Symbols
3-Card Monte, 113

A
abnormal events, 302
abort(), 355
acceptance testing, 46, 68, 70
access control lists (ACL), 142
access methods, 149
access rights, roles, 275
accessing, trustworthiness of sensitive attributes, 207
accountability, 14-15
security, 18
ACL (access control lists), secure resource containers, 142
actionability, 30
actionable error messages, 30
actions, enforcing one at a time, 277
ad-hoc tests, 46
adding derived requirements, writing reliability requirements, 25-29
ADL (architecture description language), 54
administrative functions, risk, 81
administrative processes as business processes, 162
administrative resources, 254
administrative roles, 270
agent clients for monitoring failure response, 530
aggregation, 318
transactions, 403-406
agile methodologies, 537
agile tests, 46
alert messages, 456
alerting users, 309-310, 514-515
algorithms, 312
concurrency, 90
incremental algorithms, 89
allocating capacity, 304-305
allocation, 304, 324-325. See also deallocation
alternative authentication, 291
analysis, static analysis, 52
architectural rules, 60
anonymity, 17
AOP (aspect-oriented programming) languages, 51
APP (application protection proxy), 300
application consoles, failure response, 528, 530
application protection proxy (APP), 300
application roles, 270
application security case study, 575-577
authorization rules, 578-579
contextual data, 577
credential management, 579
data classification, 578
attacks, 93
script injections, 107
social engineering. See social
engineering
technical attacks. See technical attacks
time bombs, 107
TOCTTOU, 99
trojan horses, 107, 253
attributes, policy attributes, 165
auditability, 153-154
business rules, 469
authenticated identities, 127
authentication, 17, 289, 306, 314
alternative authentication, 291
context, 203-204
exploits of incomplete
authentication, 105
authorization model, resources, 176-178
authorization, 121-124, 259-263
caching data, 263
code that performs authorization, secure
error handling, 187-188
exploits of incomplete
authorization, 105
policy repositories, 175
usability, 313
Authorization as a Precondition pattern,
261-263
authorization rules, 176
application security case
studies, 578-579
auditability, 153-154
automated reclamation, 323
avoiding resource interference, 447-448
B
back doors, 228
backend data, 285
backup systems, secure error handling, 194
bail bond scheme, 114
bank examiner frauds, 309
bankruptcy fraud, 113
batch processing, 345
Beales, Howard, 18
behavioral tests, 45
  specifications, 47
beta tests, 46
bi-directional dependencies, static objects, 330
binding resources, 248
BITs, 20
black box tests, 46
blind callbacks, 247
blocked registrant, request-response, 438-440
bottom-up tests, 46
boundaries
  privileged context
  boundaries, 220
  scoping boundaries, Java, 226
boundary condition tests, 46
brute force guessing, 108
buffer overflow, 181-182
build-time closures, 258-259
Business Delegate pattern, 414
business objectives, writing reliability requirements, 23
business rule errors, 2
business rules
  auditability, 469
  simplifying, 469
  versus security rules, 162
business workflow, 160-161
business processes, 162
canonicalization effects, 250-251
canonicalizing names, 250-251
capability, secure resource containers, 141-142
capacity, allocating, 304-305
capturing implicit context, 457-458
case studies
  application security, 575-577
    authorization rules, 578-579
    contextual data, 577
    credential management, 579
    data classification, 578
    defense-in-depth, 577
    logging, 579
    remediation, 580-581
    revocation of permissions, 579
  manageability, 583-584
    analysis, 584
    problems found, 585-588
    remediation, 588-589
  transactional integrity, 567, 569-570
    analysis, 570
    failure testing, 572
    multithreading, 572
    remediation, 572-574
    resource-level interfaces, 570-571
    web application transaction encapsulation, 571-572
  catch and log, 491-493
  certificate authority (CA), 289
certification, 20
certification testing, 46, 50-51
channel errors, 437
checked exceptions, 481
lifecycle errors, 483-486
checkpoints, 90
circumvention, 103
clearance levels, 268
clicks
  double-click problem, concurrency, 364-367
  preventing multiple clicks, 365
Client Tier MVC pattern, 362-363
client workflow, 160-161
client-side scripting, HTTP parameters, 299
cloning risks, 323
closure, 226, 237
    build-time closures, 258-259
    reusable components, 239-242
Closure of Re-Usable Components pattern, 240-241
code
    mobile code, 211
    verifying against architectural rules, 58
    writing privileged code on method policy platforms, 223-225
    XP, collective code ownership, 550
code context, 209, 211-212
callbacks, 210
code generation, 62
templates, 63
    versus patterns, 64-65
code reviews, design verification and elaboration, 43-44
collusion, 117
    insider collusion, 118
combining transactions, 396
    kinds of transaction composition, 396-398
    pre-existing transactions, 398-400
    process-specific schemas, 406-407
    transaction aggregation, 403-406
    transaction layering, 401-402
communication, secure communication in multi-role operations, 276
communication pathways, controlling, 225-227
    component-oriented design, 227-230
compartamentalization, 268-269
    role separation, 272
compartamentalized, log
    access, 195
compartamentalizing roles, 274
compartments, 267
compatibility tests, 46
compilers, 62
complexity, 84
complexity of components,
    reliability-related risk, 82
compliance with design,
    verification, 38
Component and Port pattern, 229
component responsibilities, 333
component-oriented design, controlling communication pathways, 227-230
components, 228
    external components, 336
    internal states, 336
    minimizing accessibility of sensitive components, 87
    release components, configuration management of, 559-560
    reusable components, 238-242
    roles and responsibilities to consider, 334
composability, 238, 448
composable transactions, 399
composite object transactions, 392-394
composite objects, deleting, 326
Composite of Pre-existing Transactions pattern, 400
Composite pattern, 319
composite transactions, 396
    combining pre-existing transactions, 398-400
composition, 238, 319
    value objects, 320-321
compositional relationships, 319
comprehensive testing, 65
concurrency, 90
    common GUI concurrency problems, 361-363
    remote model, 362
    double-click problem, 364-367
    multiprocessing, 343
multithreading, 343-344
  coordination of error response, 347-348, 350
design patterns, 350-355
designing thread timeout mechanisms, 359
process coordination, 344-345
thread interruptions, 355-359
patterns, 341-343
problems with Web scripting languages, 345
testing and validation, 368-376
Concurrency Test Driver pattern, 371-372
concurrency testing, 49
concurrency tests, 46
Condition Logging and Problem Notification pattern, 519-520
conditions, dynamic conditions, 169-174
confidentiality, 17
configuration processes, manageability, 446
configurations, secure
  configuration data, 254
connections
  dynamic connections, 252
  point-to-point dedicated connections, 252
  resource connections, 244-247
consistency across transactions, 385-390
  intra-transaction consistency, 390-392
consoles
  application consoles, failure response, 528, 530
  operator consoles, failure response, 526-528
constraining implementations, designing for verifiability, 86-87
constraints
  designing for verifiability, 85
  roles, 271
container-level resources, 157

containers
  secure resource containers. See secure resource containers
  static lifetime objects, 330
containment failure, 105
Contest Winner, 113
context, 202-203
  authentication, 203-204
  authorization decision context, identifying, 205-208
  capturing implicit context, 457-458
  code context, 209-212
  control context, 217-218
  embedding context mechanisms, 500
  execution context, 218
  explicit context, 203, 501-502
  identity context, 209
  implicit context, 203, 501-502
  providing context, error handling, 497-498, 500, 502
context binding, 203
context data, validating, 212-214
context propagation, 216-217
contextual data, application security case studies, 577
contextual scope, 214-216
controlling communication pathways, 225-227
  component-oriented design, 227-230
convenience versus safety, role separation, 276-278
code, authorization (secure error handling), 187-188
cookie poisoning, 284
cookie snooping, 284
cooking the books, 118
Cooperative Checkout pattern, 409-411
Cooperative Process Coordination, 346-347
coordination interference, 100
coordination of error response, concurrency, 347-348, 350
corporate assurance guidelines, writing, 557-558
correct core processing, 24
correctness proofs, 52
correlating user actions with server actions, 498
cost
  of failure, 83
  of technical problems, 6
counterfeit credentials, 116
Country Boy, 113
coverage, test suites, 45
covert channels, 96
crashes, forced crash and retrieval of crash artifacts, 100
credentials
  counterfeit credentials, 116
  managing, 286-287
    application security case studies, 579
    best practices, 287
    credential distribution, 289-291
    expirations, 288
    PKI processes, 288
  stolen credentials, 116
crisis, 112
crisis-based pressure, 114

D
DAOs (data access objects), 383
data
  backend data, 285
  context data, validating, 212-214
  embedded data, 284
  storing
    securely, 283-284
    unsecured data in secure locations, 282
  transporting, 282
data access objects (DAOs), 383
data classification, 269
  application security case studies, 578
data classification effort, 285
Data Identification pattern, 507-511
data-driven tests, 46
databases, relational databases, 393
deallocation, 324
dynamically allocated objects, 324-327
  static lifetime objects, 329-331
debugging, 515-516
decision context, 205-209
Decorator pattern, 319
decoys, 114
default policies, 174-176
defense-in-depth, application security case studies, 577
delay, 98-100
delaying affect of sensitive operations, 277
Delegating Lifecycle Manager pattern, 331-333
deleting composite objects, 326
denial of service attacks, 99, 303-307
deployment, 295, 563
designing for secure deployment, 293-296
  patches, 563
  platform-level configurations, 296
  trusted components, 294
derived requirements, 538
design assurance, 9
design for testability (DFT), 84
design languages
  elaboration, 54
design patterns. See patterns
design reviews, design verification and elaboration, 43-44
data classification effort, 285
data classification, 269
application security case studies, 578
development, 205-209
design-oriented programming, 53-55
dynamic verification, 51
human intervention, 43-44
intentional languages, 52
static analysis, 52
testing, 36, 45-47
behavioral test specifications and regression testing, 47
certification testing, 50-51
failure-response testing, 48-50
hardware design, 37
stress testing, 48-50
TDD, 47-48
design-by-contract, 51
design-oriented programming, 53-55
designing
for secure deployment, 293-296
secure operations, 297-302
security-related features, 310
thread timeout mechanisms, 359
for verifiability, 86-87
designs versus implementations, 39
desktop computing platforms, 298
destination realm, 129
detection, 30
detection and response, 17
DFT (design for testability), 84
discriminating requester, specialized services, multithreading, 352-355
disposing of data securely, 257
distractions, 115
diversion, 114-115
diversity, 150-152
DNS (Domain Name Service), 249
documentation, 466-468
universal methodological principles, 545-546
XP, 550
Domain Name Service (DNS), 249
domain-specific languages, 54
domains, 128, 231
information sharing across domains, 236
layers, 232-235
double-click problem,
concurrency, 364-367
DREAD, 19
DrinkorDie Web piracy group, 118
duties, enforcing separation of duties, 274-275
Dynamic Condition Set pattern, 171-174
dynamic conditions, 169-174
security managers, 173
dynamic connections, 252
dynamic verification, 51
dynamically allocated objects, 324-327
E
eavesdropping, 96
Eifel, 51
EJBs (Enterprise Java-Beans), 248
elaboration, 42-43
assertion languages, 52
correctness proofs, 52
design languages, 54
dynamic verification, 51
human intervention, 43-44
intentional languages, 52
static analysis, 52
testing. See testing
elevated privilege, obtaining objects, 225
embedded attacks, 106-107
embedded data, 284
embedded passwords, 283
embedded progress interpreter, patterns, 454-455
embedding
context mechanisms, 500
objects, 318
embezzlement, 118
emotional investment, 112
emotional pressure, 112-113
employee account fraud, 118
employing different credentials for each role, 278
encapsulation of privilege, 218-220
approaches, 220-221, 223
encryption, deciding what to encrypt, 284-286
end-to-end scenario, exception handling patterns, 521-523
end-to-end tests, 46
enforcing separation of duties, 274-275
Enterprise Java-Beans (EJBs), 248
enticement, 112-113
entitlement, 123
secure resource containers, 148
environmental interference, 101
erasing sensitive data, 257
error handling, 479-480
alerting users, 514-515
catch and log, 491-493
debugging, 515-516
ensuring all situations are handled, 480-483, 486-490
GUI, 492
information needs of other stakeholders, 511-512
iteration failures, 493-494
logging, 516-519
monitoring, 478
nested error handling, 496
notification requirements, 516-518
notification system design, 519
providing context, 497-498, 500, 502
providing sufficient information, 503-507, 511, 513
robust error handling, 495-496
secure error handling, 185-186
backup systems, 194
code that performs authorization, 187-188
propagation of unauthorized information, 188-194
transactions, 415-418
user status, 515
error messages, 3
error response, coordination of error response, 347-348, 350
errors
actionable error messages, 30
business rule errors, 2
channel errors, 437
initialization errors, 449
lifecycle errors, checked exceptions, 483-486
mapping to problematic input, 505-507
recoverable errors, 24
events, 302
evidence-based authorization, secure resource containers, 147
Exception Filter pattern, 189-190
exception handling, 511
exception handling patterns, end-to-end scenario, 521-523
exceptions, 490, 512
checked exceptions, 481
discarding information, 503
filtering sensitive data from, 189
manageability, 513
execution context, 218
exhaustive search attacks, 108
existence, architectural rules, 57
expiration, credentials, 288
explicit context, 203, 501-502
exploitation of non-atomicity, 99
exploits
of exposure of internal states, 105-106
of incomplete authentication or authorization, 105
of incomplete traceability, 104
of incomplete validation, 105
of residual artifacts, 106
extensibility, 470-471
extensible roles, 271-272
extensible-grammar
languages, 54
extensions, 471
external components, 336
external systems, 336
Extreme Programming. See XP
F
Factory pattern, 325
factory-created instances, 326
factory-created objects, 326
failure
causes of application software failure, 2
cost of failure, 83
system component failures, 30
failure modes, problems, 5
failure response, 14, 475-476
error handling. See error handling
monitoring and management, 525-526
agent clients, 530
application consoles, 528, 530
operator consoles, 526-528
problems with, 476-478
writing test plans, 71-74
failure response requirements, 15
failure scope, indicating, 481-483
failure testing, 67, 70
transactional integrity case study, 572
failure, recovery tests, 46
failure-response requirements
samples, 617-620, 622
writing, 29-31
failure-response testing, 48-50
false claims, 115
fault tolerance, 24
Federal Trade Commission (FTC), 18
fictitious refunds, 118
fictitious sales, 118
FileChannel.lock(), 346
filtering sensitive data from
exceptions, 189
firewalls, P2P, 437
forced crash and retrieval of crash
artifacts, 100
forced re-install, 100
forced restart, 100
form tampering, 284
forward cache, 426
with synchronous updates, 428-429
foundation patterns, 64
framework patterns, 64
FTC (Federal Trade Commission), 18
functional languages, 55
functional requirements, 14

G
Ganssle, Jack, 6
gateways
secure resource containers, 140-141
weak links as, 108
general purpose types, 335
generating progress and health messages
capturing implicit context, 457-458
heartbeats, 451, 453
progress bars, 458, 461-462
progress events, 453-455
trustworthiness of status indication, 456-457
GetLock() method, 346
ghost, 118
glitches, 2
good directions, 311
GUI toolkit, 362
guidelines
for instructional messages, 308
writing corporate assurance guidelines, 557-558
GUIs (graphic user interfaces), 5
common concurrency problems, 361-363
error handling, 492
sound GUI applications, problems, 5

H
hackers, 282
Handling Transaction Timeout pattern, 417-418
hardware, testing, 37
haste, 113
HDM (Hierarchical Development Methodology), 53
health messages, 450
capturing implicit context, 457-458
heartbeats, 451, 453
progress bars, 458, 461-462
progress events, 453-455
trustworthiness of status indication, 456-457
Heartbeat Signal, 452
heartbeats, 451-453
Hierachical Development Methodology (HDM), 53
hierarchies
  resource hierarchies, 132
  of trust, 130-133
high-risk components, 81
hijacking, 103
HTTP, 300
  client-side scripting, 299
Hughes, Ron, 78
human intervention, design verification and elaboration, 43-44
HyperJ, 52

identifying
  decision context, 205-206, 208
  quality-critical codebase, 543
  release components, 559
  resources, 248
identities, 127
identity context, 209
identity theft, 273
Interruptible Thread pattern, 357-359
implementations, 34-35
  constraining, designing for
    verifiability, 86-87
    versus designs, 39
    transparency, 468-470
implementing
  design, 34
  single sign-on, 278
implicit context, 203, 501-502
inconsistencies, 313
inconvenience, 112
incremental algorithms, 89
indicating failure scope, 481-483
ineffectiveness, 115
information services, 299
information sharing, across domains, 236
initialization errors, 449
input arguments, 335
Input Map pattern, 505-507
inputs, validation of, 335
insider collusion, 118
installation, manageability, 447
instances, factory-created instances, 326
instructional messages, guidelines for, 308
Integer, 318
integration system tests, 46
integration testing, 67
integrity
  domains. See domains
    referential integrity, 381-383
    transactional integrity, 379-381
intentional languages, 52
intentional programming, 53
interception, 96
interfaces
  resource interface design, 178-181
    simplicity, 183
    type safety, 181-183
  resource-level interfaces, transactional integrity case study, 570-571
interference, 129, 236
  coordination interference, 100
  environmental interference, 101
internal states
  components, 336
  exploits of exposure of internal states, 105-106
interrupts, 98, 342
intimidation, 112
intra-transaction consistency, 390-392
intrusion detection, 303
intrusion response, 307
intrusion-aware design, 88
isolation, 340-341
  SQL isolation levels, 384
  transactions, 383-385
issues, methodologies, 536-540
iteration failures, error handling, 493-494
J
Java
  object references, 324
  peekEvent(), 365
  scoping boundaries provided by, 226
JML, 51
Jtest, 58
K
Kaleidoscope, 110
Karger, Paul, 268
Kerberos key distribution protocol, 129
key turn, 276
keys, PKI processes, 288
kickback, 118
kiting, 115
known identities, 127
L
languages
  ADL, 54
  AOP languages, 51
  assertion languages, 52
  design languages, 54
  design-oriented programming, 53-55
  domain-specific languages, 54
  extensible-grammar languages, 54
  functional languages, 55
  intentional languages, 52
  modeling languages, 40-42
  template definition languages, 54
VHDL, 228
Web scripting languages,
  concurrency, 345
layer access rules for transactions, 411-414
Layer Interface pattern, 235
Layered Transaction pattern, 402
layers
  domains, 232-235
  resource definition layer, 150
  levels of processes within organizations,
    156-157, 159-162
  levels of verification, 65-66, 70
  testing, 66, 68
lifecycle errors, checked exceptions, 483-486
lifecycle factory mechanisms, 326
Lifecycle Factory pattern, 327-329
load tests, 45
log design, 196-201
Log Monitor, 199
log requirements, 195
logging, 189, 450
  alert messages, 456
  application security case studies, 579
  catch and log, 491-493
  Condition Logging and Problem
    Notification pattern, 519-520
  error handling, 516-519
  secure logging, 194-195
    log design, 196-201
    log requirements, 195
logon processes, 304
logs, recycling, 198
lookup services, threat analysis, 250
M
macker, 52, 59, 61
maintainability, 314-315, 465
  documentation, 466-468
  extensibility, 470-471
  regression testing, 472
  transparency of implementation,
    468-470
maintenance, messaging, 442
man-in-the-middle (MiM) attacks, 96
manageability, 445
  avoiding resource interference, 447-448
  case studies, 583-584
    analysis, 584
    problems found, 585-588
    remediation, 588-589
  composability, 448
  configuration processes, 446
  exception base class, 513
  installation, 447
managed resources, 448
management, 17
  failure response, 525-526
    application consoles, 528-530
    operator consoles, 526-528
manager/worker pattern, 351
managing
  credentials, 286-287
    best practices, 287
    credential distribution, 289-291
    expirations, 288
    PKI processes, 288
  object lifecycles, 327
  trust, 128
    hierarchies of trust, 130-133
    security realms, 128-130
manual testing, 71
mapping, 64
  errors to problematic input, 505-507
  relational mapping, 392
MDA (Model Driven Architecture), 62, 467
memory leaks, 324
message-based paradigm, 433, 440-443
maintenance, 442
messages, alert messages, 456
messaging, 440-443
  maintenance, 442
method policy platforms, writing
  privileged code, 223-225
methodologies
  agile methodologies, 537
  issues, 536-540
  mutual exclusion of roles, 274
  principles of, 540-542
    architecture that explains how
    application meets assurance
    requirements, 544
    auditable business rules, 542-543
    documentation, 545-546
    full configuration manage
    ment of all artifacts, 547
    identifying quality-critical
    codebase, 543
    quality of implementation, 544
    reviewing design, 546
  risk evaluation methodologies, 19
  XP, See XP
methodology-based requirements, 22
methods
  abort(), 355
  FileChannel.lock(), 346
  GetLock(), 346
  peekEvent(), 365
  ReleaseLock(), 346
  renego(), 355
  ResumeTask(), 346
MiM (man-in-the-middle) attacks, 96
  minimizing
    accessibility of sensitive
    components, 87
    the trusted computing base, 81
mobile code, 211
model authorization, permissions, 164
Model Driven Architecture (MDA), 62
Model-View-Controller (MVC), 361
modeling trust, 125
modeling languages, 40-42
  UML, 41
modification in place or in transit
  attacks, 97
monitorability, 30
Monitored Log Writing System pattern, 198-201
monitoring, 302-303, 450
  application-level monitoring, 307
  error handling, 478
  failure response, 525-526
    application consoles, 528, 530
    operator consoles, 526-528
    use of agent clients, 530
  thread status, 494-495
monitors, 343
MOPS, 52
multithreading, transactional integrity
  case study, 572
multiple roles, 273
Multiple-Transaction Request pattern, 404-406
multiprocessing, 343
  coordination of error response, 347-348, 350
  process coordination, 344-345
multithreading, 343-344
  coordination of error response, 347-348, 350
  design patterns, 350
discriminating requester, specialized services, 352-355
pipeline, 351-352
requester/servicer, 351
designing thread timeout mechanisms, 359
process coordination, 344-345
thread interruption, 355-359
worker threads, 347
mutable objects, 318
mutex, 342
  batch processes, 345
mutual exclusion of roles, 274
MVC (Model-View-Controller), 361

N
naiveté, 115
name resolution, threat analysis, 250
names, canonicalizing, 250-251
namespace attacks, 108
National Institutes of Standards and Technologies (NIST), 144
native resources, 149, 255-256
need to know strategies, 268
negative invoicing, 118
negative tests, 46
nested error handling, 496
Netegrity Siteminder, 87
Neumann, Peter, 77
Nigerian Letter, 113
NIST (National Institutes of Standards and Technologies), 144
  non-atomicity, 99
  non-coomposability, 399
  non-functional requirements, 14
  non-repudiation, 104
  non-synchronous pipeline, 352
  non-transactional resources, 394-395
  normal events, 302
  notification requirements, error handling, 516-518
  notification system design, error handling, 519

O
object definition, 333
Object Management Group (OMG), 62
object references, Java, 324
object responsibilities, 333
object-level scoping, 227
object-oriented design (OOD), 236
objects
  composite objects, deleting, 326
  dynamically allocated objects, 324-327
  embedding, 318
  factory-created objects, 326
  lifecycles, managing, 327
  mutable objects, 318
  obtained via elevated privilege, 225
  static lifetime objects, 329-331
  requirements for creation and destruction, 330
  value objects, 320-321
OMG (Object Management Group), 62
OO applications, problems with relational databases, 393
OOD (object-oriented design), 236
operator consoles, failure response, 526-528
operators, 30
Optimistic Concurrency Control pattern, 388-390
organizations, levels of processes within, 156-157, 159-162

P
P2P (peer-to-peer), 437
parameter tampering, 284
Parameterized Permissions pattern, 166-168
parameters
- permission parameters, 164-168
- roles, 169
Parasoft Jtest, 52
passwords, 273, 290
  - embedded passwords, 283
patches, 563
path, branch tests, 45
paths, static resource paths, 252
pathways, communication pathways. See communication pathways
pattern matching, 108
patterns
  - Application-based Propagation of Client Context, 498-499
  - approval-based trust, 126-127
  - Asynchronous Error Handler, 349
  - atomic queues, 342
  - Authorization as a Precondition, 261-263
  - blocked registrant, 438-440
  - Business Delegate pattern, 414
  - caching, 426
    - forward cache, 428-429
  - Client Tier MVC, 362-363
  - Closure of Re-Usable Components, 240-241
  - Component and Port, 229
  - Composite of Pre-existing Transactions, 400
  - Composite pattern, 319
  - Concurrency Test Driver, 371-372
  - Condition Logging and Problem Notification, 519-520
  - Cooperative Checkout, 409-411
  - Cooperative Process Coordination, 346-347
  - Data Identification, 507-511
  - Decorator pattern, 319
  - Delegating Lifecycle Manager, 331-333
  - derived roles, 271-272
  - Dynamic Condition Set, 171-174
  - Embedded Progress Interpreter, 454-455
  - Exception Filter, 189-190
  - extensible roles, 271-272
  - Factory pattern, 325
  - foundation patterns, 64
  - framework patterns, 64
  - Handling Transaction Timeout, 417-418
  - heartbeats, 452-453
  - Input Map, 505-507
  - Interruptible Thread, 357-359
  - interrupts, 342
  - Layer Interface, 235
  - Layered Transaction, 402
  - Lifecycle Factory pattern, 327-329
  - Monitored Log Writing System, 198-201
  - monitors, 343
  - Multiple-Transaction Request, 404-406
  - multitreading, 350
  - discriminating requester, specialized services, 352-355
  - pipeline, 351-352
  - requester/servicer, 351
  - MVC, 361
  - Optimistic Concurrency Control, 388-390
  - Parameterized Permissions, 166-168
  - Pessimistic Concurrency Control, 386-388
  - Privileged Context Boundary, 219-220
  - Progress Reporting of Disconnected Resource, 461-462
  - Progress Reporting to Client Only, 459-460
  - queues, 342
  - Reference Monitor, 139-140
  - request-response, 435-437
  - Requester-Responder, 438
  - Resource Proxy Gateway, 140-141
  - Role-based Permissions, 145-146
  - scaffolding patterns, 64
  - semaphore, 341
  - Singleton pattern, 329
  - versus templates, code generation, 64-65
  - test-and-set, 342
  - Timer Thread, 360-361
  - Transaction Facade, 411-414
Using Checked Exceptions for All Lifecycle Errors, 484-486
Value Object pattern, 321-322
peekEvent(), 365
peer collusion, 117
peer-to-peer (P2P), 437
performance, 14
performance requirements, 15
performance tests, 45
perimeters, 154-156
permission parameters, 164-168
permissions, 162-164
PersistenceException, 490
Pessimistic Concurrency Control pattern, 386-388
phony COD delivery, 114
physical theft, 111
Pigeon Drop, 113
pipelines, multithreading, 351-352
PKI, 312
managing credentials, 288
planting, 106
platform-level configurations, deployment, 296
platforms, 148
writing privileged code on method policy platforms, 223-225
point-to-point dedicated connections, 252
policies, default policies, 174, 176
policy administration, 275
policy attributes, 165
policy inheritance, 130
policy repositories, authorization services, 175
postconditions, 471
pre-existing transactions, combining, 398-400
preconditions, subclasses, 471
preventing multiple clicks, 365
pride, 112
priority-based task scheduling, 23
privilege, 225
privileged context boundaries, 220
Privileged Context Boundary pattern, 219-220
PrivilegedAction, 222
privileges
encapsulation approaches, 220-221, 223
encapsulation of, 218-220
problem areas
failure modes, 5
security, 4
sound database integration, 6
sound GUI applications, 5
sound transactions, 4
sound web applications, 5
problem transparency, 3
providing, 504
problems, cost of technical problems, 6
process coordination, 344-345
process-specific schemas, transactions, 406-407
processes, 7-8, 536
administrative processes, 162
business processes, 162
levels of processes within organizations, 156-157, 159-162
reliability, 78
resource name resolution processes, 249
producer/consumer pattern, 351
production area testing, 69
programmatic testing, 71-72
progress bars, 458, 461-462
progress events, 453-455
progress messages, 450
capturing implicit context, 457-458
heartbeats, 451, 453
progress bars, 458, 461-462
progress events, 453-455
trustworthiness of status indication, 456-457
Progress Reporting of Disconnected Resource, patterns, 461-462
Progress Reporting to Client Only, patterns, 459-460
prohibition, architectural rules, 56
prohibition-related features, architectural rules, 57
proof of action, 17
proofs, correctness proofs, 52
propagation
  context propagation, 216-217
  of unauthorized information, secure error handling, 188-194
protected resources. See resources protection, reliance on inadequate protection, 114-116
protection against modification, 17
protocol tunneling, 299-300
protocol-based testing, 71-72
protocols, stateful protocols, 247
Prototype Verification System (PVS), 53
  providing
  context, error handling, 497-498, 500, 502
  problem transparency, 504
  sufficient information, error handling, 503-507, 511, 513
psychological acceptability, 310
Pump-and-Dump, 113
PVS (Prototype Verification System), 53
Q
QA tests, 46
quality, applying stringent quality criteria to business components, 470
quality control, 81
queues, 342
quotas, 305
R
race condition, 340
ramp tests, 46
RBAC (role-based access control), secure resource containers, 144-146
realms
  destination realm, 129
  interference, 129-130
  trust, 128-129
reclamation, automated reclamation, 323
recoverability, 30
recoverable errors, 24
recovery, 90-91
recovery testing, 48
recycling logs, 198
reduced access mode, 306
redundancy, 79
  recovery, 90
  verification, 38
reentrancy, 341, 344
Reference Monitor pattern, 139-140
reference monitors, secure resource containers, 138-140
references, 243
  vulnerable references, 242-244
referential integrity, 381-383
regression, N+1 tests, 45
regression testing, 47, 65, 472
relational databases, problems with OO applications, 393
relational mapping, 392
relationships
  compositional relationships, 317
  trust, 124
release components, 559-560
release testing, software releases, 561-562
ReleaseLock(), 346
releases, building software releases, 560-561
reliability, 3, 8, 14, 79
  design principles, 83
  processes, 78
  writing test plans, 71-74
  XP, 550
reliability categories, writing reliability requirements, 24-25
reliability requirements, 15
  writing, 22
  adding derived requirements that affect reliability, 25-27, 29
  business objectives, 23
  reliability categories, 24-25
reliability-related risk, 82-83
reliance, 114-116
remediation
  application security case studies, 580-581
  manageability case studies, 588-589
  transactional integrity case study, 572-574
renege(), 355
replay, 97
replication, 430
reporting resource usage, 462
repudiation, 104
request-response, 433-434
  blocked registrant, 438-440
  design patterns, 435-437
  unreachable servers, 437-438
Requester-Responder pattern, 438
requester/servicer pattern, 351
requirements
  derived requirements, 538
  failure-response requirements
    samples, 617-620, 622
    writing, 29
  log requirements, 195
  methodology-based requirements, 22
  reliability requirements. See reliability requirements
  security requirements, 16-20
    samples, 20, 22
  verifying conformance to, 34
residual artifacts, exploits of, 106
resource access method, 149
resource connections, 244-247
resource containers. See containers
resource definition layer, 150
resource hierarchies, 132
resource name resolution processes, 249
Resource Proxy Gateway pattern, 140-141
resource resolution, 247
resource-level interfaces, transactional integrity case study, 570-571
resources, 135
  authorization model, 176-178
  avoiding resource interference, 447-448
  binding resources, 248
container-level resources, 157
  ensuring atomic access to resources, 184-185
  identifying, 248
  interface design, 178-181
    simplicity, 183
    type safety, 181-183
  managed resources, 448
  native resources, 149, 255-256
  non-transactional resources, 394-395
  perimeters, 154-156
  reporting usage, 462
  scoping, 244
  secure administrative resources, 254
  secure non-run-time resources, 253
  transaction-level resources, 156
  types and layers, 148
  unmanaged resource, 448
responders, attacking, 115
responsibilities, 333-334
responsibility-driven design, 333
restoration, 297
ResumeTask(), 346
retrieving crash artifacts, 100
reusable components, 238-242
reuse, human intervention, 44
reverse cache, 426
reviewing design, universal methodological principles, 546
revocation of permissions, application security case studies, 579
risk
  administrative functions, 81
  cloning risks, 323
  high-risk applications, 80-82
  reliability-related risk, 82-83
  security-related risk, 80-82
risk assessment, 83-84
robot testing, 71-72
robust error handling, 495-496
rocks-in-the-box, 113
role parameterization, 169
role separation, 270-272
  access rights, 275
  compartmentalizing roles, 274
convenience versus safety, 276-278
enforcing separation of duties, 274-275
multiple roles, 273
traceability functions, 275
role-based access control (RBAC), secure resource containers, 144-146
Role-based Permissions patterns, 145-146
roles, 127
access rights, 275
administrative roles, 270
application roles, 270
compartmentalizing, 274
constraints, 271
derived roles, 271-272
employing different credentials for each role, 278
extensible roles, 271-272
multiple roles, 273
mutual exclusion of roles, 274
secure communication in multi-role operations, 276
server roles, 278-279
sub-roles, 271
to consider for components, 334
rule-based access control, secure resource containers, 147
rules
authorization rules, 176
business rules
auditability, 469
versus security rules, 162
security authorization rules, 469
transaction authorization rules, 159
run-time scoping, architectural rules, 61
scaffolding patterns, 64
scalability tests, 45
scheduling priority-based task scheduling, 23
Schell, Roger, 268
schemas, process-specific schemas, 406-407
scope, contextual scope, 214-216
scope attackers, 110
scoped locking, 343
scoping
boundaries provided by Java, 226
object-level scoping, 227
resources, 244
run-time scoping, 61
scoping mechanisms, 247-248
script injections, 107
secrets, 281
user identity, 289-291
secure administrative resources, 254
secure configuration data, 254
secure defaults, 174
secure error handling, 185-186
backup systems, 194
code that performs authorization, 187-188
propagation of unauthorized information, 188-194
secure logging, 194-195
log design, 196-201
log requirements, 195
secure non-run-time resources, 253
secure operations, designing, 297-302
secure resource containers, 136-138
ACL, 142
capability, 141-142
entitlements, 148
evidence-based
authorization, 147
gateways, 140-141
reference monitors, 138-140
role-based access control, 144-146
safe conversion, 183
salami, 118
salting cash, 118
sandbox model, secure resource containers, 143-144
saturation, 98-99
saturation-resistant, 517
rule-based access control, 147
sandbox model, 143-144
security, 8, 15
accountability for, 18
algorithms, 312
architectural rules. See architectural rules
authentication. See authentication
bank examiner frauds, 309
caching, 427, 429
clearance levels, 268
court involvement, 18
denial of service attacks,
303-305, 307
good directions, 311
intrusion detection, 303
intrusion response, 307
monitoring, 302-303
PKI, 312
problem areas, 4
reduced access mode, 306
relaxing preconditions, 471
restoration, 297
risk, 80-82
risk evaluation methodologies, 19
tests, 19
transporting data, 282
usability, 311-312
XP, 550
security authorization rules, 469
security certification, 20
security features, 16
security levels, 19
security managers
architectural rules, 61
dynamic conditions, 173
security realms, 128-129
interference, 130
security requirements, 16-20
samples, 20, 22
security rules versus business rules, 162
security-related features, 310
semaphore, 341
sensitive data, erasing, 257
敏感 operations
delaying affect of, 277
enforcing one action at a time, 277
separation of duties, 274
separation of privilege, 274
serialization, 341
server actions, correlating with user actions, 498
server roles, 278-279
servers, unreachable servers
(request-response), 437-438
service-level agreement (SLA), 15
ServiceThread, 357
session endpoints, 203
session timeouts, 288
sessions, endpoints, 203
sharing information across domains, 236
shorting, 115
shutdown, 449
sign-on, implementing single sign-on, 278
simplicity, resource interface design, 183
simplifying business rules, 469
Singleton pattern, 329
SLA (service-level agreement), 15
smoke, sanity tests, 45
sniffing, 96
soak, volume tests, 45
SOAP, 440
social engineering, 110-111
breaking prior trust, 117-118
emotional pressure, 112-113
haste, 113-114
instilling undeserved trust, 116-117
physical theft, 111
reliance on inadequate protection, 114-116
trusted resource attacks, 118
software
causes of failure, 2
testing, 36
software releases, 559
  building releases, 560-561
configuration management of release components, 559-560
identifying release components, 559
release testing, 561-562
user acceptance, 562
sound database integration, problems, 6
sound GUI applications, problem areas, 5
sound transactions, 4
sound web applications, problem areas, 5
Spanish Prisoner, 113
specific argument types, 336
spoofing, 101-103
spyware, 295, 301
SQL, isolation levels, 384
stack overflow, 182
staging area testing, 69
startup, 449
stateful protocols, 247
static analysis, 52
  architectural rules, 60
static lifetime objects, 329-331
static objects, bi-directional dependencies, 330
static resource paths, 252
static tests, 46
status, thread status monitoring, 494-495
status indication, trustworthiness, 456-457
stolen credentials, 116
storing
  data, securely, 283-284
  unsecured data in secure locations, 282
strategies, need to know strategies, 268
stress testing, 45, 48-50, 67
structural tests, 45
sub-classes, 270
sub-roles, 271
sub-classes, preconditions, 471
survivability, 88
SUT (system under test), 45
system administration, deployment, 295
system availability, 24
system component failures, 30
system under test (SUT), 45
system-level events, 29
systems, external systems, 336

T
tampering, 284
tasks, priority-based task scheduling, 23
TCSEC (Trusted Computer Security Evaluation Criteria), 467
TDD (test-driven development), 47-48
technical attacks, 94
  brute force guessing, 108
  circumvention, 103
  coordination interference, 100
  denial of service, 99
  embedded attacks, 106-107
  environmental interference, 101
  exploit of incomplete traceability, 104
  exploit of incomplete validation, 105
  exploitation of non-atomicity, 99
  exploits of exposure of internal state, 105-106
  exploits of incomplete authentication or authorization, 105
  exploits of residual artifacts, 106
  forced crash and retrieval of crash artifacts, 100
  forced restart, forced re-install, 100
  hijacking, 103
  interception, 96
  interruption, 98
  man-in-the-middle attacks, 96
  modification in place or in transit, 97
  namespace attacks, 108
  pattern matching, 108
  replay, 97
  saturation and delay, 98-99
  scope attacks, 110
  spoofing, 101-103
  trap doors, 104
  trusted resource attacks, 109-110
  weak links as gateways, 108
technical glitches, 2
technical problems, cost of, 6
template definition languages, 54
templates
code generation, 63
versus patterns, code generation, 64-65
test plans, writing for reliability and failure response, 71-74
test suites, coverage, 45
test-and-set, 342
test-driven development (TDD), 47
testing, 35-36
acceptance testing, 68, 70
assurance requirements, XP, 556
assurance testing, 70
comprehensive testing, 65
concurrency testing, 49,
368-376
design verification and elaboration,
45-47
behavioral test specifications and regression testing, 47
certification testing, 50-51
failure-response testing, 48-50
stress testing, 48-50
TDD, 47-48
failure testing, 67, 70
hardware design, 37
integration testing, 67
levels of verification, 66, 68
manual testing, 71
production area testing, 69
programmatic testing, 71-72
protocol-based testing, 71-72
regression testing, 65, 472
robot testing, 71-72
software, 36
staging area testing, 69
stress testing, 67
transactions, 419-421
types of, 45-46
unit testing, 65-66
XP, 548
tests, security, 19
theft, 273
physical theft, 111
third-party components, 336
third-party systems, 336
thread interruptions, 355-359
thread timeout mechanisms,
designing, 359
thread-safe, 344
Threadlocal reference, 501
threads, 343
thread status monitoring,
494-495
worker threads, 347
threat analysis, name resolution, 250
threats, 83
time bombs, 107
time of check to time of use (TOCTTOU) attacks, 99
timeouts, credentials, 288
Timer Thread pattern, 360-361
TOCTTOU (time of check to time of use) attacks, 99
tolerating unreliable systems, 3
tools
Jtest, 58
Macker, 59, 61
top-down tests, 46
Tower Records, 18
traceability, 64, 314
exploits of incomplete traceability, 104
traceability functions, 275
transaction aggregation, 403-406
transaction authorization rules, 159
Transaction Façade pattern, 411-414
transaction isolation, 383-385
transaction layering, 401-402
transaction managers, 396
transaction-level resources, 156
transactional integrity, 379-381
case studies, 567, 569-570
analysis, 570
failure testing, 572
multithreading, 572
remediation, 572-574
resource-level interfaces, 570-571
web application transaction encapsulation, 571-572
factors contributing to, 380
transactions
application-defined locking, 407-409
combining, 396
kinds of transaction composition, 396-398
pre-existing transactions, 398-400
process-specific schemas, 406-407
transaction aggregation, 403-406
transaction layering, 401-402
composite objects, 392-394
composite transactions, 396
consistency, 385-390
intra-transaction consistency, 390-392
error handling, 415
non-successful conditions, 416-417
resources that do not reliably timeout, 415-418
layer access rules, 411-414
non-transactional resources, 394-395
testing, 419-421
validation strategies, 419-421
workflows, 407-409
transparency, 30
implementation, 468-470
transport systems, 282
transporting data, 282
trap doors, 104
Troj/BankAsh virus, 107
trojan horses, 107, 253
truck load scam, 113
trust, 121-124, 208
approval-based trust, 126-127
breaking prior trust, 117-118
instilling undeserved trust, 116-117
managing, 128
hierarchies of trust, 130-133
security realms, 128-130
modeling, 125
trusted resource attacks, 118
trusted components, deployment, 294
Trusted Computer Security Evaluation Criteria (TCSEC), 467
typed resource attacks, 109-110
trustworthiness
accessing, 207
of status indication, 456-457
type conversions, 182
type safety, resource interface design, 181-183
types of testing, 45-46

U
UML (Universal Modeling Language), 41, 84
under-ring, 118
unit testing, 66
levels of verification, 65
XP, 548
unit tests, 45
universal methodological principles, 540-542
architecture that explains how application meets assurance requirements, 544
auditable business rules, 542-543
documentation, 545-546
full configuration management of all artifacts, 547
identifying quality-critical codebase, 543
quality of implementation, 544
reviewing design, 546
Universal Modeling Language (UML), 41, 84
unmanaged resources, 448
unsecured data, storing in secure locations, 282
updates
- caching, 427
- forward cache with synchronous updates, 428-429

usability, 307-308
- alerting users, 309-310
- algorithms, 312
- authentication, 314
- authorization, 313
- inconsistencies, 313
- instructional messages, 308
- security features, 311-312
- traceability, 314

usability tests, 46
use-case analysis, 157
user acceptance, software releases, 562
user actions, correlating with server actions, 498
user identity, 289-291
user interface programs, 156
user status, error handling, 515
users
- alerting, 309-310, 514-515
- informing them of status, 310

Using Checked Exceptions for All Lifecycle Errors pattern, 484-486

validation
- concurrency, 368-376
- exploits of incomplete validation, 105
- of inputs, 335
- for non-repudiation purposes, 214
- transactions, 419-421
- XP, 555

Value Object pattern, 321-322
value objects, 320-321
verifiability, designing for, 86-87
verification
- of compliance with design, 38
- levels of verification, 65-66, 70
testing, 66, 68
redundancy, 38

verifying
- code, against architectural rules, 58
- conformance to requirements, 34

VHDL, 228
virtual private networks (VPNs), 109
viruses, Trojan/BankAsh, 107
visible context, 203
VPNs (virtual private networks), 109
vulnerabilities
- deciding what to encrypt, 284-286
- embedded data, 284
- storing unsecured data in secure locations, 282
- transport systems, 282
vulnerable references, 242-244

weak links as gateways, 108
web application transaction encapsulation, transactional integrity case study, 571-572
Web scripting languages, concurrency, 345
white box tests, 46
worker threads, 347
workflow, 160-161
workflow processes, 158-159
workflow tests, 46
workflows, 407-409
writing
- corporate assurance guidelines, 557-558
- failure-response requirements, 29-31
- privileged code on method policy platforms, 223-225
- reliability requirements, 22
- adding derived requirements that affect reliability, 25-27, 29
- business objectives, 23
- reliability categories, 24-25
- test plans for reliability and failure-response, 71-74
XP (Extreme Programming), 547
  code ownership, 550
  configuration control, 553
  documentation, 550
  frequent uncontrolled change, 549-550
  modifications to, 551
    analysis functions, 551-552
    architecture, 552-553
    checklists, 556
    customer concept, 551
    rules about who works on sensitive
      components, 554
    systems interactions, 553
    testing assurance requirements, 556
    training for assurance-capable
      programmers, 554
    validation, 555
  role of design, 549
  security, 550
  unit testing, 548