## Index*

<table>
<thead>
<tr>
<th>Term</th>
<th>Page Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>.net</td>
<td>9</td>
</tr>
<tr>
<td><strong>201 Principles of Software Development</strong> by Davis</td>
<td>25, 41, 47, 241</td>
</tr>
<tr>
<td>A Practitioner’s Handbook of Requirements Engineering Methods by Bickerton</td>
<td>153</td>
</tr>
<tr>
<td>about the author, xv–xvi</td>
<td></td>
</tr>
<tr>
<td>Abreo, L. Renee, 166, 177</td>
<td></td>
</tr>
<tr>
<td>academe. See also education and researchers</td>
<td></td>
</tr>
<tr>
<td>career in industry vs., 187–90</td>
<td></td>
</tr>
<tr>
<td>freedom, 188</td>
<td></td>
</tr>
<tr>
<td>how to use this book in, xii</td>
<td></td>
</tr>
<tr>
<td>impact, 189</td>
<td></td>
</tr>
<tr>
<td>job security, 188</td>
<td></td>
</tr>
<tr>
<td>personality traits, 190</td>
<td></td>
</tr>
<tr>
<td>politics, 187, 189</td>
<td></td>
</tr>
<tr>
<td>tenure, 188–89, 190, 204</td>
<td></td>
</tr>
<tr>
<td>Achieving Quality in Software Requirements</td>
<td>114, 143–54</td>
</tr>
<tr>
<td>seeds for debate, 153–54</td>
<td></td>
</tr>
<tr>
<td>acquisitions, 93–94, 97, 103–4, 105, 111, 112</td>
<td></td>
</tr>
<tr>
<td>Ada (language), 43–44, 129</td>
<td></td>
</tr>
<tr>
<td>good programmer?, 116</td>
<td></td>
</tr>
<tr>
<td>Africa, 1, 219, 245–48</td>
<td></td>
</tr>
<tr>
<td>aggregation, 134–35</td>
<td></td>
</tr>
<tr>
<td>agile development, 21, 23, 171, 176, 178, 253</td>
<td></td>
</tr>
<tr>
<td>Agile Software Development by Cockburn</td>
<td>25</td>
</tr>
<tr>
<td>Agilent</td>
<td>89</td>
</tr>
<tr>
<td>Albrecht, Allan, 50, 61</td>
<td></td>
</tr>
<tr>
<td>Alford, Mack, 20, 25</td>
<td></td>
</tr>
<tr>
<td>An Introduction to Software Architecture by Shaw and Garlan</td>
<td>141, 243</td>
</tr>
<tr>
<td>analysis of problem, 42–43</td>
<td></td>
</tr>
<tr>
<td>analysts apprenticeship, 170</td>
<td></td>
</tr>
<tr>
<td>experience effects approach, 163</td>
<td></td>
</tr>
<tr>
<td>reasons for poor performance, 164</td>
<td></td>
</tr>
<tr>
<td>role, 28–29</td>
<td></td>
</tr>
<tr>
<td>Anatomy of a Software Start-Up, 63, 99–106</td>
<td></td>
</tr>
<tr>
<td>epilogue, 105–6</td>
<td></td>
</tr>
<tr>
<td>seeds for debate, 106</td>
<td></td>
</tr>
<tr>
<td>Anderson, Hans Christian, 221</td>
<td></td>
</tr>
<tr>
<td>Andriole, Stephen, 138, 140, 145, 152, 241, 243</td>
<td></td>
</tr>
<tr>
<td>angels source of financing, 90–91, 97, 106, 111</td>
<td></td>
</tr>
<tr>
<td>Anton, Annie, 21, 25</td>
<td></td>
</tr>
<tr>
<td>applications complexity, 1</td>
<td></td>
</tr>
<tr>
<td>Applied Software Measurement by Jones, 60</td>
<td></td>
</tr>
<tr>
<td>apprenticeship analyst, 170</td>
<td></td>
</tr>
<tr>
<td>architecture (software), 12, 138, 241–42</td>
<td></td>
</tr>
<tr>
<td>adaptable, 204</td>
<td></td>
</tr>
</tbody>
</table>

*Bold entries refer to chapter titles; italicized entries refer to book titles.

Great Software Debates. By Alan M. Davis
architecture (software) *(continued)*
- evaluating alternatives, 43, 250
- how it should be selected, 241
- how performed today, 241
- object-orientation and, 142
- optimal, 130

*Are Your Lights On?* by Gause and Weinberg, 42, 141, 175, 205, 243

art
- definition, 27

*Art or Engineering, One More Time, 2, 27–30*
- seeds for debate, 29–30

Arthur, Lowell, 146, 152

assembler language, 18, 125
- good programmer?, 116

Atwood, Bill, 101

Automatic Test Executor, 192, 193, 196

Avot, Pirkei, 225

Bacon, Sir Francis, 226

balance sheet, 79, 112
- baseline agreement, 184

*Basics of Software Engineering Experimentation* by Juristo and Moreno, 39

Basili, Vic, 38

bathrooms
- offices without, 94–95

Baugh, Kerry, 140

Beck, Kent, 21, 25, 166, 171, 176, 177

Beethoven, Ludwig van, 229–34

Bennis, Warren, 107

Bersoff, Edward, 63, 89, 99–106, 111, 112
- best practices, 143, 210–11

*Between Scylla and Charybdis*, 185, 187–90
- epilogue, 189
- seeds for debate, 190

BHAG, 52, 62
- definition, 52

Bickerton, Matthew, 143, 153

bi-directional propagation, 84
- big hairy audacious goal. See BHAG

Billups, Jay, 83, 86

Binda, Belinda, 227

blacksmiths
- compared to software developers, 15–16

Bloody Mary, 226

Blum, Bruce, 208

board of advisors, 90, 96

board of directors, 55, 89, 96
- qualities, 89

Boasson, Maarten, 27, 30

Boehm, Barry, 2, 38, 52, 61

Bohlen, Kemp, 83, 89

Bollinger, Terry, 21, 25, 30

Booch, Grady, 19, 20, 25, 129, 132, 139, 140, 143, 153, 164, 166, 175

Borland, Inc., 156

Bosworth, Michael, 157, 158, 161

Botswana, 245

Boynton, Andrew, 208

brainstorming. See also collaborative session
- when to apply, 157

Bray, Ian, 164, 166, 176

breakeven, 84, 127, 144

Brooks, Fred, 70, 164, 175, 210

Brown, Charlie, 228

BTG, Inc., 89, 99–106, 112
- acquisitions, 103–4
- beginning, 100–101
- business plan, 105
- facilities, 102
- financing, 102
- Omni-Vista vs., 106

Budgen, David, 139, 140

bug
- history of, 225

building construction
- compared to software development, 1, 27–28, 30

*Built to Last* by Collins and Porras, 13

Burns, I., 20, 25

Bushman, 245–48

business
- case, 159–60
- plan, 90–91
- length, 90–91, 97
- rules, 170

Byrd, Terry, 164, 176

C (language)
- good programmer?, 116
- lemming path, 4, 7
- programmers, 7

C# (language), 9, 251, 253
INDEX  

C++ (language), 212  
Caliber RM, 156  
**Can You Survive Your Management Mistakes?**, 63, 69–71  
seeds for debate, 71  
capability maturity model, 12, 20–21, 210, 249  
backlash, 20–21, 23  
Capen, Quinn, 86  
Card, David, 32  
careers  
academe vs. industry, 187–89  
industry vs. academe, 187–89  
caring, 116  
carry the water, 68  
CASE tools, 45, 46, 202, 211, 212, 249  
lemming path, 4, 8  
relationship to word processors, 8  
cash flow statement, 79, 112  
change  
agent, xi–xii  
control, 179  
pocket, 183–84  
requests  
as requirements source, 157  
Chen, Peter, 170, 177  
Cherry, George, 133, 140  
Cheshire Cat, 99, 208  
Chesterton, Gilbert, 222  
CHILL (language), 69  
Clark, Frank, 228  
class  
definition, 129  
cleanroom engineering, 253  
CMM. See capability maturity model  
Coad, Peter, 25, 132, 139, 140, 143, 153  
Cobol, 249  
Cockburn, Alistair, 20, 21, 25  
COCOMO, 52, 58, 61  
cohesion, 46  
collaborative session, 145, 165, 172, 173, 204  
definition, 169  
to catch missed opportunities, 169  
when to apply, 146, 169  
Collins, James, 13, 52, 62, 227  
comments  
role in assembler language, 18  
commercial off-the-shelf software. See COTS  
compassion, 116  
competition, 51, 127, 150  
complete, 130  
complexity  
growth of, 210  
Computer Support for Co-operative Work  
by Spurr, et al., 158  
computer-assisted cooperative work, 158  
concurrent engineering, 209–10  
conflict resolution, 169, 204  
Conger, Sue, 166, 176  
conservative, 1  
consistent, 130, 160  
Constantine, Larry, 46, 139, 140, 166  
Corbin, Juliet, 168, 172, 177  
corporations  
closing, 81  
why they exist, 31–34  
correct, 160  
cost estimation. See estimation  
COTS (commercial off-the-shelf software), 9  
lemming path, 4, 9  
Couger, Daniel, 145, 153, 164, 176  
coupling, 46  
Covey, Stephen, 227, 250  
Cox, Brad, 129, 133, 140  
Creative and Innovation in Information Systems by Couger, 153  
creditors, 81  
Crossing the Chasm by Moore, 153, 175  
CS90, 208–9, 213  
cubicles  
as offices, 236  
cumulative probability graphs. See software estimation: cumulative probability graphs  
Customer-Centered Products by Hooks and Farry, 177  
customers  
caring about, 116  
complaints, 31–32  
delivery date, 51, 69  
demand, 126  
dissatisfied, 120  
distrust for development, 202  
do they know what they need?, 202  
extisting customers as requirements source, 157
customers (continued)
feedback, 95
involvement, 42
needs, 19, 28–29, 46, 59, 180, 187–98, 202
needs from customer vs. supplier perspectives, 187–98
needs in flux, 180, 202
of customers as requirements source, 157
pain, 88
potential customers as requirements source, 157
priorities, 46
reporting progress to, 70
satisfaction, 126
cost, 126, 150
cost for marketing, 150
distrust for marketing, 126, 179, 202
delivery
ready for, 11–12
delivery date
how created, 49, 51, 56
role of belief, 51–52, 58
selected by marketing, 51, 56
DeMarco, Tom, 19, 25, 38, 49, 61, 116, 138, 141, 164, 165, 166, 175, 219, 226, 227, 235–37, 250
design (of software), 125, 138, 139. See also system: design
documentation role, 46
evaluating alternatives, 43
for change, 46
how optimized, 138
transitioning from requirements, 6, 129–42
trends, 250–51
Design Patterns by Gamma, 250
designers (of software). See also developers role, 28–29, 120
Designing Object-Oriented Software by Wirfs-Brock, 141
developers (software)
adding to a late project, 70
disconnect with others, 209
morale, 209
responsibility to society, 235–37
stress, 209
development (software)
art or engineering?, 27–28
as a machine, 77–79
compromising with marketing, 59–60
compromise with marketing, 150
cost, 126, 150
distrust for marketing, 126, 179, 202
history of, 17–21
is right, 61
lying to management, 209
politics and, 17
purpose, 31–34
religion and, 17
risk, 126, 150
role in requirements sign-off, 182
DFD. See data flow diagram
Diana, Princess, 235, 236
Dickenson, Emily, 31
Digital Divide by Norris, 248
Dijkstra, Edsger, 44
Doctors Without Borders, 235
documentation
importance of, 46
Dorffman, Merlin, 143, 153, 160
Dörner, Dietrich, 11, 13, 144, 153, 228
dot-com’s, 9
Duvall, Lorraine, 74
dynamic binding, 135
earned value, 71
*Earned Value Project Management* by Fleming and Koppelman, 71
economic downturn, 85
Edison, Thomas, 225
education (for software), 215–17
curriculum, 216
faculty, 216
flaws, 205
internship, 216
laboratory, 216
modeled after a medical school, 215–17
new model of, 185
role of teacher, 223, 225
silos, 77
trends, 252
*Effective Requirements Practices* by Young, 177, 183
Eiffel (language), 129
Einstein, Albert, 223
elevators, 42–43, 120, 134–35
elicitation, 20, 121–22, 144–46, 147, 157–58, 163–78
definition, 155, 157
iterative process, 165
level of detail, 155
one size fits all, 164
result, 158
state of the practice, 165
technique selection, 121, 163
techniques, 157–58
to requirements specification, 133
tools, 156, 158
*Elicitation: How Do the Experts Do It?*, 114, 163–78
seeds for debate, 177–78
Embley, David, 132, 136, 141
emperor has no clothes, 221
empiricism, 2, 222
importance, 39
in technology transfer, 37–38
used for current fads, 39
employees
communicating with, 96
evaluating, 12–13
firing, 65–68
hiring the best, 42, 45
terminating, 65–68, 87
understanding, 67–68
encapsulation, 46, 129, 131
engineering
definition, 27
education, 127
entity relationship diagram, 131–32, 170
entrepreneur, 99, 109
entropy, 46
Epictetus, 221, 222, 223
ER. See entity-relationship
*Eras of Software Technology Transfer*, 2, 37–39, 243
seeds for debate, 39
error analysis, 46
estimation (of software cost and schedule), 2, 11, 49–61, 90
cumulative probability graph, 56, 58
game playing, 49, 50–51, 56
magic measures, 55
poor record, 49
problems, 49
role of belief, 51–52, 58, 61
tools, 49, 52–53, 58, 61
ethics
management, 68
ethnography, 145
observation, 165
when to apply, 146, 158, 169–70
“Even Monkeys Fall from Trees” and *Other Japanese Proverbs* by Galef, 223
Evers, Ed, 73
evolutionary development. See incremental development
examples
elevator, 42–43, 120, 134–35
fruit fly, 119
hotel telephony, 147
remote mouse, 156
telephony, 119–20, 147
excellence
expecting, 46
surrounding yourself with, 109
executive
as change agent, xii
exit strategies, 111
INDEX

Exploring Requirements: Quality Before Design by Gause and Weinberg, 153, 157
extreme programming, 21, 171, 176, 178
backlash, 21
Extreme Programming Explained by Beck, 25, 176, 177

Fagan, Michael, 44
Fairly, Richard, 44, 90, 134, 141
Farley, Kristen, 166, 177
Farson, Richard, 116, 222
Faulkner, William, 227

feature points, 50, 62, 82
why better, 50
features. See also requirements
definition, 119
rejected from earlier releases as requirements source, 157

Federighi, Francis, 19, 24

Fifteen Principles of Software Engineering, 2, 41–47
seeds for debate, 47

finance
department as a machine, 77–79
education, 77
role in requirements sign-off, 182
financial experience, 82
plan, 112

finite-state machines, 115
role in software development, 242

firing employees, 65–68
pain of, 67

Fleming, Quentin, 71

foreign key, 131–32
formal methods, 241–42
Fortran, 249

founder’s shares, 90
France, Anatole, 136
Freedman, Dan, 243

freedom
academe, 188
industry, 188
friends and family financing, 97

From Wonderland to the Real Problem, 185, 207–13
seeds for debate, 213
fruit fly, 119

Fujigaki, Yuko, 209
function points, 50, 62, 82
why better, 50
fundamentalism, 21–22
and libertarianism dynamic, 23–24, 26
definition, 21
effect on software industry, 21–22

Furey, Sean, 50, 61

Gaffney, John, 50, 61
Galef, David, 223
Galletta, Dennis, 210
Gamma, Erich, 250
Gane, Chris, 19, 25
Garlan, David, 139, 141, 243, 250
Gates, Roger, 165, 176
Gause, Donald, 42, 138, 141, 144, 153, 157, 164–66, 175, 176, 202, 205, 243
Geller, Rob, 82–83, 89, 90, 109, 112
genericity, 135
Gilb, Tom, 45, 243

glaciers
analogy to method fads, 249–50
Glaser, Barney, 168, 172, 177
Glass, Robert, 113, 136, 141, 164, 176, 185, 207–13, 219, 240
Goguen, Joseph, 145, 153, 157, 164, 166, 176

going public, 111
golden rule
modified, 222
Gomaa, Hassan, 139, 140, 141, 243
Gottesdiener, Ellen, 20, 25, 165, 166, 176
Graham, Dorothy, 45, 243
group session. See collaborative session
GroupSystems, Inc., 156, 158
Growth Strategies, Inc., 89, 109

GTD-120, 115

Hall, Anthony, 243

handgun control, 21
hardware requirements specification
definition, 151
Harel, David, 121, 177
Hatley, Derek, 5
Hay, David, 166, 176
Heninger, Kathy, 164, 175
Hewlett-Packard, Inc., 83, 89, 90, 227
Hickey, Ann, 114, 163–78, 199–206
high-level programming languages, 18–19, 125
backlash against, 19
Hillel, 228
hiring freezes, 193
Hoffer, Jeffrey, 166, 177
home construction
compared to software development, 27–28
different from software development, 30
Homer, 185
Hooks, Ivy, 166, 177
hospitals
data collection, 6
hotel example, 147
Hudlicka, Eva, 164, 166, 176
Hull, Elizabeth, 166, 177
hype, 3–10, 211–12
IBM, Inc., 24, 42, 109, 156, 208, 227
IEEE Computer Society portal, 156
IEEE International Conference on Requirements Engineering, 250
IEEE International Conference on Software Engineering, 6
IEEE Standard 830-1998, 151, 153
income statement, 79, 112
Incorvaia, A. J., 243
incremental development, 43, 146, 193, 196
when to apply, 146
industry
career in academe vs., 187–90
freedom, 188
impact, 189
job security, 188
personality traits, 190
politics, 187, 189
Information for Decision Makers, 63, 107
seeds for debate, 107
inheritance, 129
initial public offering (IPO), 111
inspections, 38–39, 42, 44–45, 242
instantiation, 135
integrity, 112
intellectual
distance, 44
property, 84, 97
internship, 216
Internet
have’s and have not’s, 245
really important?, 245–48
interviewing, 144, 165
definition, 144
never rely on completely, 169
when to apply, 146, 157–58, 169
investors
angels, 90–91, 97, 106, 111
communicating with, 96
friends and family, 97
venture capital, 91, 97, 111
IPO. See initial public offering
Iraq, 1
ISO 9000, 12
issues list
during elicitation, 171
It Feels Like Déjà vu All Over Again, 2
seeds for debate, 36
iterative development, 43, 146, 193, 196
Jackson System Development, 44
Jackson, Michael, 5, 44, 136, 164, 175,
205
Jacobson, Ivar, 20, 25, 130, 132, 136, 141,
145, 153
Java (language), 9, 251, 253
Jennison, Leslie, 158
Jewish, 1
Jirotka, Marina, 145, 153, 157
job security
academe, 188
industry, 188
Johnson, John, 166, 177
Johnson, Kent, 21, 25
Johnson, Samuel, 222
Joint Application Development by Wood and Silver, 175
Jones, Capers, 2, 49, 50, 52, 55, 60, 61,
156–57, 163, 175
Jørgensen, Magne, 49, 60
Juristo, Natalia, 39
Just Enough Requirements Management by Davi ...
Kgalagadi. See Kalahari Desert
Kitchenham, Barbara, 50, 61
Knoedler, Thomas, 74
know thy customer, 199, 203
KnowledgePLAN, 52
Kooshian, Sarah, 74
Koppelman, Joel, 71
Kotonya, Gerald, 143, 153, 160, 164, 166, 175
Kowal, James, 165, 176
Kulpa, Margaret, 21, 25
languages
selecting, 43–46
trends, 251
Laranjeira, Luiz, 50, 61
Lauesen, Sorel, 164, 166
Layzell, Paul, 158
leadership, 226
Lee, Jonathan, 20, 25
Leffingwell, Dean, 109, 157, 160, 164, 166, 176
left-brained, 20, 79, 113, 116–17, 202
LeGuin, Ursula, 222
Lehman, Manny, 32, 146, 153
lemmings, 1, 3–10, 212, 222, 239
level playing field, 127–28
liberal, 2
libertarianism, 22–23
and fundamentalism dynamic, 23–24, 26
definition, 22
effect on software industry, 22–23
Lichtenberg, Georg, 228
Lieutenant America and Miss Apple Pie by DeMarco, 237
life beyond software, 222
Life and Software, 185–248
Lin Chi, 223
Linde, Charlotte, 164, 166, 176
Lindvall, M., 134, 141
lines of code, 50, 62
why better, 50
Lirou, Yuval, 74
listening, 116
Lister, Tim, 166, 235–36, 237
Lockheed Martin, Inc., 106
Lockwood, Lucy, 166
Lorenz, Mark, 133, 141
Loucopoulos, Peri, 143, 153
Macaulay, Linda, 164, 166, 175
Mad Hatter, 207
Mahlangu, Bongazana, 247, 248
Maiden, Neil, 164, 166, 176
Making a Mark on the World, 219,
235–37
epilogue, 237
seeds for debate, 237
management, 63–112
appropriate background, 6
as change agent, xii
ethics, 68
importance of good, 45
in the gut, 227
inexperienced, 65–68
laws, 69
mistakes, 69
one of the guys, 66, 68
role in estimation games, 56
role in requirements sign-off, 182
transition from technical role, 65–68, 69
what makes a good manager, 116
Management of the Absurd by Farson, 116
Managing Software Requirements by Leffingwell and Widrig, 157, 176
Managing the Design Factory by Reinertsen, 153, 158
Marca, David, 143, 153
market
research, 171
share, 32, 150
size, 84, 126
window, 51, 150
marketing, 51, 55, 58, 59, 85, 180
as a machine, 77–79
as requirements source, 157
communications, 87–88
compromising with development, 59–60
conflict with development, 150
distrust for development, 126, 179, 202
education, 77, 127
experience, 82
involvement from the beginning, 96
is right, 61
market research, 171
market share, 32, 150
market size, 84, 126
market window, 51, 150
marketing (continued)
vice president of
hiring, 87–88
qualities of, 97
Marriott Hotels, 227
Maslow, Abraham, 194, 195, 247, 248
Mastering the Requirements Process by
Robertson and Robertson, 176, 205
McCabe complexity measure, 46
McCabe, Tom, 46
McConnell, Steve, 21, 25, 29, 156–57
McDaniel, Carl, 165, 176
McGowan, Clement, 21, 25, 143, 153
Mead, Nancy, 75
measurement. See metrics
medical school, 215–17, 252
medicine
compared to software development, 1, 215–17, 252
Mellor, Stephen, 133, 141
methods
abuse, 1, 11–13
evolution, 249–50
metrics
are good, 22
feature points, 50, 62
function points, 50, 62
in technology transfer, 37–38
lines of code, 50, 62
magic, 49, 50, 55–56
meaningful, 31–34
parable, 229–33
subversion, 33
trends, 251–52
Meyer, Bertrand, 129, 133, 141
Microsoft, Inc., 251
Excel, 84, 86, 112
Project, 87
Mili, Mafedh, 73
Miscellaneous Thoughts on Evolution,
219, 245–48
seeds for debate, 248
Misra, Santosh, 73
models, 165, 178
favorite techniques, 170
using multiple models, 170
when to apply, 170–71
Modeling Reactive Systems with Statecharts
by Harel and Politi, 121
moderate, 1, 21
Modern Structured Analysis by Yourdon, 153
Modern Systems Analysis and Design by
Hoffer, 177
Moore, Geoffrey, 150, 153, 164, 175
More Words of Wisdom, 219, 225–28
seeds for debate, 228
Moreno, Ana, 39
Moslem, 1
Murphy, Edward, 225
music
software vs., 229–33
Nakajima, Tsuyoshi, 140
Namibia, 245
National Security Agency, 102
New York City
subways, 3
Norris, Pippa, 245, 248
Nummerich, Herr, 229–34
Oberg, Roger, 89
object
aggregation, 134–35
analysis, 4, 6
definition, 129
design time, 131, 134
differences between OOA and OOD, 133–36
dynamic binding, 135
genericity, 135
instantiation, 135
lemming path, 4, 5–6
orientation
software architecture and, 142
oriented analysis, 20, 113, 115, 129–42,
143, 202
backlash, 20, 23
useless for requirements specification, 136
oriented design, 4, 5–6, 12, 19–20, 22,
44, 113, 129–42, 212
backlash, 23
purpose, 131
oriented programming, 4, 5, 113, 129
oriented requirements. See
object:oriented analysis
requirements time, 131, 134
same as “good,” 5
Object (continued)

validation, 136
verification, 136
Objective-C (language), 129
Object-Oriented Analysis and Design by
Booch, 25, 153, 175
Object-Oriented Analysis by Coad and
Yourdon, 25, 140, 153
Object-Oriented Analysis to Object-
Oriented Design: An Easy
Transition*, 113–14, 129–42
epilogue, 141–42
seeds for debate, 142
Object-Oriented Design by Booch, 140
Object-Oriented Design by Coad and
Yourdon, 140
Object-Oriented Modeling and Design by
Rumbaugh, 141
Object-Oriented Programming by Cox, 140
Object-Oriented Software Construction by
Meyer, 141
Object-Oriented Software Development by
Lorenz, 141
Object-Oriented Software Engineering by
Jacobson, 25, 141, 153
Object-Oriented Systems Analysis by
Embley, 141
Object-Oriented Systems Analysis by Shlaer
and Mellor, 141
observation, 165. See also ethnography
when to apply, 158
Odysseus, 185, 187, 189
office space, 94–95, 96, 102
cubicles, 236
no bathrooms, 94–95
Omni-Vista, Inc., 63, 81–97, 109–10, 112
acquisitions, 93–94
board of advisors, 90
board of directors, 89
BTG vs., 106
business plan, 90–91
cradle to grave, 8 1–98
end, 81, 92–93
estimation game playing, 56
facilities, 94–95
financing strategies, 90–93
how estimation performed, 58
intellectual property, 84, 97
marketing and sales, 87–88
partnerships, 93–94
products, 84–86
staff, 86–87
tools, 84–86, 156, 160
On Software Development Strategies,
Politics, and Religion, 1, 17–26
seeds for debate, 26
OnYourMark Pro, 85–86, 93, 156
open-mindedness, 116
optimization, 44
Oregon Graduate Institute, 90
outsourcing, 209, 213
OVSP, 84–86
ownership of start-ups, 90, 109
P&L statement. See income statement
pain
relationship to requirements, 157
sheet, 158
parking lot, 171
partitions
as offices, 236
Patterns of Software Failure and Success
by Jones, 156–57, 175
Paulk, Mark, 20, 25
payroll
can’t make it, 87, 104
people
importance of, 6
time vs., 46
Peopleware by DeMarco and Lister,
235–36, 237
performance, 138
periodicals
for practitioners, 204, 206
for researchers, 204, 206
Persig, Robert, 100, 221
personal guarantees, 95, 96
personnel
insufficient, 193–94
Peters, Larry, 139, 141
Petri nets, 115
role in software development, 242
Pfleeger, Shari, 30, 49, 60
phenotypes
system vs. genetics, 119–24
physicians
education vs. software developers,
215–17

INDEX
INDEX  265

Pohl, Klaus, 182, 183, 184
Polanyi, Michael, 166, 177
Politi, Michal, 121
politics, 1–2
academe vs. industry, 187, 189
avoiding collaborative sessions, 169
conducting interviews, 169
ethnography and, 169
industry vs. academe, 187, 189
software development and, 17–26
Porras, Jerry, 13, 52, 62, 227
positioning, 84, 97
Potts, Cohn, 196–97, 203, 205, 252
Practitioner, Heal Thyself, 185, 2 15–17
epilogue, 216–17
seeds for debate, 217
practitioners, xi–xii, 185, 201
reading journals, 204
Primavera, Inc.
tools, 160
principles, 221
before process, 194–95
before tools, 194–95
of software development, 2, 41–47
should they change over time?, 47
priority of requirements, 147
problem analysis, 130
Problem Frames by Jackson, 175, 205
process
before tools, 194–95
maturity, 1, 4, 6–7, 12, 45, 249
model
standardization, 210, 213
principles before, 195–96
selecting appropriate, 6–7, 43
product
delivery record, 49
introduction
strategy, 95
late delivery, 85
plan, 159–60
planning, 125–28. See also triage
part of requirements?, 128
positioning, 84, 97
Product, Not Process: A Parable, 219,
229–33
epilogue, 231
seeds for debate, 233
professors. See also academe
problems, 215–17
profit, 31, 32, 84
profit and loss statement. See income
statement
program management, 94
programming languages, 18–19, 125
maturity, 1
progress
definition, 246
illusionary, 245–46
regularity of reporting, 71
project
management, 81–82, 93, 94
management punished, 18
qualities of, 71
punishment for failure, 18
prototyping, 42, 45, 145, 165, 171, 200,
212, 241
how to use, 7–8
lemming path, 4, 7–8
when to apply, 146
Putnam, Larry, 2, 50, 52, 61
QSS. See Quality Systems and Software,
Inc.
quality
assurance
role, 28–29
definition, 41–42
importance of, 41–42
measures, 11–12
retrofitting, 7
software, 6
Quality Systems and Software, Inc.
tools, 156, 158, 160
questionnaires, 145
when to apply, 146, 171
Rapid Application Prototyping by Andriole,
140, 152, 243
Rapid Development by McConnell, 156–57
Rapid Evolutionary Development by
Arthur, 152
Rational Software, Inc. 89, 109, 177
tools, 156, 158
rational thought, 2
Rational Unified Process, 142
rationalism
in technology transfer, 37
INDEX

RCA, 101

Recovering from Method Abuse, 1, 11–13

seeds for debate, 13

Redwine, Samuel, 205

Reifer, Donald, 21, 25, 166

Reilly, Edward, 19, 24

Reinertsen, Donald, 150, 153, 158

releases

multiple simultaneous, 193, 196

religion, 1–2

software development and, 17–26

remote mouse, 156

repeatable, 6

requirements, 113–84, 125

annotation, 148–49

attributes of well-written, 151

avoiding solutions, 147, 153–54

balancing against schedule, 82, 150

baseline agreement, 182

baselining, 179

bibliography, 156

changes, 180–81, 182–83

creep, 49, 204

definition, 119, 155

effort to satisfy, 148

elicitation. See elicitation

importance, 147

level of detail, 120, 147

made easy, 155–62

management, 93, 199

definition, 155

history, 155

relationship to project success, 156–57

tools, 156

maximum rate of change, 180–81

metrics

number agreed upon per hour, 33

practitioners

what do they do?, 20 1–2

priority, 147

qualities of, 160

quality, 143–54

range of activities, 113, 161

relationship to selling, 157, 161

relationships among, 148, 149

relative importance, 147

relative risk, 148

relative volatility, 148

research, 199–206

researchers

systemic problems, 204–5

what do they do?, 202–3

right level of detail, 120

risk, 148

sources, 157

specification, 122–23, 130, 144, 147–49, 151–52, 160–61. See also software

requirements specification and system requirements specification

content, 151

definition, 151, 156, 160

from elicitation, 133

how optimized, 137–38

level of detail, 156

purpose, 160

storing in a database, 147

transitioning to design, 6, 129–42

trends, 250

triage. See triage

understanding more important than documenting, 115

volatility, 148

Requirements Analysis by Hay, 176

Requirements Are But a Snapshot in Time, 114, 179–84

seeds for debate, 183–84

Requirements by Collaboration by

Gottesdiener, 25, 176

Requirements Engineering by Jirotka and Goguen, 153, 157

Requirements Engineering by Kotonya and Sommerville, 153, 160, 175

Requirements Engineering by Macaulay, 175

Requirements Engineering by Sommerville and Sawyer, 153, 177

Requirements Engineering by Wieringa, 153

Requirements Engineering Specialist Group, British Computer Society, 156

Requirements Language Processor, 191–92, 193, 196

Requirements Management Made Easy, 114, 155–61

seeds for debate, 161
Requirements Researchers: Do We Practice What We Preach?, 185, 199–206
   seeds for debate, 206
Requisite, Inc. 83, 89, 90, 109
RequisitePro, 156
research
   emphasis on publishing, 204
research and development
   as a machine, 77–80
researchers
   perspective of customer, 187–98
   understanding literature, 199
view of customer problems, 199
reuse, 9, 45, 242
   lemming path, 4, 8–9
   vs. “software use,” 8
revenue, 84, 144, 150
reviews, 242
Rewards of Taking the Path Less Traveled, 219, 239–43
   seeds for debate, 243–44
Richards, Neil, 158
Riddle, William, 205
right-brained, 20, 79, 113, 116, 117, 201, 202
risk
   associated with requirements, 148
   management, 90
Robertson, James, 164, 166, 176, 204, 205, 206
Robertson, Suzanne, 164, 166, 176, 204, 205, 206
Rockaway Beach, 3
   role playing, 66, 67
Rolodex
   role in marketing, 101
Roosevelt, Theodore, 226
Rosenberg, Doug, 20, 25
Ross, Douglas, 164, 175
Royce, Winston, xi
Rugg, Gordon, 164, 166, 176
Rumbaugh, James, 132–33, 136, 141
RUP. See Rational Unified Process
SADT by Marca and McGowan, 153
sales
   how to do it, 86–87
   as requirements source, 157
   salvaging, 242
Sanden, Bo, 139, 141
Sanders, James, 219, 229–33
Sarson, Trish, 19, 25
Sawyer, Peter, 143, 153, 166, 177
Saxena, Garurank, 74
scaffolding, 28–29
scenarios, 20, 141, 170, 202, 204
history, 20
   limitations, 20
schedule
   balancing against requirements, 82, 150
   compression, 54–55, 59–60, 69
   reason, 226
   estimation. See estimation.
   quality vs., 236
Schutawie, Jay, 74
SCIF, 102
Sebesta, Robert, 19, 24
security, 138
Seidman, Irving, 166, 167, 177
self-financed, 111
selling
   requirements gathering vs., 157, 161
Seven Habits of Highly Effective People by Covey, 250
shareholders, 81
Shaw, Mary, 27, 30, 139, 141, 243, 250
Shlaer, Sally, 133, 141
Should He Stay or Should He Go? Advice for a Beleaguered Manager, 63,
   73–76
   epilogue, 75
   seeds for debate, 75–76
Shiver, Bruce, 35, 36
Siddiqi, Jawed, 44, 134, 141
silos, 77, 127
silver bullet, 164
   marketing strategy, 87
Silver, Denise, 164, 175
Sinclair, Dale, 124
situational research, 204
Stack by DeMarco, 237
SLIM, 52
society
   corporate contributions to, 32
Software architecture. See architecture
companies as machines, 77–80
Crisis, 207–8
design. See design
designers. See designers
developers. See developers
development. See development
education. See education
engineering
in 1964, 41
principles of, 41–47
engineers
appropriate background, 6
Entropy. See entropy
industry, 1–62
Obsolescence, 14–16
Progress, 35
reputation, 107
Segmentation, 249
Stagnation, 35–36
instability, 193
Lemmings. See lemmings
management. See lemmings
music vs., 229–34
research, 185–213
reuse. See reuse
requirements specification. See software
requirements specification
startups. See startups
Software Architecture and Design by Witt, 141
Software Architecture by Shaw and Garlan, 250
Software Construction by Object-Oriented Pictures by Cherry, 140
Software Design by Budgen, 140
Software Design by Peters, 141
Software Design Methods for Concurrent and Real-Time Systems by Gomaa, 141
Software Engineering Concepts by Fairley, 141
Software Engineering Institute, 210
Software Inspections by Gilb and Graham, 45, 243
Software Lemmings, 1, 3–10, 212, 239
epilogue, 9–10
seeds for debate, 10
Software Requirements: Objects, Functions and States by Davis, 114, 140, 153, 160, 176, 243, 250
Software Requirements Engineering by Dorfman and Thayer, 153, 160
software requirements specification, 115, 138
definition, 151
necessity, 136–37
source for candidate objects, 139
Software Requirements Using the Unified Process by Windle and Abreo, 177
Software Systems Construction with Examples in Ada by Sanden, 141
Solution Selling by Bosworth, 157, 161
solving problems
rather than looking like solving problems, 193
Some More Tips for the Would-Be Entrepreneur, 63, 111–12
epilogue, 112
seeds for debate, 112
Some Tips for the Would-Be Entrepreneur, 63, 109–10
epilogue, 110
seeds for debate, 110
Sommerville, Ian, 143, 153, 160, 164, 166, 175, 177
South Africa, 1
South Pacific, 226
Sparks, Ben, 90
SPICE, 12
Spiral model, 43, 212
spreadsheets
for requirements management, 158
Spurr, Kathy, 158
stakeholder
definition, 157
Standish Group, 17, 24, 125, 156, 163, 175, 201, 206
Starbase, Inc, 156
startups, 81–97, 99–106
statecharts, 121, 170
role in software development, 242
Steiner, Alan, 90
Stevens, Wayne, 19, 24
stock options, 86
Storage Tek, Inc, 81–82
stories, 167. See also scenarios
storyboard, 145, 170
  when to apply, 146
Strauss, Anselm, 168, 172, 177
structured, 19
  analysis, 4, 5, 19, 115, 132, 143, 202
  backlash, 19, 23
  design, 4, 5, 19, 20, 136
  backlash, 19, 23
  lemming path, 4, 5
  programming, 4, 5, 21
  backlash, 23
  same as “good,” 5
*Structured Analysis and System Specification* by DeMarco, 25, 141, 175
*Structured Design* by Constantine and Yourdon, 46, 140
*Structured Systems Analysis* by Gane and Sarson, 25
Sun Microsystems, Inc. 251
surveys
  when to apply, 158
SWOT, 88
symphonies
  software vs., 229–33
system, 200
  design, 138
  phenotypes, 119–24
  requirements specification. See system requirements specification
System Development Corporation, 101, 102
*System Phenotypes*, 113, 119–24
  seeds for debate, 124
*System Requirements Engineering* by Loucopoulos and Karakostas, 153
system requirements specification, 138
  definition, 151
Taylor, Bruce, 20, 25
TBI. See Technology Builders, Inc.
  techniques
    adoption, 38
    before tools, 44
  technology
    as scapegoat, 213
    transfer, 2, 37–39, 187–98
Technology Builders, Inc. 158
  tools, 156, 158
Telelogic, Inc. 156, 160
telephony, 119–20, 147
Tenner, Edward, 225
tenure, 188–89, 190, 204
Teresa, Mother, 235, 236
Test Plan Generator, 192, 193, 196
testing
  independent, 46
  measuring progress, 70
Thayer, Richard, 143, 153, 160
*The Deadline* by DeMarco, 226, 227, 236–37
*The Future*, 249–53
  seeds for debate, 253
*The Harmony in Rechoirments*, 113, 115–17
  seeds for debate, 116–17
*The Logic of Failure* by Dörner, 11, 153
*The Missing Piece of Software Development*, 113, 125–28
  seeds for debate, 128
*The New Software Engineering* by Conger, 176
The Requirements Place, 156
*The Rise and Fall of a Software Startup*, 63, 81–97
  seeds for debate, 97
*The Software Company Machine*, 63, 77–79
  seeds for debate, 79
*Thoughts on Software Estimation*, 2, 49–62
  seeds for debate, 61–62
throughput, 138
time
  people vs., 46
  to market, 84, 126
Titan Systems, Inc. 105–6
*Tomorrow’s Blacksmiths*, 1, 15–16
  epilogue, 16
  seeds for debate, 16
tools
  adoption, 38
  techniques before, 44
  principles before, 195–96
  process before, 195–96
  without technique, 35–36
total quality management (TQM), 211, 212
INDEX

Toward a Psychology of Being by Maslow, 248
TQM. See total quality management
Tracz, Will, 243
trade-off analysis, 241
triage, 113, 121, 122, 125–28, 144, 147, 150–51, 158–60, 200. See also product: planning
definition, 155, 158
from development manager perspective, 158–59
goal, 159
level of requirements detail, 150
tools, 156, 160
variables to change, 159
Trial by Fire: Saga of a Rookie Manager, 63, 65–68
epilogue, 68
seeds for debate, 68
tricks
in programming, 46
trouble reports, 194
as requirements source, 157
Turski, Vlad, 139
Tutorial: Software Reuse by Tracz, 243
UML. See unified modeling language
unambiguous, 130, 160
unified modeling language, 25, 170, 177
Unisys, Inc. 101
usability, 138
use cases. See scenarios
users
as requirements source, 157
satisfaction, 1
V & V. See verification and/or validation
validation, 136
VantageNet, 158
venture capital, 91, 97, 111
source of financing, 91
verifiable, 160
verification, 136
Vessey, Iris, 210
vice president of marketing
hiring, 87–88
qualities of, 97
viewpoints, 170
visual languages, 249
volatility of requirements, 148
walkthroughs, 242
Walkthroughs, Inspections and Technical Reviews by Freedman and Weinberg, 243
Walzing with Bears by DeMarco and Lister, 237
Ward, Paul, 5
Warren, Carol, 166, 177
waterfall model, 209–10
Watson, Thomas, 227
weaknesses
knowing your own, 110
website development, 93
Weinberg, Jerry, 42, 138, 141, 144, 153, 157, 164–66, 175, 176, 202, 205, 243
Western man
future for, 246
Westpac, 208–9
what vs. how, 119
Whited, John, 74
Why Build Software?, 2, 31–33, 243
seeds for debate, 33
Why Does Software Cost So Much? by DeMarco, 38, 116, 236
Why Industry Often Says “No Thanks” To Research, 185, 187–97
seeds for debate, 196–97
Why Leaders Can’t Lead by Bennis, 107
Widrig, Donald, 157, 160, 164, 166, 176
Wiegers, Karl, 164, 166, 176
Wieringa, Roel, 143, 153, 164, 165, 176
Windle, Daniel, 166, 177
win-win, 126
Wirfs-Brock, Rebecca, 133, 141
Witt, Bernard, 139, 141
Wood, Jane, 164, 175
word processors for requirements management, 158
Words of Wisdom, 219, 221–23
seeds for debate, 223
Writing Effective Use Cases by Cockburn, 25
xml, 9
XP. See extreme programming
Xue, Nien-Lin, 20, 25
Yadav, Surya, 164, 176
Young, John, 227
Young, Ralph, 166, 177, 180–81, 183
Yourdon, Edward, 5, 25, 46, 63, 64, 90, 132, 139, 140, 143, 153, 166, 200, 206
Yourdon, Inc., 90

Z (language), 116
role in software development, 242
Zen, 221, 222, 223
Zen and the Art of Motorcycle Maintenance
by Persig, 100
Zweig, Ann, 56, 63, 81–97, 113, 125–28, 141, 155–61, 184
biography, 83–84