600+ fully searchable online books. For a limited time, you can get your first 14 days free.

Catalog
Review online sample chapters, author biographies and customer rankings and choose exactly the right book from a selection of over 5,000 titles.
Wouldn’t it be great if the world’s leading technical publishers joined forces to deliver their best tech books in a common digital reference platform?

They have. Introducing InformIT Online Books powered by Safari.

- Specific answers to specific questions. InformIT Online Books’ powerful search engine gives you relevance-ranked results in a matter of seconds.

- Immediate results. With InformIT Online Books, you can select the book you want and view the chapter or section you need immediately.

- Cut, paste and annotate. Paste code to save time and eliminate typographical errors. Make notes on the material you find useful and choose whether or not to share them with your work group.

- Customized for your enterprise. Customize a library for you, your department or your entire organization. You only pay for what you need.

Get your first 14 days FREE! For a limited time, InformIT Online Books is offering its members a 10 book subscription risk-free for 14 days. Visit http://www.informit.com/onlinebooks for details.