Index

ABC. *See* Architecture Business Cycle
ABM. *See* Atomic Broadcast Manager
"Abstract common services" tactic, 106, 135, 138, 148–149, 442
Abstraction, 21, 37, 43, 210, 477, *478*
  mastering at CelsiusTech, 375–376
  and packaging, 142
  and SS2000, 396
Abstract syntax tree, for information extraction, 235, 236
Access control, 117, 118
Accessibility, and the Web, 331
Access layer, for libWWW, 336
Access (Microsoft), 468, 470
Access procedures, 54, 64
access_read relationship, 235
access_write relationship, 235
Account bean, 414, 415, *416*, *419*
Acronyms, 218, 485
Activation, of stateful bean, 410
Activation list, in Active Object design pattern, 124
Active design review, 268
Active Object design pattern, 124
"Active redundancy" tactic, 103, 141
ActiveX, 343
Activity control problems, 162
actually_calls relation, 240
Ada, 132, 139, 144, 371, 380, 387, 462
Ada package specification, and signature specification, 212
Ada programs, 142, 147, 378, 390
Adaptation data, 147–148
Adapters, 109
ADD. *See* Attribute-Driven Design
Address spaces, 139, *140*
"Adherence to defined protocols" tactic, 111
  and client-server view, 142
  and code templates, 149
  and integrability, 175
Ad hoc tools, for information extraction, 236
Advanced Automation System (AAS), 129, 131
Aerodynamics, and flight simulator, 181, 182
Aggregate, 122
Aggregation, in UML, 219–221
Aircraft structure, unclear mapping between software structure and, 181
Aircraft systems, and airframe, 193, 195
Air traffic control case study, 129–151
  architectural solution in, 135–149
  requirements and qualities in, 132–135
Air traffic controllers, 132, 133, *134*
Air traffic control system, *132*, *150*
  ABC applied to, *132*
  availability requirements for architecture of, 155
Air vehicle model, 182, 183
  application modules, 187–192
  executive modules, 185–187
  n-square chart for, 194, *195*
AIX (IBM), operating system, 144
  Ada tasks mapped onto processes, 142
  and SS2000, 373
Algorithms, voting, 102
"Allocated-to" relation, 38
Allocation structures, 36, 38–39
Allocation views, 41, 206
  options in, 208–209
  in UML, 227–229
Allowed-to-use structure, 60, *61*, 62, 63
Amazon.com, 348–349
Analysis results, in architecture background, 209
Analyze architectural approaches
  in ATAM evaluation phases, 282–284, 285
  in Nightingale system evaluation phases, 297–299, 302
"And" states, 210
Annotations, interfaces as, 224
Anomaly detection, 118
"Anonymity of processor assignment," and SS2000, 390
"Anticipate expected changes" tactic, 107, 138, 148, 442
AOLbyPhone, 454
AOL/Netscape, 347
Apache project, 346, 347
Aperiodic operations, 187, 191
Aperiodic processing, and flight simulator, 183
AppleScript, CGI scripts in, 339, 340
Application constituent, of Structural Model architectural pattern, 185
Application Data Type Module, 57, 62
Application module layer, for libWWW, 336

Bass.book Page 496 Thursday, March 20, 2003 10:29 PM

Application programming interfaces (APIs), 457
  and libWWW, 336
  and Luther architecture, 441
Application proxies, 343, 344
Applications, 138, 139, 147
  and Ada packages, 142
  as clients and servers, *141*
  and Luther architecture, 432, 433–434, 440–442
Application servers, for modifiability, performance, and scalability, 345
Application-specific business logic, and Luther architecture, 441
Architects, 69, 108, 233, 245, 285
  and ABC for the Web, 347
  and architectural approaches, 285
  and architectural documentation, 202–203
  architecture and background/experience of, 8
  and architecture reconstruction, 232
  and ATAM, 272, 277
  and conceptual integrity, 96
  design decisions by, 261
  and EJB system design, 423
  and evaluations, 267
  and impact of components, 482
  model problems used by, 463
  and Nightingale system, 289
  product line, 360
  and recovery of "lost" architecture, 247
  and relationship of tactics to architectural patterns, 123–124, 126
  and scheduling policies, 115
  and stakeholders, 7
  tactics used by, 99, 100
  and testing cost reduction, 118
  and usability, 122
  and viable product lines, 398
  views produced by, 145
Architectural abstractions, as hypotheses, 245
Architectural approaches
  and ATAM, 279, 282–284, 285
  example of analysis, *283*
  and Nightingale system, 293–294, 297–299, 302
Architectural changes, 31
Architectural decisions, 231
  architecture reconstruction and redetermination of, 246
  benefits of, 307
  costs with, 309
  quality requirements mapped to, 274
Architectural design, 23, 355
Architectural drivers, 154, 157, 166, 277
  choosing in ADD, 158
  determining, 155
Architectural evaluations, ATAM and limited time for, 288
Architectural integrity, 96
Architectural metrics, 265
Architectural mismatch, 33, 43, 456–462, 476
Architectural patterns, 24–25, *26*, 33, 42, 62, 99, 100, *126*, 127, 232, 462
  choosing, 157, 158, *159*
  and off-the-shelf components, 456
  and query languages, 237
  and reconstruction, 242
  and styles, 124–125
  tactics related to, 123–124
  use of, 279
Architectural quality, and system quality, 92
Architectural reconstruction, using results of, 232
Architectural representation, 42
Architectural solution for the Web, 334–340
  common gateway interface, 339–340
  early client-server architecture using libWWW, 337–339
  and libWWW, 334–337
Architectural solution, in air traffic control case study, 135–149
  adaptation data, 147–148
  client-server view, 141–142
  code templates, 148–149
  code view, 142
  fault tolerance view, 144–147
  ISSS physical view, 135–138
  layered view, 142–144
  module decomposition view, 138
  process view, 138–141
  relating views to each other in, 147
Architectural solution in flight simulation case study, 182–196
  decomposing groups into systems, 195–196
  functionality allocated to controller children, 192–193
  group decomposition, 193–195
  modules of air vehicle model, 185–192
  skeletal system, 192
  Structural Modeling architectural pattern, 185
  time treatment in flight simulator, 183–184
Architectural solutions
  achievement of application needs and, 150
  for J2EE and EJB, 406–419
  and Luther architecture, 434–451

Architectural strategies, 99, 100
and benefit and normalization determination, 313
and CBAM, 309, 316
ROI value for, 313
and scenarios addressed in NASA ECS Project, *320–323*
side effects, 312–313
Architectural structures
for A-7E avionics system, 47, *48*
choosing, 41–42
and organizational structure, 168, 169
relationships between, 40–41
Architectural style, 25, 349
Architectural tradeoffs, and ATAM, 307
Architectural views, 43, 66
and architecture presentation, 278
and ATAM, 308
in Dali, *242*
Architecture, 3, 25, 31, 43–45, 67, 71, 217, 218, 478. *See also* Software architecture
analyzing/evaluating, 14
and architects' background/experience, 8
for A-7E avionics system, 54–65
behavior of elements in, 22
communicating about, 13–14
components, connectors, and, 24
components and impact on, 455–456
constraints on implementation, 29
creating/selecting, 13, 69–70
designing, 155–166, 171, 479–481
developing organization and, 8
ensuring conformance to, 14
evaluations of, 23, 265
feedback loops and, 9–12
functionality and, 72
future of, 477–484
goals/requirements for, 266
"good," 14–16
hierarchy of test interfaces in, 120
as high-level design, 23
implementing based on, 14
influences on, 9, *10*
layered, 62
in life cycle, 153–155, 481–482
Mk2, 376
moving from, to code, 482
.NET, 402
Nightingale system and presentation of, 292–293
organizational structure dictated by, 29–30
as overall structure of system, 23
pivotal role of, 19
process-centered definitions of, 24
for product lines, 360–363
purposes of, 205
qualities in, 96–97
and quality attributes, 73–74, 453
rationale for, 218
relationship between organization and, 167
representations, 35
and re-use, 351, 353
and software product lines, 363
sources of, 6–12
stakeholders and communication needs served by, 27–29, *204*
stakeholders and influence on, 6–8
technical environment and influence on, 9
Architecture activities, 12–14
Architecture analysis, 261–269
costs and benefits, 263–265
planned or unplanned, 266
preconditions, 266–267
results, 268
techniques, 265–266
when, 262
why, 261–262
Architecture background, in documenting a view, 209
Architecture Business Cycle, 6, *11*, 12, 69, 71, 261, 271
and air traffic control system, *132*
and Celsius Tech, *370*–371, 383, 399
backward flow of, 452
current state of, 346–347
flight simulators and relationship to, 176–177
and future of software architecture, 479
relationship of, to A-7E avionics system, *48*–49
relationship of, to Inmedius and Luther architectures, 429–432
software processes and, 12–14
and story of *Vasa*, 4–5
and Sun and J2EE/EJB, 402, *403*
and Web-based architectures, 340–342
and the Web, 327, 328–329
Architecture documentation
in ATAM-based evaluation, 274
importance of, 22
and Nightingale system, 290
stakeholders and helpfulness of, *206*
and structures, 41
uses of, 203–203
Architecture presentation, example, *278*

Architecture reconstruction, 22, 231, 232,
 activities, 233, *234*
 complexity with, 258
 motives for, 481, 482
 and redetermination of architectural decisions, 246
Architecture stakeholders, and ATAM, 272
Architecture Tradeoff Analysis Method, 14, 31, 203, 262, 265, 268, 271–305, 307, 465, 466
 and evaluation team roles, *273–274*
 historical roots of, 305
 initial draft on training course on, 304
 in Nightingale system, 288–303
 outputs with, 274–275
 participants with, 272
 reviews based on, 263
 stakeholders and evaluations with, 267
 steps and outputs, correlated, *287*
Architecture Tradeoff Analysis Method evaluation phases steps, 276–285
 analyze the architectural approaches (step 6), 282–284
 analyze the architectural approaches (step 8), 285
 brainstorm and prioritize scenarios (step 7), 284–285
 generate quality attribute utility tree (step 5), 279–282
 identify architectural approaches (step 4), 279
 present architecture (step 3), 277–279
 present results (step 9), 285
 present the ATAM (step 1), 277
 present the business drivers (step 2), 277
Architecture Tradeoff Analysis Method phases, 275–288
 evaluation (phase 1), 276, 277–284
 evaluation (phase 2), 276, 284–287
 follow-up (phase 3), 276
 partnership and preparation (phase 0), 275
Arch/Slinky pattern, 123, 127
Arcs, decomposition in UML with, *221*
ArithmeticOp class, 239
ARM, 258
Artifact
 of modifiability general scenarios, 82
 of performance general scenarios, 84
 of security general scenarios, 87
 of testability general scenarios, 89
 of usability general scenarios, 91
Artifact stimulated, in quality attribute scenarios, 75
"As-built" architecture, 231, 232, 481
"As-designed" architecture, 232, 252, 481
ASEILM example. *See* Automated SEI Licensee Management system
A-7C aircraft, 49
A-7E architecture, 54–65
 allowed-to-use specification, *61*
 modifiability and performance requirements for, 154
 Physical Models Module of, 168
A-7E avionics system case study, 47–68
 A-7E project description, 52–53
 architectural structures, 47, *48*
 process structure, 63–65
 relationship to ABC, 48–49
 requirements and qualities, 49–52
 uses structure, 61–63
A-7E Corsair II, *49*, 51
A-7E module decomposition structure, 56–59, 73
 quality goals achieved by, *56*
A-7E module decomposition view, textual primary presentation in, 208
A-7E software, 49, *59*
 coarse-grained data-flow view for, *67*
 design of, 461
 primary sensors read/managed by, 50
As-implemented architectures, as-designed architectures related to, 252
Aspect-oriented programming, 120
Aspect-oriented software development, 481
Assemblies of compatible components, 453
Assembling elements, 33
Assembly language, 477
Asset base, and CelsiusTech product line, 396
Asset re-use, 8
Assign utility step, 316, 319
Association classes, connector types as, 226
Association role, UML, 225
Associations
 connector types as, 225
 and nodes, 228
Assurance, 86, 116
AST. *See* Abstract syntax tree
Asymmetric encryption, 117
AT&T, architectural evaluations at, 263, 269
ATAM. *See* Architecture Tradeoff Analysis Method
ATC. *See* Air traffic control
Atomic Broadcast Manager, 144
Attachment relation, 38
AttachOp class, 239
Attacker identification, 118

Attacks, 85, 86, 87
  detecting, 118
  recovering from, 118
  resisting, 116–117
Attribute-Driven Design method, 155–166, 171, 203
Attribute requirements, and product line architecture, 362
Auditing, 86
Audit trails, 87, 118
Authentication, 116, 339
Authorization, 117
Authorship, in documenting a view, 209
Automated SEI Licensee Management system, 466, 467
  example, 466–476
Automatic build checking, 211
Automatic detection, 434
Automatic repair strategies, 79
Automotive industry, and architecture reconstruction, 232
Availability, 7, 69, 79, 86
  and air traffic control system, 132–133
  and component repair, 457
  databases for, 345
  and e-commerce, 341
  load balancing for, 344
  and Nightingale system, 292
  of resources, 112
  Simplex architecture for, 127
  and SS2000 product line, 388
  and system quality attributes, 74
  and tactics, 100
  and Web-based applications, 404
Availability scenarios, 76, 77, 80, *81*
Availability tactics, 101–105, 125
  fault detection, 102
  fault prevention, 104–105
  fault recovery, 102–104
  and process view, 141
Avionics, 168, 193, 478
awk, 232, 239

Back-office systems, front-line workers integrated with, 427, 432, 452
Backup Communications Network, 137, 138
Balanced specificity, with J2EE and EJB, 406
Base System, 382
Base System 2000, 392
Basic SS2000 Configuration Project, at CelsiusTech, 385, 387
BCN. *See* Backup Communications Network
Bean-managed persistence entity beans, 410
Beans, 443
Behavior, 7, 230
  documenting, 209–211
  of elements, 22
Behavior-hiding module, in A-7E module decomposition structure, 56
Benefit, of architectural strategy, 313
Best-case quality attribute level, in utility-response curve, 311–312
Best-case response, for NASA ECS Project scenario, 317
Billing, 345
Binding time, 124, 209
Biometric identifications, 116
"Bird's-eye" program overview capability, 258
Blackboard architectural pattern, 57
"Black box" elements, 3
Blocked time, 112–113
Boolean algebra, 213
"Borgs," 428
Bottlenecks, 40, 171
Bottom-up adoption, of product line, 364
Bottom-up approach, for architecture reconstruction, 258
Bound execution times tactic, 114
Bound queue sizes tactic, 114
Box-and-line diagrams, 230
Box-and-line sketches, 24
Brainstorm and prioritize scenarios, 284–285, 300–302
Branching, and work-flow component, 445
Branching statements, 477
"Brick-and-mortar" bookstores, 348
Bridge pattern, 109
Bridges, 458–459
  and Logical Communications Network, 137
Broadly scoped product lines, 358
Broker bean, 414, *415*, 417, *418*
Broker EJB, 413
Broker pattern, 110
Broker remote interface, *414*
Browser-based clients
  intended use for, 439
  and Luther architecture, 437, 439
Browsers, 329, 342
BSpline element, 255
B-2 Weapons System Trainer, 177
Budgets, 170, 354, 355
Buffers, 112
Buhr-style use case maps, 248
Buildability, 97, 480
  of EJB-based systems, 416
Build files, 234, 236

Build first, buy-later strategy, 433
Built-in monitors, and internal monitoring, 120
Business case, 12–13
Business component tier, in J2EE multi-tier model, 408
Business constraints, and Nightingale system, 291
Business context, for CelsiusTech, 378–380
Business delegate pattern, 441, 452
Business drivers
  ATAM and presentation of, 277
  Nightingale system and presentation of, 291–292
  and risk themes, 302
  and SS2000 product line, 377
Business goals
  and ATAM, 271, 274, 308
  identifying, 155
  and quality requirements, 99
  and risk themes, 275
Business influences, and software architecture, 5, 6
Business logic, and component packaging, 443
Business qualities, 74, 95–96
Business requirements, 154
  and Nightingale system, 291
  and Web architecture, 340
Business rules, 345
Business-to-business Web sites, 340
Business-to-consumer Web sites, 340
Business units, and product development, 366
Buttons, 435

C, 144, 339
C&C. *See* Component-and-connector views
C++, 239, 339
Caching, 114, 336, 343, 345
Calculate total benefit from architectural strategy, 316, 322
Callback methods, 413
Call graph, 235
Call-out functions, 336
calls relationship, between functions, 60, 235, 240
Cancel, 122
Candidate view lists, producing, 206–207
Capability, and CelsiusTech product line, 396
Capability partitioning, and Luther architecture, 442
Captured rationales, and architectural evaluations, 264
Case studies
  air traffic control and designing for high availability, 129–151
  A-7E avionics system, 47–68
  CelsiusTech AB, 369–399
  flight simulation, 175–199
  Luther architecture, 427–452
  NASA ECS Project, 317–324
  Nightingale System, 288–303
  the World Wide Web, 327–349
CBAM. *See* Cost Benefit Analysis Method
CDL. *See* Chart description language
CelsiusTech, 351, 355, 431, 432
  corporate evolution of, 370, *371*, *377*
CelsiusTech AB case study, 369–399
  architectural solution, 390–398
  business context analysis, 378–380
  economics of product lines results, 373–374
  mastering abstraction, 375–376
  motivations at CelsiusTech, 376–377
  organizational structure at CelsiusTech, 380–387
  product schedules, *373*
  relationship to ABC, 370–374
  requirements and qualities, 387–388
CERN. *See* European Laboratory for Particle Physics
cfx program, 258
CGI. *See* Common gateway interface
Champions, and product line adoption, 364
Change
  architecture and management of, 31
  cost of, 80
  timing of, 81
Change histories, in documenting a view, 209
Chart description language, 470
ChartWorks (Visual Mining), 468, 469, 470, 472
C header (.h) files, and signature specification, 212
Checklists, 265
Checkpointing, and database models, 237
"Checkpoint" tactic, 104
Checksums, 117
Child modules, 157, 161, 163–164
Choose architectural strategies based on ROI, 316, 323–324
Choose deadline-driven scheduling policy, 159
"Chunking" design work, 156
Circle Module, 222
Class box, in UML, 219
Class containment, code segments for, *254*
class_contains-method relation, static and dynamic information about, *240*
Classes
  connector types as, 226
  interfaces as, 225

Bass.book Page 501 Thursday, March 20, 2003 10:29 PM

structure of, 38
in UCMEdit model, 250
UML, 223
Classifiers, in UML, 219, 220
Class symbol, in UML, 219
Clients, 334
applications as, *141*
and EJB, 414
testing by, 89
Client-server pattern, 25
Client-server structure, 38, 40
Client-server system, 170
Client-server view, 141–142, 147, 207
Client tier, in J2EE multi-tier model, 407
Clustering services, 422
Coarse-grained EJBs, and J2EE, 450
Cockpit display devices, in A-7E aircraft, 50
Code, 234
moving from architecture to, 482
repair, 458
re-use, 32, 374
servlet, 408
structure, 47
Code coverage analysis tools, 236
Code instrumentation, 236
Code patterns, 125
Coders, and architecture, 27
Code segments, 242, 244
for class/file containment, *254*
developing, 246
and UCMEdit, *252*, *255*
Code templates, 135, 141, 148–149, 150
Code view, 42, 142, 147
Cognitive psychologists, and Luther architecture user interface, 436, 437
Cohesion, and teams, 168
Cohesion metrics, 106
Cold restarts, 145, 146
Collaboration diagrams, in UML, 227
Collate scenarios step, 316, 317
COM, 236
COM+, 425
Command, control, and communication (C3) systems, 370, 394
Command-and-control systems, CelsiusTech as supplier of, 370
Commercial components, growth in capability/availability of, 482
Commercial off-the-shelf (COTS) products, 33, 95, 398. *See also* Off-the-shelf components
and Nightingale system, 289, 290, 292, 298, 303
Commercial world, success and failure in, 65–66
Common gateway interface, 338, 339–340
Common object manager, *391*
Common Object Request Broker Architecture (CORBA), 236, 292, 345, 401–402, 403, 425, 458
and EJB architecture interoperation with, 405
Common services, abstracting, 106, 107
Common System Services, in ISSS, 138
Communication
and architecture, 13–14
restricting paths of, 109
skills, 9
Communication virtual machine module, in example, 165, 166
Compiler generators, 12
Compile-time artifacts, 236
Compile-time switches, 81
Completeness, 96
Complexity, of flight simulators, 180, 181
Component-and-connector structure, 36, 38, 40, 41
Component-and-connector views, 66, 139, 162, 206, 207, 212
and architecture presentation, 278
options in, 208
in UML, 222–227
Component-based systems, 342
and architectural mismatch, 456–457
design of, 462
Component design
at Inmedius, 442
and J2EE, 450
Component developers, and Luther architecture, 444
Component Development Group, at Inmedius, 431
"Component glue," 457
Component/interface repair, 457
Component projects, and SS2000, 383
Component qualification, 460
Component replacement, 111
"Component replacement" tactic, and client-server view, 142
Components, 24, *224*, 436, 441
and allocation views in UML, 228
core, 442
and design process, 453
elements related to, 255, 256
future impact of, 482–483
impact of on architecture, 455–456
life-cycle services, 401
for Luther architecture, 442–444
and object containment, 227

Bass.book Page 502 Thursday, March 20, 2003 10:29 PM

Components *continued*
  types, in patterns and styles, 125
  in UML, 223–224
Composing elements, 33
Composition of systems, and SS2000, 387
Composition relation, in UML, 221
Computation, dependency on, 112–113
Computational efficiency, increasing, 113
Computational overhead, reducing, 113
Compute method, 239
Computer Software Components, 37
Computer Software Configuration Items, 37
  in ISSS, 138, 147
.com Web sites, growth of, 332
Conceptual integrity, 13, 16, 44, 74, 96
Conceptual view, 42
Concrete scenarios, 75, 78
Concurrency, 25, 210
  enhancing, 124
  introducing, 114
  Java servlet ensemble and management of, 475
  structure, 38
  view, 147, 161–162, 164
  and work-flow model, 446
Confidence, increasing, 91
Confidentiality, 86, 116, 117
Configuration control data, in documenting a view, 209
Configuration control system, 167
"Configuration files" tactic, 111, 147
Configuration management, and software product lines, 363
Configuration management systems, software architecture within, 481–482
Configuration parameters, 213
configure operation, 187, 189, 190, 191
Configure state, in flight simulator, 179
Confirm results with intuition step, for CBAM, 316
Connector instances, 225, 226
Connectors, 24, 36, 225–226
Connector types, 225, 226
constituent_event operation, 187
Constraints, 9, 165
Construction problems, and electronic flight simulators, 176
Contained objects, systems as, 227
Container-managed persistence, 410, 415, 422
Containers
  EJB, 408, 410, 413
  J2EE-compliant, 443–444
Container services, 413
contains class, 235
Contention for resources, 112
Content on the Web, labeling of, 333
Content providers, and ABC for the Web, 346, 347
Context
  of availability scenarios, 80
  diagrams, 208, 278
  of modifiability general scenarios, 82
  of performance general scenarios, 85
  of quality attribute scenarios, 75
  of security general scenarios, 87
  of testability general scenarios, 89
  of usability general scenarios, 92
Control, sequence of, 108
Control frequency of sampling tactic, 113
Controller children, 192–193, 198
Controller children module, and air vehicle structural model, 187, 190–192
Control mechanisms, 41
Control Process (CP), 20, 21
Controls (or widgets), 97
Conversational with respect to calling process, 409, 410
COOB. *See* Common object manager
Cooperating sequential processes, 52
Coordinating processes structure, 38
CORBA. *See* Common Object Request Broker Architecture
Core asset base, 354, 357
Core assets, and CelsiusTech, 374
Core layer, for libWWW, 335, 336
Correctness, 96
Cost and benefit, as business quality goal, 95
Cost Benefit Analysis Method, 14, 265, 268, 304, 308–325, 479
  basis for, 309–313
  decision-making context for, 308–309
  implementing, 314
  and NASA ECS Project, 317–324
  process flow diagram for, *315*
  and product line evaluations, 363
  results of exercise with, 324
Cost-effectiveness, with evaluations, 267
Cost efficiency, and Web-enabled enterprise systems, 423
Cost estimates, and software architecture, 32
Cost modeling, 325
  and buildability, 97
  importance of, 314
Costs
  with architectural decisions, 309
  and architectural evaluations, 263, 265

Bass.book Page 503 Thursday, March 20, 2003 10:29 PM

of change and modifiability, 80
development, 95
and flight simulators, 181, 196
of testing, 118
COTS products. *See* Commercial off-the-shelf products
Coupling, 106
metrics, 127
of scenarios and tactics, 480
Crew simulator training, 179
"Cross-cutting" requirements, 481
Cross-side scripting, 474
Cross-view documentation, 215, *216*
Cross-view rationale, 218
Cryptography, 349
CSCI. *See* Computer Software Configuration Item
CSCs. *See* Computer Software Components
C-17 Aircrew Training System, 177
C Shell, CGI scripts written in, 340
Currencies, and Nightingale system, 291
Current-case response, for NASA ECS Project scenario, 317
Current utility level, for scenario, 312
Custom clients, in J2EE, 439
Custom component, and Miva Script applications, 467
Customer projects, and SS2000, 383
Customer relations management, 345
Customer requirements, 11
Customers
and ABC for the Web, 347
and architecture, 27
and evaluations, 267
Custom user interfaces, advantages with custom Web-based clients over, 440
Custom Web-based clients, and Luther architecture, 437, 439–440
Cutovers, 146
C/X-band antenna, on Göteborg class, 371
"Cyberporn" industry, 333
Cyclic executive schedule, 115

Dali, 233, *238*, *242*, 247, 258
code segments created in, 248
and visualization and interaction, 241
workbench, 238
Dashed-arrow dependencies, and allocation views in UML, 228
Data, 109
semantics and sequence of, 108
syntax of, 107
Data access, designer's decisions, J2EE, and implementation of, 450
Data Access Working Group, of ECS Project, 317, 321
Data analysis, and the Web, 331
Data Banker Module, 57–58
Database construction
and database tables, 239
guidelines for, 239
and reconstruction, 233, 237–239, 248–249
Database management, 478
Database management systems, 453
Database models, choosing, 237
Databases, for performance, scalability, and availability, 345
Data confidentiality, maintaining, 117
Data controllers, 133–134
Data exchange problems, 162
Data extraction tools, 233
Data-flow analysis, 477
Data-flow view, 66
Data integrity problems, 162
Data loss, 85
Data-producing conventions, and COOB, 391
Data publishing, and the Web, 331
Data sources, separation and abstraction of, 435
Data tags, and ISSS physical view, 136
Data type definitions, in interface documentation, 213
Data types, and avionics applications, 57
DAWG. *See* Data Access Working Group
Deadline monotonic, and fixed-priority scheduling, 115
Deadlines in processing, 84
Deadlock, 40, 209
avoidance of, 64
Dead man timer, 102
DBMS. *See* Database management systems
Debugging, and cost with flight simulator, 181
Decision Support and Report Generation Manager, 292, 298
Declarative transaction management, 450, 451
Decomposition structure, 36–37, 40
for A-7E avionics system, 54–59
Decomposition view, 147, 206–207
Defect elimination, and re-use, 356
"Defer binding time" tactic, 110–111, 159, 417
Define the starting evaluation criteria (step 2), in model problem process, 469
Demand process, 63
Demilitarized zone, 117
Denial-of-service attacks, 86, 87
Dependency, UML notation for, 222
Dependency types, 107–108
Deployment, and quality attributes, 73

Bass.book Page 504 Thursday, March 20, 2003 10:29 PM

Deployment descriptors
  for Account entity bean, *419*
  for Broker bean, 417, *418*
  and EJBs, 417
  and support for Sun's J2EE quality attribute requirements, *418*
Deployment structure, 38–39
Deployment time, 110
Deployment type diagram, 228
Deployment view, 41, 164, 205
  with ADD, 163
  and architecture presentation, 278
  of Luther applications, *436*
  in UML, *229*
  UI as a C&C view mapped onto, *439*
Design
  artifacts, 236
  attribute-driven, 155–156
  beginning, 154–155
  of flight simulator software, 181
  improving, 480–481
  and interface documentation, 214
  judging, 69
  and quality attributes, 73
  questions, 464
  rationale for, 214
  reviews, 119
  as search, 462–465
  work-flow component, 445–447
Design alternatives, restricting vocabulary of, 33
Design decisions, 261
  quality attribute requirements related to, 305
  and software architecture, 26, 29–32
Designers, and evaluations, 267
Design patterns, 99, 125, 173
  and ARM, 258
  Business Delegate, 441
  EJB, 422
Design teams, model problems used by, 463
Design-time tactics, 123
Desired-case response, for NASA ECS Project scenario, 317
Desired utility level, for scenario, 312
Detached operations, and work-flow model definition, 449
Determine utility of "expected" quality attribute response levels step
  for CBAM, 316
  in NASA ECS Project, 321–322
Develop architectural strategies for scenarios step, 316, 320
Developers, 13
  and architectural integrity, 96
  and ATAM, 272
  changes made by, 81, 82
  and usability, 123
Developing organizations, 8, 10
Development costs, 95
Development department model, and product development, 366
Development Group, at CelsiusTech, 384, 385
Development view, 41
Device Interface Module, 58, 62, 62
Device Interface procedure, 64
Device types, and Luther architecture, 432, 433
Diagnosis module, in example, 165, 166
Dials, 435
Digital artifacts, and work-flow component, 445
Digital certificates, 116
dir_contains_dir, 235, 239
dir_contains_file, 235
Directory structure, code segments based on, 246
Display Management, in ISSS, 138
Distributed computing
  growth of, 402
  and Luther architecture, 432, 434
Distributed object-oriented systems, 401
Distributed objects, 8
Distributed object technology, and enterprise information systems, 403
Distributed operations, and work-flow model definition, 449–450
Distributed systems, and deployment structure, 39
Distributed transactions, and EJB systems, 423
Diversity, 103
DMZ. *See* Demilitarized zone
Documentation, 70, 211–214, 354
  of architectures, 14
  in ATAM-based evaluations, 274, 275
  of behavior, 209–211
  of decomposition structure, 55
  of interfaces of modules in ADD, 163–164
  as introduction to software architecture, 203
  and Nightingale system, 290, 303
  for product line architecture, 361
  for software interfaces, 58
  stakeholder access and organization of, 216–218
  within tool environment, 481
Documenting software architectures, 201–230
  basic principle of, 205
  choosing relevant views, 205–207

Bass.book Page 505 Thursday, March 20, 2003 10:29 PM

documentation across views, 215–218
and UML, 218–229
Documenting views, 204–205, 207–214
architecture background, 209
context diagram, 208
element catalog, 208
glossary of terms, 209, *210*
primary presentation, 207–208
variability guide, 208–209
Documents, 478
Document Type Definition (DTD), 417
Domain engineering units, and product development, 366, 367
Domain experts, and Luther architecture user interface, 436, 437
Domain model, 53
Domain name services, 117, 118
Domains, 168, 478
Domain-specific architectures, availability of, 482
Domain-specific components, 442
Downdrafts, 179
Downtimes, 79, 104
Drift angle, 50
DSRGM. *See* Decision Support and Report Generation Manager
Dual redundancy, 103
Dynamic extraction, 241
Dynamic information, 235, 236, 237, 239, *240*
Dynamic link libraries (DLLs), 361
Dynamic priority scheduling, 115
Dynamic views, 15, 236

Earliest deadline first, and dynamic priority scheduling, 115
Earth Core System (ECS), 317
Earth Observing System Data Information System (EOSDIS), 317
e-commerce architecture, 332, *341*
and ABC, 340–342
applications, and J2EE framework, 403
quality goals achieved by, *346*
e-commerce systems
deployment view of, 342
and entity beans, 422
and performance, 349
Economic models, of software, 307, 308
Efficiency, and usability, 90
EJB client code, simplified example of, *415*
EJBContext interface, 413
EJBHome, 413
EJBObject interface, remote interfaces and extension of, 413
EJBs. *See* Enterprise JavaBeans
Electronic flight simulators, 176
Electronic warfare systems, 370
Element catalog, in documenting a view, 208
Element list, 217
Elements, 21, 22, 44, 47
clustering of, into Path and Component, 256
extraction of, from UCMEdit, *248*
graphical representation and subset of, *243*
and information extraction, 234, *235*
raw set of, *249*
requirements, in interface documentation, 214
and re-use, 355
elements table, 238, 250
Ellipsis (...), in UML, 222
e-mail, 117
Emergency mode, 85
Encapsulation, 66
Encryption, 117, 332, 343
End demand process, 64
Ending evaluation criteria, 465
End periodic process, 63
End users
changes made by, 81, 82
and Luther architecture, 429–430
Engine control systems, 83
Enhanced Direct Access Radar Channel (EDARC), 137
En route centers, 130, *131*, 133, 147
En route display consoles, and ISSS physical view, 136
Ensembles
model-process work flow and definition of, 468
of off-the-shelf components, 462, 463
Enterprise information systems tier, in J2EE multi-tier model, 408
Enterprise JavaBeans, 345, 346, 351, 352, 401, 425, 443, 482
ABC for, 402, *403*
aims of, 404–405
architectural approach for, 408–410
and dependence on JVM performance, 424
deployment descriptors, 417
and distributed transactions, 423
and J2EE containers, 443–444
package diagram, *413*
programming, 413–416
resource pooling with, 424
and system deployment decisions, 419–420
and work-flow component packaging, 447, 448, 449

Bass.book Page 506 Thursday, March 20, 2003 10:29 PM

Enterprise software systems, requirements for, 404
Entity beans, 410, 443
  and EJB, 409, 410
  using or not using?, 422
  work-flow component packaged as, 447
Entity-relation diagrams, 477
Environment
  and airframe, 193
  in ATAM scenarios, 280
  in CBAM scenarios, 310
  and flight simulator, 178, 182, 183
Errors, 44
  minimizing impact of, 91
  Nightingale system and recovery of, 303
ESI. *See* External System Interface
e-stores, 348–349
European Laboratory for Particle Physics, 328, 329, 332, 334, 346, 347
Evaluate the model solution (step 6), in model problem process, 471–472
Evaluation criteria, 469, 470–471
Evaluation leader
  ATAM and role of, *273*, 277
  in Nightingale system, *289*
Evaluation of model solution, 465
Evaluation phases
  and ATAM, 276
  in Nightingale system, 290–303
Evaluations, importance of, 262
Evaluation teams
  architecture approaches analysis and, 285
  and ATAM, 272, 275, 276, 279
  competent, 267
  and Nightingale system, *289*, 290
  and risk theme identification, 286
  and scenario brainstorming, 284
Evaluators
  and ATAM, 275
  and utility tree, 282
Event-based time management, in flight simulator, 184
Event handler
  in air vehicle model executive, 185, 187
  for flight simulator, 192
Event rate, managing, 113
Events, 82, 83
Event streams, and resource demand, 113
Event systems, subpatterns in, 125
Evolutionary Delivery Development Process, 63
Evolutionary Delivery Life Cycle, *154*, 171, 173
Evolutionary prototyping, and architecture, 31–32
Examples. *See also* Case studies
  ASEILM, 465, 466–476
  garage door opener, 156–166
  reconstruction, 248–257
Exception definitions, in interface documentation, 213
Exception handler, 102
Exceptions, 44
Exceptions tactic, 102, 147
"Excludes" relation, 65
Execution times, bounding, 114
Execution traces, 234
Execution view, 42
Executive constituent, of Structural Modeling architectural pattern, 185
Executive states, of flight simulators, 189
Exemplar systems, and re-use, 356
Existence of A dependency type, 108, 110
Expectations, evaluations and management of, 267
Expected changes, anticipating, 107
Explicit invocation, 125
Exposure, limiting, 117
Extended Computer Module, 56, 57, 62, 63
Extensibility, and the Web, 329
Extension points, 361
External sources, and product line evolution, 365
External System Interface (ESI), 137

FAA. *See* Federal Aviation Administration
Facade pattern, 109
Factory pattern, 109, 110
Failures, 101
  differing standards for, 65–66
  faults *versus*, 79
  within SAS, 140
Fast attack craft (FAC), 371
FAST process, 358, 367
Fat clients, 440
Fault detection, and availability, 102
Fault prevention, and availability, 104–105
Fault recovery
  and availability, 102–104
  and ISSS, 146
Faults, 101
  discovering, 119
  failures *versus*, 79
Fault tolerance, 139
  ISSS C&C hierarchy for, *145*
  mechanisms, 150
  view, 144–147
Fault-tolerant hierarchy, 146

Fault-tolerant ISSS applications, code structure template for, *149*
Federal Aviation Administration, 70, 129, 130, 132
Fidelity, range of and flight simulators, 179
File containment, code segments for, *254*
Files, 234
  in UCMEdit model, 250
File Transfer Protocol, 336
Filter architectural type, 227
Filter Behavior Module, 58
Filter class, 223
Filtering, 118
Filters, and bridges, 459
Financial benefits, from architectural inspections, 263
findByPrimaryKey method, 415
finder method, 415
Finite-state-machine models, 13
Fire control systems, 370, 372
Firewalls, 87, 117, 336, 343–344, 471, 472
First-in/First-out (FIFO), 115
First-order effects, and flight simulator, 193
"Fitness for use" criteria, and component qualification, 460
Fitness of purpose, evaluating software product line for, 362
Fixed-priority scheduling, 115
Flight controls system, in air vehicle model, 195
Flight data, and ISSS physical view, 136
Flight simulation case study, 175–199
  architectural solution, 182–196
  relationship to ABC, 176–177
  requirements and qualities, 177–181
Flight simulators, 70
  and ABC, 176–*177*
  design challenges with, 175
  execution states with, 179–181, 189
  geographically distributed areas, 180
  properties of, 179–181
  purpose of, 177
  reference model for, *182*
  roles in, 177–178
Flight simulator software, 155
Flight strips, 136, 151
Follow-up (phase 3)
  and ATAM, 276
  in Nightingale system, 303
Formal specification languages, 13
Fortran, CGI scripts in, 339
Fortran libraries, 458
Forward-looking radar, 50, 51
"Four Plus One" approach, 41
Frame rates
  for flight simulators, 179–180
  and periodic time management, 183
Frameworks, 478, 482
Front-line workers, computer support for, 427, 429
FTP. *See* File Transfer Protocol
Fuel system, in air vehicle model, 196
Functional group (FG), 139, *140*
Functionality, 72
  allocating in example, 160, 161
  and architecture, 72
  for controller children, 192–193
  and market share, 95
Functional requirements, for child modules in ADD, 164–165
Functional subsets, rapid identification of, 60
Function calls, disambiguating, 241
function_calls_function, 239
Function Driver modules, 58, 62, 65, 66
Function pointers, 236
Functions, 234, 253
Function type, 244
Fused views, items, *240*
Future, in Active Object design pattern, 124

Garage door opener example, 156–166, 480
Garbage collection, and heap size, 424
Gateways, 336
-gcverbose compiler option, 424
Gen++, 235
Generalization, in UML, *221*–222
Generalized capability components, 442
"Generalize the module" tactic, 107, 147
  and code templates, 149
  and module decomposition view, 138
General quality scenarios, generation of, 78
General scenarios, 75, 97
  availability, *76*
  communicating concepts and use of, 93–94
Generate quality attribute utility tree
  in ATAM evaluation phase 1, 279–282
  in Nightingale system evaluation phase 1, 294–295, 297
Generators for systems, 478
Generic utilities, for libWWW, 335, 336
get method, 422
get_outbound_msg operation, 187
Global Availability Management, 144
Global variables, in UCMEdit model, 250
GNU make utility, 248
Good design, promotion of, 265
Gopher, 336

Bass.book Page 508 Thursday, March 20, 2003 10:29 PM

Göteborg class (Swedish), 371, *372*
goto statement, 477
gprof, 236
Graphical images, 478
Graphical notations, in primary presentation, 208
Graphical user interface (GUI) facilities, and ISSS, 135
Graphic artists, and Luther architecture user interface, 436, 437
Graphics, 331, 332, 343
Graphics element, 252
grep, 232, 236
grok fact manipulator tool, 258
Ground control, 130
Group decomposition, in air vehicle model, 193–195
Groups, 195–196
GUI-builder tools, 405

Hackers, 71, 86, 457, 474
Halt state, in flight simulator, 179
Handheld computers, 352, 427
Handheld, pen-based tablets, 433
Hard real time, 129
Hardware-hiding module, in A-7E module decomposition structure, 56
Hash results, 117
has_subclass relation, 240
Header files, 234
Head-mounted displays, 428, 429, *430*
Heads-up displays, 50, 51, 63
Health monitoring, and availability, 101
Heap, 424
"Heartbeat" tactic, 102, 146
Helicopter guidance systems, architecture reconstruction used for, 232
Heterogeneity, and the Web, 330
Hierarchical domain engineering units, and product development, 367
High-bandwidth communication, 167, 168
High-level design, architecture as, 23
Highly distributed software applications, 129
High-order effects, and flight simulator, 193
hold_parameter, 190
home class, 417
Home information system, and example, 157, 166
home interface, 413, *414*, *416*
"Horizontal" scaling (or scaling out), 344, 422, 425
Host Computer System (HCS), 134, 135–136
Hot restart, 103
Hot spots, 343
Human–computer interaction research, 122
Human–computer interface (HCI), 150, 382, 383, 388, 391, 394
Hypertext, 349
HyperText Markup Language (HTML), 158, 329
Hypertext systems, 329
  criteria/features with, 331
HyperText Transfer Protocol (HTTP), 334, 336, 343, 470
  and browser-based interfaces, 439
  and custom Web-based clients, 440
  and load balancers, 344
  and Luther architecture, 435
HyperText Transfer Protocol Secure (HTTPS), 332, 336, 343, 344, 470
Hypotheses, 250, 255
  architectural abstractions as, 245
  in identify a design question, 469
  and reconstruction, 233, 234

IDE. *See* Integrated development environment
Identify a design question (step 1), in model problem process, 468–469
Identify architectural approaches, in ATAM evaluation phase 1, 279
Identify ending evaluation critiera (step 5), in model problem process, 470–471
Identify implementation constraints (step 3), in model problem process, 469
Identity of an interface of A, 108, 110
Idioms, 125
IDL. *See* Interface Definition Language
IEEE. *See* Institute of Electrical and Electronics Engineers
IEEE 802.11b, 435
IIOP (Internet Inter-ORB Protocol), 440
IIS. *See* Internet Information Server
Imagix, 235
Immediate business influence, and architectures, 8
Implementation
  architecture and defining constraints on, 29
  architecture *versus*, 43
  changes made to, 81
  constraints, 465
  interface separated from, 120
  and quality attributes, 73
  structure, 39
Implicit invocation, 8, 125
import operation, 186, 189, 190

includes relationship, 235
"Increase available resources" tactic, 114
"Increase computational efficiency" tactic, 113, 480
Index, 217
Industrial development projects, architectural views in, 43
Inertial measurement set (IMS), 50
Information extraction, 234–237
  guidelines for, 237
  and reconstruction, 233, 248
  typical extracted element and relations, 234, *235*
Information hiding, 16, 42, 44, 45, 49, 66, 159, 168, 477
  and Active Object design pattern, 124
  and A-7E module decomposition, 54–56
  and A-7E Project, 52
  and packaging, 142
  prevention of ripple effects and, 108–109
Infrastructure technology, and CelsiusTech, 379
Inheritance, 219, 477
"Inherits-from" relation, 38
initialize state, 190
Initial Sector Suite System, 130–131
  availability requirement for, 132–133
  C&C hierarchy for fault tolerance, *145*
  functions required of, 134–135
  levels in fault-tolerant hierarchy, 146
  physical view, 135–138
  scale of, 134
  software architecture layers, *143*
  software audit of architecture, 150–151
Inmedius, 352, 427, 428, 435, 443, 452
  ABC as it pertains to, *429*
  and components, 436
  device types and, 433
  library of components at, 442
  Luther architecture, and business approach at, 431–432
  and Luther's quality goal achievement, 450
  solution development at, 430
  and user interfaces, 432
  work-flow design rationale at, 445
Input/output (I/0), managing for testing, 120
InputValue method, 239
Instance-level collaboration diagram, 227
Instances, as objects in UML, *223*
Institute of Electrical and Electronics Engineers, 98
Instructors, simulation, 178
Instructor station, and flight simulators, 182, 183
Integrability, 100
  designing for, 175
  and flight simulators, 180, 196
  Structural Modeling pattern and achievement of, 198
Integral factor of base rate, and flight simulator, 180
Integrated development environment, 345
Integration, 354
Integration mechanisms, 96
Integration testers, 89
Integrators, and ATAM, 272
Integrity, 86, 116
Intelligent data fusion, and mediation, 459
Interaction mechanisms, in architectural patterns and styles, 125
Interchangeability, 33
Interconnection strength, 255
Inter-element communication, 30
Interface Definition Language, 212, 458
Interface documentation template, 212–214, *215*
  data type definitions, 213
  element requirements, 214
  exception definitions, 213
  interface identity, 212
  quality attribute characteristics of, 214
  rationale and design issues, 214
  resources provided, 212–213
  usage guide, 214
  variability provided by, 213–214
Interface mismatch, 457
  avoidance techniques for, 461–462
  detection techniques for, 460–461
  repair techniques, 457–460
Interfaces, 230
  adding, 109
  as annotations, 224
  of child modules, 157
  of child modules, in example, 163–164
  as classes, 225
  as class/object attributes, 225
  documenting, 211–214
  and information hiding, 54
  no explicit representation of, 224
  separating from implementation, 120
  to software element, 44–45
  specification of, 211
  stability, 109
  subsystems, 194
  translation of, 458
  in UML, *219*, 224–225
<<interface stereotype>>, in UML, 212
Intermediaries, using, 109–110

Intermediary (modifiability), and Active Object design pattern, 124
"Intermediary" tactic, 413
Internal monitoring tactic, 120
Internal sources, and product line evolution, 365
Internal Time Synchronization, 144
Internet, 117, 329, 404
  backbones, in United States, *333*
  growth of, 332
  and security, 332
Internet-enabled business systems, requirements of, 403
Internet Information Server (Microsoft), 467, 468, 470, 475
Internet Protocol (IP), 332
Interoperability, 7, 94, 478, 480
  and language-independent interfaces, 58
  World Wide Web case study in, 327–349
Interprocess communication, for SS2000, 390
Inter-product commonality, and software product lines, 354
"Introduce an intermediary" tactic, and SS2000, 391
"Introduce concurrency" performance tactic, 124
"Introduce concurrency" tactic, 114, 345
Intrusion detection systems, 118
Inventory, 345
Inverse mappings
  and extracted information, 244
  and reconstruction, 233
IP addresses, 343
"is-an-instance-of" relation, 38
"is-a" relation, 125, *126*
  and module class view, 219
"is-a-submodule-of" relation, 36, 54
"is-part-of" relation, 208, 219
ISSS. *See* Initial Sector Suite System
is_subclass_of class, 235
IT architecture, 478
Iterative prototyping, 9

Java applets, 343
Java Database Connectivity (JDBC), 435
Java Naming and Directory Interface, 413
Java programming language, 346, 401, 402, 404, 425
JavaScript, 439
JavaServer Pages (JSP), 407, 437, *473*
Java servlet ensemble
  evaluation of, 472, 474–476
  layers of custom component, *474*
Java 2 Enterprise Edition (Sun Microsystems), 351, 352, 401, *407*, 452, 482
  ABC for, 402, *403*
  and aims of EJBs, 404–405
  browser-based clients and, 437
  business needs and creation of, 425
  and component packaging, 443
  deployment view of multi-tier architecture for, *406*
  EJB support for quality attribute requirements of, *411–412*
  and Luther architecture, 428, 431, 434–435, 450–451
  major platform features, 406
  Sun's quality attribute requirements for, *405*
  and system deployment decisions, 419–420
  and transaction and security services, 408
  and the Web, 403–405
Java Virtual Machine (JVM), 402, 408, 424, 443, 448
Jini, 236
Jitter, 84, 85
JNDI. *See* Java Naming and Directory Interface
J2EE. *See* Java 2 Enterprise Edition

"Keeping interfaces small, simple, and stable" tactic, and integrability, 175
Keys, 117, 208
Kinetics, and airframe, 193
Kinetics group, systems in, 195–196
Knowledge and insight, reconstruction process guided by, 258

LAN. *See* Local area network
Landing gear system, in air vehicle model, 195
Language-independent interfaces, 58
Language requirements, and Nightingale system, 291
Late binding, 236, 237
Latency, 84, 85, 111
  and contention for resource, 112
  reducing, 113, 114
Layered architectures, 62
Layered patterns, 279
Layered structure, 37, 44
Layered view, 41, 142, 144, 147, 205, 207
  and architecture presentation, 278
  for SS2000, 392
Layers, 37, 62
LCN. *See* Logical Communications Network
LCN interface units (LIU-H), 137
Learning system features, 90
"Least effort" extraction, 237
"Leftover" functions, in UCMEdit model, 250
Legacy systems, 96, 232, 257, 341

Bass.book Page 511 Thursday, March 20, 2003 10:29 PM

Lexicon analyzers, for information extraction, 235, 236
lex/yacc, 232
libWWW, 329, 330, 334–336, 349
  content producers/consumers interacting through clients/servers, *334*
  deployment view of Web client server, *338*
  early client-server architecture use of, 337–339
  layered view of, *335*
  lessons from, 336–337
Life cycle
  architecture in, 153–155, 481–482
  electronic flight simulators and problems with, 176
  evolutionary delivery, *154*
“Limit exposure” tactic, 144
“Limit possible options” tactic, 107
Links, 435
  connector instances as, 225
  and the Web, 331
Linux system, 258
List class, 254, 255
List element, 255
ListItem class, 254, 255
ListIterator class, 254, 255
List method, 239
Load balancing, 114, 443
  and e-commerce sites, 344
  and EJB container, 444
  and Luther architecture, 434
Loading factors, and arrival patterns, 83
Load time, 110
Local area network, and SS2000, 371
Local Availability Manager, 144, 146
Local changes, 31
Local file systems, 336
“Localize changes” tactic, 159
“Localize modifications” tactic, 106–107
Local variable aggregation, and SQL and perl, *244*
Location independence, and Luther architecture, 434
Location of A (runtime), 108, 110
Location sensing, 434
Locks, and work-flow model definitions, 447, 448, 449
Logging, 336
Logical Communications Network, 137, 138
Logical_Interaction architectural element, 244
  query for identifying, 245
Logical threads, 38
Logical view, 41
Lollipop notation, in UML, 219, 225
Long-term business influences, and architectures, 8
Looping, and work-flow component, 445
Loose coupling, and teams, 168
“Lost” architecture, beginning process of recovering, 246–247
Low-bandwidth communication, between teams, 167
lsedit visualization tool, 258
LSME, 235
Luther applications, deployment view of, *436*
Luther architecture, 427–452
  ABC as it pertains to, *429*–432
  and browser-based clients, 437, 439
  and business approach, 431–432
  and business delegate pattern, 452
  components for, 442–444
  and custom Web-based client, 439–440
  and organizational structure, 431
  quality goals achieved by, 451
  requirements/qualities met by, 432–434
  and software developers’ experience, 431
  strategy and quality goals achieved by, *451*
  and use of J2EE, 450–451
  work-flow component and, 444–450
Luther system, 358

M&C. *See* Monitor-and-Control consoles
mailcap, 338
main procedure, 241
Maintainability, and Nightingale system, 291
“Maintain an audit trail” tactic, 118
Maintainers
  and architectural integrity, 96
  and ATAM, 272
Maintain existing interfaces, prevention of ripple effects and, 109
“Maintain interface stability” tactic
  and client-server view, 142
  and code templates, 149
  and module decomposition view, 138
“Maintain multiple copies” tactic, 114
“Maintain semantic coherence” tactic, 106–107
Maintenance phase, 14
Makefiles, 236
Malfunctions, 191, 193
Manageability, and Web-based applications, 404
“Manage event rate” tactic, 113
Manage input/output tactics, 120
Management information, in documenting a view, 209
Many-to-many mappings, between structures, 40
Mapping between views, 217, 481

Bass.book Page 512 Thursday, March 20, 2003 10:29 PM

Marketing Plan, at CelsiusTech, 383, 384
Market segmentation, and product scope, 358
Marshalling, 112
Measuring techniques, 265–266
Mediator pattern, 109
Mediators, 459–460
Melard Sidearm, 433
Memory footprints, for various device classes, 433
Memory leaks, preventing, 105
Memory utilization, 7
MergeAndSort, in UML, 223, 224
Merging, and work-flow component, 445
Mesa, 250
Message passing, 139
Method request, in Active Object design pattern, 124
Metrics, 265, 269
Microsoft, 347
  Access, 468, 470
  COM+, 425
  development practices at, 173
  and Evolutionary Delivery Life Cycle, 171
  IIS, 467, 468
  .NET, 402
  Passport, 332
Middleware Company, The, 425, 478
"Migrates-to" relation, 38
Migration, and COTS adoption, 476
mimerc, 338
Mining machinery, and architecture reconstruction, 232
Mismatch correction, 457
Miss rate, 85
Misuse detection, 118
MIT, Media Laboratory at, 428
Miva Empressa ensemble, 467, 468–472
Miva Script applications, 467
Mixed initiative, 122
Mixed-time systems, in flight simulator, 184
Mk2, 376
Mk2.5, 376, 377, 378, *380*, 381
Mk3, 377
ML polymorphism, 462
Mobile devices, 439
  and Luther architecture, 437
Modeling and analysis, and re-use, 355
Model problem process steps
  define the starting evaluation criteria, 469
  evaluate the model solution, 471–472
  identify a design question, 468–469
  identify ending evaluation criteria, 470–471
  identify implementation constraints, 469
  produce a model solution, 469–470
Model problems, work flow, 463, *464*
Models
  flight simulators and use of, 179
  and system initiative tactics, 122
Model solution, 463, 465
  evaluating, 471–472
  for Java servlet ensemble, 472
  producing, 469–470
Model-View-Controller pattern, 109, 123, 127
Modifiability, 7, 16, 25, 30, 69, 73, 80–81, 127, 442
  application servers for, 345
  and avoidance of interface mismatch, 461
  and common services, 106
  and component repair, 457
  and e-commerce, 341
  and flight simulators, 196
  performance *versus*, 159
  and software architecture, 201
  and SS2000 product line, 388
  Structural Modeling pattern and achievement of, 198
  and system quality attributes, 74
  Web browsers for, 342–343
Modifiability scenarios, 76–77, 82, *83*
Modifiability tactics, 105–111, 125, 480
  and client-server view, 142
  and code templates, 148
  defer binding time, 110–111
  design-time tactics related to, 121
  localize modifications, 106–107
  and module decomposition view, 138
  prevention of ripple effects, 107–110
Modification, and flight simulators, 180, 181
MODN. *See* Noise Model
MODP. *See* Prop Loss Model
MODR. *See* Reverb Model
Modularity, and Nightingale system, 292
Module-based programming languages, 477
Module-based structures, 36–38
Module decomposition
  goals of, 55
  SS2000, and units of software in, *394*
Module decomposition structure, 40, 41, 47, *48*
  and A-7E system, 66
  and team structure, 168
Module decomposition view, 138, 147, 164
  with ADD, 161
  of A-7E software architecture, *59*
  for SS2000, 392, 394–395
Module guide, 55, 57, 58
Module interconnection view, 42
Module re-use, and CelsiusTech product line, 397

Bass.book Page 513 Thursday, March 20, 2003 10:29 PM

Modules, 35, 36, 97, 477
 of air vehicle model application, 185–192
 and avoidance of interface mismatch, 461
 choosing those for decomposition in ADD, 157
 in decomposition structure, 54
 for flight simulator, 192
 instantiating in example, 160
 and SS2000, 396
 structures, 35–36
 in UML, 219
Module views, 17, 41, 206
 and architecture presentation, 278
 options in, 208
 in UML, 218–222
Monitor-and-Control (M&C) consoles, 137
MRI scanners, 232, 233
Multi-process applications, name clashes in, 241
Multi-processor systems, 170, 236
Multi-threaded servers, 344
MVC pattern. *See* Model-View-Control pattern

Name clashes, in multi-process applications, 241
Names, in ATAM utility tree, 281
Name servers, 110
Naming conventions, code segments based on, 246
Naming services, 401
Narrowly scoped product lines, 358
NASA ECS Project case study
 architectural strategies and scenarios addressed, *320–322*
 assign utility step, 319
 calculate total benefit obtained from architectural strategy step, 322–323
 CBAM applied to, 317–324
 choose architectural strategies based on ROI value step, 323–324
 collate scenarios step, 317
 collected scenarios, *318*
 determine utility of "expected" quality attribute response levels step, 321–322
 prioritize scenarios step, 318–319
 refine scenarios step, 317, *319*
 response goals for refined scenarios, *318*
 ROI of architectural strategies, *323*
NASA satellite data system, ATAM applied to, 305
NASA systems, architecture reconstruction used for, 232
National Airspace System Modification (NASM), in ISSS, 138
National Reconnaissance Office, 354
Natural language, semantic information conveyed with, 213
Naval Business Unit, at CelsiusTech, 384
Naval product lines, SS2000, 371–373
Naval Research Laboratory (NRL), 52, 66
Negotiated interface, 462
Negotiated resolutions, 264
Nesting, 210, *221*
Nesting of symbols, and node symbol in UML, 229
.NET architecture, 402
Netscape Secure Sockets Layer, 343
Network address translation (NAT), 343
Network interface sublayer (NISL), 144
Network News Transport Protocol, 336
Network-related conventions, and COOB, 391
Network usage, 7
NeXT platform, 329
Nightingale ATAM exercise, tabular form of utility tree for, *296–297*
Nightingale system case study, 288–303
 analyze architectural approaches step (phase 1: evaluation), 297–299
 analyze architectural approaches step (phase 2: evaluation), 302
 brainstorm and prioritize scenarios step (phase 2: evaluation), 300–302
 catalog architectural approaches step (phase 1: evaluation), 293–294
 follow-up (phase 3), 303
 generate quality attribute utility tree step (phase 1: evaluation), 294–295, 297
 hospital installation scenario in, 302
 partnership and preparation (phase 0), 288–290
 present architecture step (phase 1: evaluation), 292–293
 present business drivers step (phase 1: evaluation), 290–292
 present results step (phase 2: evaluation), 302–303
 present the ATAM step (phase 1: evaluation), 290
NNTP. *See* Network News Transport Protocol
NobelTech AB, 370
Nodes, 227, 228, 388
Noise Model, 20
Noncentralization, and the Web, 331
Nonintegral factor of base rate, and flight simulator, 180
Nonlocal changes, 31
Non-real-time systems, 6
Nonrepudiation, 86, 116, 118

Bass.book Page 514 Thursday, March 20, 2003 10:29 PM

Non-risks
  and architectural approaches, 282, 283, 284
  and ATAM, 274, 304
  and Nightingale system, 299, 302, 303
Normalization, and architectural strategy, 313
Normal mode, 85
Notation, UML, 219
n-square charts, *194*–195, 198
n-tier applications, 345
n-tier architecture, 341
Numerical analysis, 168

Object-based design, 44, 53, 55
Object containment, and systems, 227
Object Management Group, 112, 230
  and CORBA, 401–402
Object-oriented analysis, 13
Object-oriented decomposition, and controller children, 192
Object-oriented design methods, 160
Object-oriented systems, 401
Object request broker (ORB), 402
Objects, 45, 477, 478
  and allocation views in UML, 228
  connector instances as, 226
  within work-flow model definition, 448
Object technology, and CelsiusTech, 379
Obstacle detection algorithm, 480
Obstacle detection module, in example, 160, 165, 166
ODBC, 435
Off-the-shelf components, 476
  and architectural mismatch, 456–462
  challenges with, 456
  ensembles of, 462, 463
  and Quack.com, 454–455
  requirements negotiation with, 467
  systems built from, 453–476
OMG. *See* Object Management Group
Omission, 80
One-time passwords, 116
Online bookshop example, and state management, 420–422
Online stores, 348–349
OnLine Transaction Manager (OLTM), 298
  communication, data flow, and processors of, *294*
  data flow architectural view of, *295*
  layered view of, *293*
  and Nightingale system, 292
Openness, and air traffic control system, 133
operate operation, 187
operate state, 179, 189
Operating systems, 373, 453
Operating system threads, 162
Operational modes, 85
Operational units, 139, *140*–141
Opportunistic evaluation, 269
Options, 208, 209
Optronic directors, on Göteborg class multi-role corvette, 372
Organizational structure, 8, 32, 70, 171
  and architectural structure, 168, 169
  architecture and dictation of, 29–30
  for CelsiusTech, 380–387
  and Luther architecture, 431
  and system structure, 167
Outputs, ATAM, 285–287
Output values, of avionics software, 63
Overloading, 361
Overload mode, 85
Ownership changes, for CelsiusTech, 378–379

Packages/packaging, 142, 147
  UML, 219, 222, 226
  work-flow components, 447–449
Packet filter firewalls, 344
Packet filters, 343
PAC pattern. *See* Presentation-Abstraction-Control pattern
Paging, and heap size, 424
Parallelism, 38, 97
Parallel systems
  architectures for, 16
  and deployment structure, 39
Parameterized function call, 477
Parameterized interface, 462
Parameterized modules, and CelsiusTech product line, 397–398
Parameter setting, 81
Parsers, for information extraction, 235, 236
Partitioning
  flight simulator sample, 192–193
  and *n*-square charts, 194–195
Partnership and preparation (phase 0)
  in ATAM-based evaluation, 275
  in Nightingale system, 288–299
PAS. *See* Primary address space
PAS failures, and switchovers, 139–140
Passivation
  and scalability, 420
  of stateful bean, 410
Passive redundancy tactic, 103–104
Passwords, 116, 118
Paths, elements related to, 255, 256
Pattern definition and recognition, and reconstruction, 241
Pattern matching, 236

Pattern-oriented software architecture, 42, 127
Patterns, 33, 261
  architectural, 24–25, 232, 242, 279
  and ARM, 258
  business delegate, 452
  in Dali, 242
  design, 259
  and Nightingale system, 290
  and tactics, 100, 480–481
Payload sizes, 118
PDF. *See* Portable Document Format
.pdf files, 337
Performance, 7, 69, 73, 82–84, 100, 171
  and air traffic control system, 133
  application servers for, 345
  databases for, 345
  and e-commerce, 341, 349
  and entity beans, 422
  for flight simulators, 179, 196
  and JVM, 424
  load balancing for, 344
  and market share, 95
  modifiability *versus*, 159
  and Nightingale system, 291
  proxy servers for, 343
  and quality attributes, 74
  and software architecture, 201
  and SS2000 product line, 388
  Structural Modeling pattern and achievement of, 197–198
  and system quality attributes, 74
  Web servers for, 344–345
Performance engineers, and ATAM, 272
Performance scenarios, 83
  generation of, *85*
  sample, *84*
Performance tactics, 111–116, 125
  goal of, 111, *112*
  resource arbitration tactic, 114–116
  resource demand tactic, 113–114
  resource management tactic, 114
Periodic event arrival, 83, 84
Periodic operations, and controller children, 191
Periodic process, 63
Periodic sequencer
  in air vehicle model executive, 185, 186
  for flight simulator, 192
Periodic time management, in flight simulator, 183
perk, 239
perl, 232, 236, 238, 243, 339
  expressions and Dali, 242
  and local variable aggregation, *244*
Perl proxy server, 470, *471*
Persistence services, 401
Personal computers, price/performance ratio for, 402
Personal digital assistants (PDAs), 433
Personnel, and re-use, 356
Philips Elektronikindustrier AB, 370
Philips Research, 258, 358, 367
Physical distribution, 40
Physical Models Module, 58
  of A-7E architecture, 168
Physical Models procedure, 62
Physical view, 41
  and ISSS, 135, *136*–138
Physics simulation systems, and architecture reconstruction, 232
Ping/echo fault detectors, 102
"Ping/echo" tactic, 102, 146
Pipe-and-filter style, in UML, *223*
Pittsburgh Digital Greenhouse, Spot, 433
Planned evaluations, 266
Planning function, with mediators, 459
Platform changes, 81
Platform compatibility, 7
Platform for Internet content selection (PICS), 333
Plug-and-play operation, 110
Plug-ins, 335
Polygon Module, 222
Polymorphism, 111, 236
Portability, 25, 81, 94, 100
  and Java, 402
  with J2EE/EJB, 425
  and language-independent interfaces, 58
  and Luther architecture, 434
  and market share, 95
  and quality attributes, 74
  and the Web, 340
Portable Document Format, 459
Port number, 118
Ports, 224
POSIX, and SS2000, 373
Postconditions, 213
POSTGRES relational database, 238
Post-mortem meetings, and ATAM, 276
PostScript, 337
Preconditions, 213
Prepare BCN Messages, 144
Prepare Messages (PMS) layer, 144
Preprocessor macros, 120
Present architecture step
  in ATAM evaluation phase 1, 277–279
  in Nightingale evaluation phase 1, 292–293

Bass.book Page 516 Thursday, March 20, 2003 10:29 PM

Presentation-Abstraction-Control pattern, 123, 127
Presentation class, 250, 254, 255
Presentation logic, and Java servlets, 472
Present results step
  in ATAM evaluation phase 2, 285
  in Nightingale evaluation phase 2, 302–303
Present the ATAM step
  in ATAM evaluation phase 1, 277
  in Nightingale evaluation phase 1, 290
Present the business drivers step
  in ATAM evaluation phase 1, 277
  in Nightingale evaluation phase 1, 290–292
Preset adaptation data, 147
"Prevent the ripple effect" tactic, 107–110, 159
Primary address space, 139, 148
Primary component, 103, 104
Primary presentation
  in documenting a view, 207–208
  and UML, 218
PrimitiveOp class, 239, 240
PrimitiveOp::Compute, 240
printf function, 248
Prioritization, and utility tree generation, 280
Prioritization strategies, common, 115
Prioritize scenarios step
  for CBAM, 316
  in NASA ECS Project, 318–319
Private interfaces, 461
Private keys, 117, 343
Private links, and the Web, 331
Proactive product lines, and scope, 364
Problems, early detection of, 264
Procedures, processes and systems, and Luther architecture, 432, 433
Proceedings scribe, ATAM and role of, *273*
Process communication, 40
Process enforcer
  ATAM and role of, *273*
  in Nightingale system, *289*
Processes, 138
  for A-7E, 63
  and re-use, 356
process_event operation, 189, 190, 191
Processing deadlines, 84
process_malfunction, 190
"Process monitor" tactic, 105
Process observer
  ATAM and role of, *273*
  in Nightingale system, *289*
Processor groups, 139, *140*
Processors, 139
Processor speeds, for various device classes, 433
Process recommendations, in "good" architecture, 15
<<process>> stereotype, 224
Process structure, 38, 40, 47, *48*
  for A-7E, 63–65, 66
Process views, 41, 138–141, 147, 207
  and element catalog, 208
  for SS2000, 390–392
Produce a model solution (step 4), in model problem process, 469–470
Product architectures, results of evaluation of, 362–363
Product-based organizations, 375
Product line architects, considerations by, 360
Product line architecture, 11, 388
  evaluating, 362–363
Product-line-based organization, 375
Product line development, for SS2000, 369
Product line organizational models, types of, 366–367
Product line pitch, 203
Product lines, 45, 367
  architecture-based, 232
  and CelsiusTech's organizational structure, 380–387
  failure with, 359
  scope of, 357–358
  SS2000 architecture and achievement of, 396
  summary of CelsiusTech's conversion to, 398–399
  and variation, 82
Product line support, for SS2000, 390–392
Product Line Users Group, at CelsiusTech, 384
Product (or structural) recommendations, 15
Product Plan, at CelsiusTech, 383, 384
Products Group, at Inmedius, 431
Profilers, for information extraction, 236
Profiling tools, 236
Program families, 42, 44, 45
Programmatic transaction management, 450, 451
Programming, 33
  and EJBs, 413–416
Programming languages, early, 477
Project decision makers
  and ATAM, 272, 275, 276, 304
  and utility tree, 279
Project glossary, 218
Project managers, 272
  and architecture, 27
  and Nightingale system, 290
Project planning, and re-use, 355

Prop Loss Model, 20
Propulsion subsystem, *196*
Propulsion system, in air vehicle model, 195
Protocols, 118, 214
Prototypes/prototyping
  and architecture, 31–32
  creation of, 13
  and Luther architecture user interface, 437
Provides assumptions, 457, 458
Proxy, in Active Object design pattern, 124
Proxy pattern, 109
Proxy servers, 336, 343
Pseudo-threads, and libWWW, 337
Public keys, 117, 343
Publish-subscribe
  in example, 161
  patterns, 109
  registration, 110
python, 232

Quack.com, 453, 454–455
Qualification, and COTS adoption, 476
Quality, 69
Quality attribute characteristics of interface, in interface documentation, 214
Quality attribute names, in ATAM utility tree, 281
Quality attribute parts, *75*
Quality attribute requirements, 305
  and future impact of components, 483
  for J2EE, *405*
  meaningful generation of, 78
  for typical Web-based applications, *404*
Quality attributes, 14, 16, 42, 69, 99, 480
  and architectural structures, 40
  and architecture, 73–74, 453
  and ATAM, 279, 280
  and Attribute-Driven Design, 155
  and CBAM, 324
  and cross-cutting requirements, 481
  and documentation, 205
  functionality and architecture, 72
  importance of architecture and, 144–145
  and integrability, 175
  and Nightingale system, 292
  and patterns, 25
  and scenarios, 75–78
  and software architecture, 30–31
  stimuli, *94*
  system, 74–75
  and tactics, 100
  understanding, 71–98
Quality-attribute scenarios, 13, 70, 74, 75, 78–94
  availability scenario, 76, 79–80
  generation of, 77
  modifiability scenario, 76–77, 80–82
  performance scenario, 82–85
  in practice, 78–94
  security scenario, 85–88
  testability scenario, 88–90
  usability scenario, 90–93
Quality-attribute-specific tables, 78
Quality attribute utility tree
  ATAM and generation of, 279
  Nightingale system and generation of, 294–295, 297
Quality goals
  A-7E module decomposition structure and achievement of, *56*
  A-7E process structure and achievement of, *64*
  A-7E uses structure and achievement of, *61*
  Luther and achievement of, 451
Quality of service/data provided by A dependency type, 108
Quality requirements, 99
  and architectural drivers, 158
  and Attribute-Driven Design, 156
  mapping of architectural decisions to, 274
Quality scenarios
  and architectural drivers, 158
  and business goals, 155
  for example, 156
  verifying/refining as constraints for child modules, 164, 165–166
Queries, and database models, 237
Questioner
  ATAM and role of, *274*
  in Nightingale system, *289*
Questioning techniques, 265, 266
Questionnaires, 265
Queue sizes, bounding, 114

R&D group, at CelsiusTech, 385
Radar controllers, 133
Radar reports, 134
Radio buttons, 435
Raising/lowering door module, in example, 165, 166
Rate monotonic, and fixed-priority scheduling, 115
Rate Monotonic Analysis, 110
Rationales
  in architecture background, 209
  capturing, 264
  in interface documentation, 214
Rational Unified Process, 41, 43, 155, 173

Bass.book Page 518 Thursday, March 20, 2003 10:29 PM

Reactive model, and product family, 364–365
Reactive systems, and statecharts, 210
Reader's viewpoint, 230
  and architectural documentation, 202
Real-time deadlines, 6
Real-time embedded software systems, 53
Real-time performance, with Structural Modeling pattern, 197
Real-time performance constraints, for flight simulators, 179–180
Real-time scheduling mechanism, 163
Re-architecting tool suite, 258
Reconstructing software architectures, 231–259, 481
  database construction, 237–239
  example, 248–257
  information extraction, 234–237
  reconstruction, 241–246
  reconstruction activities, 233–234
  view fusion, 239–241
  workbench approach, 232–233
Reconstruction, 203, 233
Reconstruction example, 248–257
  database construction, 248–249
  information extraction, 248
  view fusion, 249–257
Reconstruction workbenches, 258
Reconstructors, 233, 244, 245
  and UCMEdit example, 248
Recording, Analysis, and Playback, in ISSS, 138
"Record/playback" tactic, 179
  and managing input/output for testing, 120
Recovery, 101
"Reduce computational overhead" tactic, 113
Reduce demand, 159
Redundancy, 70, 146
  and availability, 101
  and Web-enabled enterprise systems, 423
"Redundancy" tactic, 100
Re-engineering workbench, 258
Reference architectures, 25, *26*, 42, 53
  for e-commerce systems, *341*
Reference model, 25, 42
  for flight simulator, *182*
Refine, 235
Refine scenarios step
  for CBAM, 316
  in NASA ECS Project, 317
Reflective programs, 361
Relational database model, and Dali workbench, 238
Relational databases, 12
Relations
  extraction of from UCMEdit, *248*
  and information extraction, 234, *235*
  raw set of, *249*
Relationship models, *26*
Relationships
  graphical representation of, *243*
  subset of, *243*
  tables, 238
Releases, and Evolutionary Delivery Life Cycle, 154, 173
Relevant views, choosing, 205–207, 229
Reliability, 7, 44
  and market share, 95
  and SS2000 product line, 388
Remote access, and the Web, 329
remote class, 417
remote interface, 413, 414, *416*
Remote method invocation, 113, 402, 440
"Removal from service" tactic, 104–105, 141
Reorganizations, at CelsiusTech, 379
Repair code, classes of, 458
Repair of system failure, 79
Repairs, 101
Replay state, in flight simulator, 179
Replicas, purpose of, 114
Replication, 163
Reports
  after ATAM-based evaluations, 275, 276
  evaluation, 268
Repositories, 109
Repository structure, 38
Requirements, 12, 479
  for garage door opener example, 158
  and re-use, 355
  specification, 6
  understanding, 13
Requirements analysis
  and architecture design, 171
  preliminary, 154
Requirements document, 3, 7, 8
  for A-7E Project, 53
Requirements validation, 264
"Requires assumptions" interface, 457, 458
Research world, success and failure in, 65–66
Resource allocation decisions, and implementations, 29
"Resource arbitration" tactic, 114–116, 159
Resource behavior of A, 108, 110
Resource consumption, 112
Resource contention areas, 15
Resource contention problems, 162
Resource demand tactics, 113–114, 159

Bass.book Page 519 Thursday, March 20, 2003 10:29 PM

Resource management tactics, 114
Resource managers, 110
Resource name, 212
Resource ownership, 108
Resource pools, 424, 425
Resources, contention for, 112
Resource semantics and syntax, in interface documentation, 212–214
Resource usage restrictions, in interface documentation, 213
Response, 80
    in ATAM scenario, 280
    of availability scenario, 80
    CBAM and scenarios in, 310
    jitter of, 84
    of modifiability general scenario, 82
    of performance general scenario, 85
    of quality attribute scenario, 75
    of security general scenario, 87
    tactics for control of, *100*
    of testability general scenario, 90
    of usability general scenario, 92
Response measure
    of availability scenario, 80
    of modifiability general scenario, 82
    of performance general scenario, 85
    in quality attribute scenario, 75
    of security general scenario, 87–88
    of testability general scenario, 90
    of usability general scenario, 92–93
Response time, 112–113
Restrict communication paths, prevention of ripple effects and, 109
"Restrict communication" tactic, 188
Results, Nightingale system and presentation of, 302–303
return function, 248
Return on investment (ROI), 309, 310, 315
    calculating, 313
    CBAM and architectural strategies based on, 316
Re-use/reusability, 44, 353
    and architecture, 351
    and CelsiusTech product line, 396, 397
    and components, 442
    failures with, 356
    of Java components, 405
    and Luther architecture, 444–450
    of modules, 477
    and software architecture, 26, 32–35
    and software product lines, 355
    of work-flow component, 445
Re-use library, 356, 357
Reverb Model, 20
Reverse-direction arrows, 155
Reverse engineering expert, 232
Reviews, forced preparations for, 264
rigiparse, 235
Rigi Standard Form, 233, 238, 258
Rigi tool, 241, 258
Risks, 268
    and architectural approaches, 282, 283, 284
    and ATAM, 274, 304, 309
    and COTS adoption, 476
    and Nightingale system, 299, 302, 303
Risk themes
    and ATAM, 275, 285, 286
    for Nightingale system, 303
rlogin, 336
RMI. *See* Remote method invocation
ROI. *See* Return on investment
Roles, in flight training simulator, 177–178
Rollback tactic, 104
Rollout schedule, 96
Root node, utility of, 279
Rose model, and Nightingale system, 303
Round robin, and dynamic priority scheduling, 115
Routers, for security, 343
RTL/2, 376, 378
Rule-based system, 170
Rules engine, and Nightingale system, 298–299, 303
Runtime, 237
    architectural and interface mismatch and errors in, 457
    binding at, 110
    components, 36
    parameterization, 236
    registration, 110
    structure, 47
Runtime tactics, 121–122
RUP. *See* Rational Unified Process

SAAM. *See* Software Architecture Analysis Method
Safety, and SS2000 product line, 388
Safety-critical systems, 13, 129
Sampling, control frequency of, 113
SAS. *See* Standby address space
Satellites, and architecture reconstruction, 232
Scalability, 30, 94
    application servers for, 345
    databases for, 345
    and e-commerce, 341
    load balancing for, 344

Bass.book Page 520 Thursday, March 20, 2003 10:29 PM

Scalability *continued*
  and Luther architecture, 434
  and Nightingale system, 292
  and passivation strategies, 420
  and Web-based applications, 404
  and the Web, 329
Scaling out (or "horizontal" scaling), 422
Scaling up (or "vertical" scaling), 422
scanf function, 248
Scenario-based techniques, 14
Scenarios, 13, 40, 99, 265
  and ATAM, 280, 304
  ATAM and brainstorming/prioritizing of, 284–285
  CBAM and collating/prioritizing/refining of, 316
  coupling tactics and, 480
  and Luther architecture, 449
  and Nightingale system, 295, 297, *300–301*, 302
  and product line architecture, 362
  quality attribute, 69, 70
  and quality attribute names, 281
  usability, 122
  utility and priorities of, 312
  utility and variations of, 310
  and utility tree, 280
Scenario scribe
  ATAM and role of, *273*
  in Nightingale system, *289*
Schedule estimates, and software architecture, 32
Scheduler, in Active Object design pattern, 124
Scheduler module, in example, 165, 166
Schedules, 170, 354, 355
Scheduling policies, 114–115, 159
  and Active Object design pattern, 124
  and flight simulator, 183
Scheduling theory, 115, 127
Scope/scoping, 354
  controlled, 267
  and product lines, 357–358, 364
SCR. *See* Software Cost Reduction
Screen shots, and Luther architecture user interface, 437
Scribe, 279
Scripts, and bridges, 459
Secondary address space (SAS), 139, 148
Sector suite, controllers at, 133, *134*
Secure communications, between IIS and perl proxy server, 470
Secure Sockets Layer, 117, 332, 343
Security, 7, 25, 69, 85–86, 127, 443
  designer's decisions, J2EE, and implementation of, 450
  and e-commerce, 341
  and EJB container, 444
  HTTPS for, 343
  and Java servlet ensemble, 472, 474
  and Nightingale system, 292
  routers and firewalls for, 343–344
  and software architecture, 201
  and system quality attributes, 74
  and the Web, 332, 340, 349, 404
Security general scenarios, 86–88
Security kernel, 73
Security scenarios, sample, *87*
Security tactics, 116–119, 125
  detecting attacks, 118
  recovering from, 118
  resisting attacks, 116–117
sed, 232
Seeheim pattern, 123, 127
SEI. *See* Software Engineering Institute
Self-configuring software, 462
Semantic coherence, 107, 123, 159
  and CSCIs, 138
  maintaining, 106
"Semantic coherence" tactic
  and code templates, 149
  and separation of concerns with EJB, 417
Semantic constraints, in architectural patterns and styles, 125
Semantic importance, and fixed-priority scheduling, 115
Semantics, 108
Semicolon, 477
send operation, 187
Sensitivity points
  and architectural approaches, 282, 283, 284
  and ATAM, 274, 309
  and Nightingale system, 298, 299, 302, 303
Sensor/actuator virtual machine, synchronization mechanisms for, 162, 165
Sensory inputs, and flight simulator, 180
"Separate user interface" tactic, 123, 435, 442
"Separating interface from implementation" tactic, and managing input/output for testing, 120
Separation of concerns, 41, 45, 55, 417
Sequence, 108
Sequence diagrams, 209, 211
Servant, in Active Object design pattern, 124
Servers, 334
  applications as, *141*
  multi-threaded, 344

Server-side components, EJB, 413
Server tier, processing power increases in, 422–423
Service, 109
  semantics of, 108
  syntax of, 107
Service providers, and ABC for the Web, 346, 347
Service request message, 139
Servlet code, 408
Servlets, 407, 437
Session beans, 409, 443
Session EJBs, and work-flow component packaging, 448
Session state, and Luther architecture, 441
set method, 422
set_parameter, 190
SF300 class multi-role patrol vessels (Danish), 371
“Shadowing” tactic, and process view, 141
“Shadow mode,” 104
“Shadow operation” tactic, 104
Shadow pages, 104
Shape Module, 222
Shared data structure, 38
Shared data view, 207
Shared Services procedure, 62
“Shares-a-secret-with” relation, 54
Shipboard systems, requirements of, 388
Ship System 2000
  applying architecture, 395–398
  architectural goals for, *395*
  history behind product line, 377
  layered software architecture for, *393*
  layered view for, 392
  module decomposition view for, 392, 394–395
  naval product line, 371–373
  organization, *381*–383, *384*, 385
  process view for, 390–392
  product line development for, 369
  staffing, 385–387
  and STRIC, 374
Show multiple views, 122
Side effects, 457
  and architectural strategies, 312–313
Signature(s)
  of elements, 212
  interface, 211, 219
  matching, 211
  and resource syntax, 212
Simplex approach, 103
Simplex architecture for availability, 127
Simulation algorithm, for controller children, 190
Simulation instructor, and flight simulator, 178
Simulation parameters, and controller children, 191
Simulator development, 180
Simulator sickness, 180
Site-specific adaptation data, 147
Size, of flight simulators, 180
Skeletal system, 15, 31, 70, 173
  creating, 170–171
  for flight simulator, 192
SLOC. *See* Source lines of code
SNiFF+, 235
Social influences, and software architecture, 5
Socket connections, and resource pooling, 424
Software
  A-7E, 49
  economic models of, 307, 308
  flight simulator, 181
  self-configuring, 462
Software architects, 25, 97
Software architecture, 19–23, 43, 69. *See also* Architecture Tradeoff Analysis Method; Reconstructing software architectures
  central role of, 201
  and change management, 31
  common structures, *37*
  within configuration management systems, 481–482
  and cost/schedule estimates, 32
  definition of, 3, 21
  and design decisions, 29–32
  documentation as introduction to, 203
  in education, 483
  and evolutionary prototyping, 31–32
  future of, 477–484
  importance of, 26–35
  and product lines, 360
  and quality attributes, 30–31
  relationship between reference models, architectural patterns, reference architecture, and, *26*
  re-use of, 353
  system architecture *versus*, 34
  and template-based development, 34–35
  and training, 35
  as transferable, re-usable model, 26, 32–35
  typical, uninformative presentation of, *20*
  and usability, 91, 92, 97
Software Architecture Analysis Method, 305
Software architecture documentation, fundamental principle of, 230

Bass.book Page 522 Thursday, March 20, 2003 10:29 PM

Software Cost Reduction, 52, 66
Software decision module, 56–57
Software design patterns, 173
Software elements, architecture and definition of, 21
Software engineering, 42
 education, future of software architecture in, 483
 and software product lines, 354
Software Engineering Institute, 268, 399
 ASEILM developed at, 466
 Dali developed by, 258
 practice areas identified by, in fielding product lines, 363
 software architecture Web page, 43
Software engineers, 233
Software interfaces, documentation for, 58
Software life cycle, testing within, 89
Software process, and ABC, 12–14
Software product lines, 170, 351, 353–368, 367
 adoption of, 364–365
 architectures for, 360–363
 common architecture shared by, 32
 and crafting common components, 442
 creating products and evolution of, 365–366
 difficulties with, 363–367
 failure of, 359
 and organizational structure, 366–367
 scoping of, 357–358
 working discussion of, 355–357
Software quality evaluation, 262
Software renovation factories, 258
Software re-use, and Luther architecture, 433
Software structures, 39–40
 unclear mapping between aircraft structures and, 181
Software systems, acquiring, 262
Software testability, 88
Software Utility Module, 58
Solution Group, at Inmedius, 431
Source code, 236
Source lines of code, 376
Source of stimulus
 of availability scenario, 80
 of modifiability general scenario, 82
 of performance general scenario, 84
 of quality attribute scenario, 75
 of security general scenario, 86
 of testability general scenario, 89
 of usability general scenario, 91
Spare tactic, 104, 147
"Specialized access routes/interfaces" tactic, and managing input/output for testing, 120
Special Operations Forces family of trainers, 177
Specification-level collaboration diagram, 227
Spline Module, 222
Splitter class, 223
Splitter role, 227
Sporadic event arrival, 83, 84
Spot (Pittsburgh Digital Greenhouse), 433
"Spot the Architecture," playing, 246–247
Spreadsheets, 12, 478
SQL (Structured Query Language) code, generation of in Dali, *238*
SQL dialect, and Nightingale system, 298
SQL format, conversion of extracted information to, *238*
SQL queries, 243
 and Dali, 242
 and local variable aggregation, *244*
 and recovery of "lost" architecture, 247
SQL Server, 472
SS2000. *See* Ship System 2000
SSL. *See* Secure Sockets Layer
stabilize state, 189
Staff-days, and architectural evaluations, 263
Stakeholder communication, and architecture, 27–29
Stakeholder list, 206
Stakeholders, 201, 229, 479
 and air traffic control system, 133
 and architectural integrity, 96
 architecture reviewed by, 15
 and ATAM, 271, 272, 275, 285–287, 304
 and CBAM, 309
 in CelsiusTech experience, 369, *370*
 and communication needs served by architecture, *204*
 documentation organization for, 216–218
 engagement of, 9
 and future impact of components, 483
 general scenarios and communication with, 93
 helpful architecture documentation for, *206*
 influence on architectures by, 6–8
 and Nightingale system, 289, 300, 301
 and scenario brainstorming, 284
 seasoned and new, 202
 software architecture and communication among, 26, 42
 and utility-response curves, 310, 311, 312
Standards, source of, 172
Standby address space, 139
Standby components, 103, 104
Starting evaluation criteria, 464

Bass.book Page 523 Thursday, March 20, 2003 10:29 PM

Statecharts, 209, 210
Stateful model, and EJB server tier, 420
Stateful session bean instances, clients' static bindings to, *421*
Stateful session beans, 409, 410
  in online bookshop example, 420
Stateful session EJBs, and work-flow component packaging, 448
Stateless model, and EJB server tier, 420
Stateless session bean instances, clients' dynamic bindings to, *421*
Stateless session beans, 409, 414, 417, 425, 443
  advantages with, 420–421
  in online bookshop example, 420
  work-flow component packaged as, 447
Stateless session EJBs, and work-flow component packaging, 448, 449
State management
  and EJB, 420–422
  J2EE and simplification of, 441
"State resynchronization" tactic, 104, 141
Static extraction, 241
Static extractor tool, 239
Static information, 235, 239, *240*
Static libraries, 361
Static scheduling, 115
Static views, 15
Station Manager, 144
Stimuli, tactics and responses to, *100*
Stimulus
  of ATAM scenario, 280
  of availability scenario, 80
  of CBAM scenario, 310
  of modifiability general scenario, 82
  of performance general scenario, 84
  of quality attribute scenario, 75
  of security general scenario, 86
  of testability general scenario, 89
  of usability general scenario, 91
Stochastic arrival, 83, 84
Storyboards, and Luther architecture user interface, 437
Strategy pattern, 109
Stream layer, for libWWW, 336
STRIC, 374
StringOp class, 239
Strong typing, and SS2000, 390, 391
Structural-model-based flight simulators, 176
Structural Modeling pattern, 181, 185–187
  of air vehicle system processor with focus on executive, *186*
  and controller children, 188
  and flight simulation, 176
  goals achieved with, *197*–198
  uses for, 177
Structural rules, for "good" architecture, 16
Structures, 35, 40–41
Stubbing implementations, 120
"Stubbing out" code sections, 170, 171
Stubs, 109
Styles, architectural, 125
Subclassing, 38
Subcontractors, 167
Submodules, 55, 157
Subprograms, 142
Subroutines, 477
Subsets, 62
  delivery of, 63
  of systems, 42
Subsystem controllers, 198
  for air vehicle model, 192
  capabilities of, 189
  for flight simulator, 192
  module, and air vehicle structural model, 187, 188–190
<<subsystem>> package stereotype, 226
Subsystems, 157
  and groups, 193
  integrating, 198
  malfunctions with, 191
  and periodic sequencer, 186
  propulsion, *196*
  skeletal, 171
  and surrogates, 187
  in UML, 219, 226
Success, differing standards for, 65–66
Sun Microsystems, 347
  home page, 425
  Java 2 Enterprise Edition, 401, 404
Supersets, of systems, 42
Supportability, and Nightingale system, 292
<<supports>> stereotype, 228
Surrogates
  in air vehicle model executive, 187
  for flight simulator, 192
Surveillance data, and ISSS physical view, 136
Surveys, 276
Swedish Navy, CelsiusTech contract with, 370, 376
Switchovers, 104
  and PAS failures, 139–140
Symmetric encryption, 117
Synchronization, 40, 70
  and active redundancy, 103
  and passive redundancy, 104
Synchronization points, 162

Bass.book Page 524 Thursday, March 20, 2003 10:29 PM

Synchronization tactic, 100
"Synchronize-with" relation, 64
Syntax, 107
Synthetic interviewing, 434
System administrators, changes made by, 81, 82
System architecture
  and e-commerce, 341
  software architecture *versus*, 34
System design, and tactics, 100
System development, architectural constraints in, 462
System failures, 79, 80
System functions/function groups
  for CelsiusTech, 390, 392
  in SS2000, 394, 395
System Generation Module, 58
System initiative, 122
System overview, 215, 217
System quality, 31
  and architectural quality, 92
System quality attributes, 74–78, 94–95
  component-dominated architectures and achievement of, 463
  problems with previous discussions of, 74
System requirements, and design, 154
Systems, 42, 479
  architectural structures of, *39–40*
  ASEILM, 466–476
  building from OTS components, 453–476
  as collaborations, 227
  concurrency in, 162
  as contained objects, 227
  elements in, 47
  and groups, 193
  in kinetics group, 195–196
  maintaining model of, 122
  projected lifetime of, 95
  qualities of, 74
  in UML, 226–227
Systems Definition Group, at CelsiusTech, 384, 385
Systems of systems, 478
System-specific scenarios, 78, 156
System structure, and organizational structure, 167
System testers, 89

Table-driven operating systems, 12
Tables
  fusing, 241
  in primary presentation, 208
Tactics, 127, 129
  architectural patterns and, 123–124, 125
  availability, 101–105, 125
  coupling scenarios and, 480
  and design, 99
  and integrability, 175
  introduction to, 100–101
  modifiability, 105–111, 125
  performance, 111–116, 125
  predicting results of application of, 480
  security, 116–118, 125
  selecting, 158
  testability, 118–121, 125
  usability, 121–123, 125
  "weaving" of into systems, 481
Targeted markets, 95–96
Task models, 122
Tasks, Ada, 142
TCP (Transmission Control Protocol) flags, 118
TCP/IP, 137, 343, 435
Team leaders
  ATAM and role of, *273*
  in Nightingale system, *289*
Teams, 30
  formation of, 10, 167–170
Technical-Architecture Plan, at CelsiusTech, 383, 384
Technical constraints, and Nightingale system, 291
Technical documentation
  fundamental principle of, 230
  and reader viewpoint, 202
Technical environment, influence on architectures by, 9
Technical influences, and software architecture, 5, 6
Technical Steering Group, at CelsiusTech, 385
Technology changes, at CelsiusTech, 379–380
Technology development, and Luther architecture, 431
Telephone switches, architecture reconstruction used for, 232
TELNET, 336
Telub, 370
Template-based development, and architecture, 34–35
Templates, for documenting interfaces, 212–214
Terminal control area (TCA), 130
Testability, 69, 88–89, 98, 127
  and Nightingale system, 292
  and SS2000 product line, 388
  and system quality attributes, 74
Testability scenarios, 89–90
Testability tactics, 118–121, 125
  internal monitoring, 120
  managing input/output, 120

Test and Training subsystems, 137
Testers, 14
  and architecture, 27
  and ATAM, 272
Test harness, 89, 119, 120
Testing, 354
  cost of with flight simulator, 181
  and re-use, 355
Test plans and cases, 354
Textual primary presentations, 208
Thin clients, 440
Thread model, for Java servlet ensemble, 475
Threads, and resource pooling, 424
Thread safety, and libWWW, 337
Tiers, 341
Time budgets, 15
  with Structural Modeling pattern, 197
Timekeeper
  ATAM and role of, *273*
  in Nightingale system, *289*
Timeline synchronizer
  in air vehicle model executive, 185
  for flight simulator, 192
Time sequences, in sequence diagrams, 211
Time-to-market, 95, 97
Timing/time, 80
  in flight simulator, 183–184
  and performance, 82
Token ring networks, Logical Communications Network composed of, 137
Tomcat
  application server, 472
  and IIS, 475
Tool environment, documentation within, 481
Tools
  and architecture reconstruction, 258
  for information extraction, 235
  and re-use, 356
Tool support
  for architecture reconstruction, 232
  for design process, 481
Top-down adoption, of product line, 364
Top-down approach, for architecture reconstruction, 258
"Top-level architecture," 19
Topological layout, in architectural patterns and styles, 125
Traces, semantic information conveyed with, 213
Tradeoff points
  and architectural approaches, 282, 283, 284
  and ATAM, 274, 309
  and Nightingale system, 299, 302, 303
Training
  architecture as basis for, 35
  and documentation, 203
  with flight simulators, 179, 180
Transaction management, and J2EE, 450
Transaction monitors/processors, 345
Transactions services, 401
"Transactions" tactic, 105
Transaction support, 443
  designer's decisions, J2EE, and implementation of, 449, 450
  and EJB container, 444
Translator library layer, 458
Triggering events, 63
Triple redundancy, 103
TSG. *See* Technical Steering Group
Turbulence, 179
type field, 249
Types, as classes in UML, *223*

UCMEdit model
  after application of common code segments, *255*
  after application-specific direct manipulation, *256*
  after clustering based on application domain, *257*
  classes, files and "leftover" functions shown in, *251*
  graphics subsystem, code, classes, files, and remaining functions in, 252, *253*
  reconstruction of architecture for, 248–257
UI. *See* User interface
Ultrix (Digital), and SS2000, 373
Unauthorized access, protection from, 117
Underwriters Laboratories, 482
undo command, 122
Unified Modeling Language (UML), 201, 218–229
  aggregation in, 219–221
  allocation views, 227–229
  and behavioral descriptions, 209
  C&C views, 222–227
  connectors in, 225–226
  decomposition in with nesting, *221*
  dependency in, 222
  deployment view in, *229*
  detail representation, 223
  generalization documented in, with two line styles, *221*
  interfaces in, *219*
  layers represented in, *222*

Bass.book Page 526 Thursday, March 20, 2003 10:29 PM

Unified Modeling Language *continued*
  module views and, 218–222
  and Nightingale system, 303
  for primary presentation, 208
  systems in, 226–227
Uniform Resource Locators (URLs), 329
United States, Internet backbones in, *333*
Unit tests, 89
Universal Description, Discovery, and Integration (UDDI), 347
UNIX operating system, 142, 298
UNIX processes, Ada tasks mapped onto, 142
Unplanned evaluations, 266
Unstructured evaluations, 269
updateAccount method, 414, *415*
update method, 415, *416*
update operation, 186, 190, 191
  and subsystem controllers, 189–190
Updating, 190
Updrafts, 179
U.S. Air Force, 176
  and simulator design problems, 180–181
U.S. Department of Defense standards, 37
U.S. Naval Research Laboratory, 48
Usability, 7, 25, 69, 73, 90–91, 121, 127
  and market share, 95
  and Nightingale system, 291
  and software architecture, 97
  and system quality attributes, 74
Usability engineers, 123
Usability scenarios, 91–93, 122
Usability tactics, 121–123, 125
  design-time, 123
  runtime, 121–122
Usage guide, in interface documentation, 214
"Use an intermediary" tactic, and surrogates, 187
Use case maps, computer-based editing of, 254
Use cases, 155
  and concurrency in system, 162
  verifying/refining as constraints for child modules, 164
Usenet, 172
Usenet messages, 336
User, maintaining model of, 122
User experience paradigm, and J2EE, 435
User initiative, 122
User input devices, for various device classes, 433
User interaction styles, and Luther architecture, 433
User interface
  and libWWW, 337
  and Luther architecture, 432, 436–437, 439–440
  revising, 123
  separating from rest of application, 123
"User interface agnostic" applications, 441
User interface module, in example, 164, 166
User manuals, 354
User needs, system adapted to, 91
User profile data, 118
Users
  and architecture, 27
  and ATAM, 272
  authorization of, 117
uses relation, 37, 60–61
Uses structure, 37, 40, 42, 44, 47, *48*, 62, 170
  and A-7E system, 53, 60–63, 66
Uses view, 207
Utility
  architectural strategies, 312–313
  and CBAM, 309, 316
  priorities of scenarios, 312
  utility-response curves, 310–312
  and variations of scenarios, 310
Utility-response curves, 310–312, 324
Utility scores, scenarios with (NASA ECS Project), *319*
Utility trees, 155, 282, 284, 305, 309
  ATAM and generation of, 279
  Nightingale system and generation of, 294–295, 297
  tabular form of, for Nightingale ATAM exercise, *296–297*

Validation agencies, 482
Validation of extracted information, 237
Variability guide, in documenting a view, 208–209
Variability provided by interface, in interface documentation, 213–214
Variability techniques, 367
Variables, 234
Variation, 82
  in event arrival time, 85
  in latency, 84
Variation points
  core assets designed with, 354
  identifying in product lines, 360
  supporting in product lines, 361–362
Vasa (Swedish ship), *4*–5
VAX/VMS minicomputers, 378
"Vertical" scaling (or scaling up), 345, 422
View catalog, 215–216
View fusion, 235, 239–241
  and architecture reconstruction, 233
  and database models, 237

Bass.book Page 527 Thursday, March 20, 2003 10:29 PM

disambiguating function calls, 241
and reconstruction, 249–257
Views, 35, 129, 204–205
allocation of, 206, 227–229
architecture represented with, in example, 161–163
choosing, 205–207, 229
combining, 207
C&C, 206, 222–227
documentation across, 215–218
documenting, 207–209
improving, 239–240
mapping between, 217, 481
module, 206, 218–222
prioritizing, 207
View template, 215, 216
Virtual documents, 339
Virtual hardware, 56
Virtual machines, 37, 62, 159
Virtual private network (VPN), 117
Virtual threads, 162, 163
Viruses, 86
Vision, 201
Visual Basic, 339, 340, 405, 455
Visualization and interaction, and reconstruction, 241
Visual Mining, 468
Voice input, 435
Votes
refined scenarios with (NASA ECS Project), *319*
and scenario brainstorming, 285
Voting tactic, 102

Warm restarts, 103, 146
Weapons control, and system function groups, 394
Wearable computing, 352, 452
devices, 433
history of, 428
Wearable Project, at Carnegie Mellon University, 427, 428
Weather patterns, modeling, 179
Web-based applications, quality attribute requirements for, *404*
Web-based financial systems, 83
Web browsers, 329, 342–343
Web-enabled enterprise systems, distribution and scaling issues with, 422–423
Web pages, 167
Web servers, 332, 342, 344–345
Web server styles, comparison of, 349
Web services, 117
Web sites, successful, 404
Web site software, and client surges, 404
Web spiders, 335
Web tier, in J2EE multi-tier model, 407
White noise, *249*
Wide Area Information Server (WAIS), 336
Widgets, 97
Windowing systems, 12
Windows-based platforms, and .NET, 402
Windows CE PC devices, handheld, with pen and keyboard, 433
Windows 98 and NT 4.0 operating system, 468
Windows NT platform, 298
Wireless access, and Luther architecture, 432
Wireless Application Protocol (WAP), 437
Wireless Markup Language (WML), 437
Wireless technology, 352
Work assignment structure, 39
Work assignment view, 167
Workbench approach, 232–233, 238, 258
Work breakdown structure, 29–30, 167
Work flow, 345
Work-flow component
class diagram for, *446*
design rationale for, 445–447
distributed and detached operations and, 449–450
and Inmedius, 442
and Luther architecture, 444–450
packaging for, 447–449
Work-flow management, and Luther architecture, 427, 435
Workflow Management Coalition, 452
at Inmedius, 445, 446
Work-flow model definition, 446
Work practices, 167
World Wide Web (WWW), 12, 268, 327–349, 479
and Amazon.com, 348–349
current ABC for, 346, *347*
growth of, *330*, 332
initial quality goals achieved by, *340*
and J2EE, 403–405
original ABC for, *328*
original requirements for, 330–332
requirements and qualities for, 329–332
success of, 348
World Wide Web Consortium (W3C), 334, 346
Reference Library, 349
Worms, 86
Worst-case quality attribute level, in utility-response curve, 312

Bass.book Page 528 Thursday, March 20, 2003 10:29 PM

Worst-case response, for NASA ECS Project
  scenario, 317
Wrappers, 458, 462
Wrapper session beans, 422
"Write Once, Run Anywhere," 408
XForms, 250
Xlib, 250
XML (eXtensible Markup Lnnguage), 343,
  417